

Introduction to Econometric Production Analysis with R (Seventh Draft Version)

Arne Henningsen

with contributions of Tomasz G. Czekaj

Department of Food and Resource Economics
University of Copenhagen

November 11, 2025

Introduction to Econometric Production Analysis with R

© 2025 by [Arne Henningsen](#)

is licensed under [CC BY-NC-SA 4.0](#) 

Foreword

This is a collection of my lecture notes for various courses in the field of econometric production analysis. These lecture notes are still incomplete and may contain many typos, errors, and inconsistencies. Please report any problems to arne.henningsen@gmail.com. I am grateful to Tomasz G. Czekaj who drafted and revised some parts of these lecture notes and to my former students who helped me to improve my teaching and these notes through their questions, suggestions, and comments. Finally, I thank the R community for providing so many excellent tools for econometric production analysis.

November 11, 2025

Arne Henningsen

How to cite these lecture notes:

Henningsen, Arne (2025): Introduction to Econometric Production Analysis with R. Collection of Lecture Notes. 7th Draft Version. Department of Food and Resource Economics, University of Copenhagen. Available at Zenodo (<http://doi.org/10.5281/zenodo.11093657>) and Leanpub (<http://leanpub.com/ProdEconR/>).

History

First Draft Version (March 9, 2015)

- Initial release at Leanpub.com

Second Draft Version (February 2, 2018)

- added a chapter on distance functions
- corrected the equation for calculating direct elasticities of substitution
- a large number of further additions, improvements, and corrections

Third Draft Version (January 29, 2019)

- corrected several typos and minor errors
- some minor additions and improvements

Fourth Draft Version (May 27, 2019)

- added three sections about homotheticity of cost functions (sections [3.1.4](#), [3.2.6](#), and [3.5.6](#))
- added a section about cost functions with multiple outputs (section [3.1.6](#))
- added a section about cost functions with technical change (section [9.4](#))
- added a proof that cost minimisation and profit maximisation imply that the output elasticities of the inputs divided by the elasticity of scale are equal to their cost shares (section [2.1.11](#))
- added a proof that revenue maximisation and profit maximisation imply that the distance elasticities of the outputs derived from an output distance function are equal to their revenue shares (section [8.1.1.4](#))
- added a proof that cost minimisation and profit maximisation imply that the negative distance elasticities of the inputs divided by the elasticity of scale derived from an output distance function are equal to their cost shares (section [8.1.1.4](#))
- added an approach for obtaining unobserved (shadow) prices of outputs from an output distance function (section [8.1.1.5](#))
- a few minor additions and improvements

Fifth Draft Version (October 28, 2020)

- added two subsections that discuss the suitability of the production function and the cost function, respectively, in econometric applications (sections [2.1.13](#) and [3.1.7](#))
- added a comparison of the observed cost shares with the cost shares that would minimize costs according to an estimated Cobb-Douglas production function in section [2.4.14](#)
- added a section about imposing monotonicity on the Translog input distance function (section [8.5.6](#))
- several minor corrections and improvements

Sixth Draft Version (April 30, 2024)

- several small corrections, improvements, and extensions

Seventh Draft Version (November 11, 2025)

- several small corrections, improvements, and extensions

Contents

1	Introduction	14
1.1	Objectives of the course and the lecture notes	14
1.2	An extremely short introduction to R	14
1.2.1	Some commands for simple calculations	15
1.2.2	Creating objects and assigning values	16
1.2.3	Vectors	17
1.2.4	Simple functions	18
1.2.5	Comparing values and Boolean values	19
1.2.6	Data sets (“data frames”)	20
1.2.7	Functions	22
1.2.8	Simple graphics	22
1.2.9	Other useful commands	23
1.2.10	Extension packages	23
1.2.11	Reading data into R	24
1.2.12	Linear regression	24
1.3	R packages	28
1.4	Data sets	31
1.4.1	French apple producers	31
1.4.1.1	Description of the data set	31
1.4.1.2	Abbreviating name of data set	32
1.4.1.3	Calculation of input quantities	33
1.4.1.4	Calculation of total costs, variable costs, and cost shares	33
1.4.1.5	Calculation of profit and gross margin	34
1.4.2	Rice producers on the Philippines	34
1.4.2.1	Description of the data set	34
1.4.2.2	Mean-scaling quantities	35
1.4.2.3	Logarithmic mean-scaled quantities	35
1.4.2.4	Mean-adjusting the time trend	36
1.4.2.5	Total costs and cost shares	36
1.4.2.6	Specifying panel structure	36
1.5	Mathematical and statistical methods	37
1.5.1	Exponentiation	37
1.5.2	Logarithms	37

1.5.3	Partial derivatives	38
1.5.4	Aggregating quantities	38
1.5.5	Concave and convex functions	40
1.5.6	Quasiconcave and quasiconvex functions	41
1.5.7	Delta method	42
2	Primal Approach: Production Function	43
2.1	Theory	43
2.1.1	Production function	43
2.1.2	Average products	43
2.1.3	Total factor productivity	43
2.1.4	Marginal products	44
2.1.5	Output elasticities	44
2.1.6	Elasticity of scale and most productive scale size	44
2.1.7	Marginal rates of technical substitution	45
2.1.8	Relative marginal rates of technical substitution	46
2.1.9	Elasticities of substitution	46
2.1.9.1	Direct elasticities of substitution	46
2.1.9.2	Allen elasticities of substitution	46
2.1.9.3	Morishima elasticities of substitution	47
2.1.10	Profit maximization	48
2.1.11	Cost minimization	49
2.1.12	Derived input demand functions and output supply functions	51
2.1.12.1	Derived from profit maximization	51
2.1.12.2	Derived from cost minimization	52
2.1.13	Suitability of the production function for econometric applications	52
2.2	Productivity measures	53
2.2.1	Average products	53
2.2.2	Total factor productivity	56
2.3	Linear production function	58
2.3.1	Specification	58
2.3.2	Estimation	59
2.3.3	Properties	59
2.3.4	Predicted output quantities	60
2.3.5	Marginal products	61
2.3.6	Output elasticities	62
2.3.7	Elasticity of scale	64
2.3.8	Marginal rates of technical substitution	67
2.3.9	Relative marginal rates of technical substitution	68
2.3.10	First-order conditions for profit maximization	69

2.3.11	First-order conditions for cost minimization	71
2.3.12	Derived input demand functions and output supply functions	73
2.4	Cobb-Douglas production function	74
2.4.1	Specification	74
2.4.2	Estimation	74
2.4.3	Properties	75
2.4.4	Predicted output quantities	75
2.4.5	Output elasticities	76
2.4.6	Marginal products	77
2.4.7	Elasticity of scale	79
2.4.8	Marginal rates of technical substitution	81
2.4.9	Relative marginal rates of technical substitution	82
2.4.10	First and second partial derivatives	83
2.4.11	Elasticities of substitution	84
2.4.11.1	Direct elasticities of substitution	84
2.4.11.2	Allen elasticities of substitution	85
2.4.11.3	Morishima elasticities of substitution	89
2.4.12	Quasiconcavity	90
2.4.13	First-order conditions for profit maximization	91
2.4.14	First-order conditions for cost minimization	93
2.4.15	Derived input demand functions and output supply functions	96
2.4.16	Derived input demand elasticities	99
2.5	Quadratic production function	101
2.5.1	Specification	101
2.5.2	Estimation	101
2.5.3	Properties	103
2.5.4	Predicted output quantities	104
2.5.5	Marginal products	105
2.5.6	Output elasticities	107
2.5.7	Elasticity of scale	107
2.5.8	Marginal rates of technical substitution	108
2.5.9	Relative marginal rates of technical substitution	110
2.5.10	Quasiconcavity	112
2.5.11	Elasticities of substitution	114
2.5.11.1	Direct elasticities of substitution	114
2.5.11.2	Allen elasticities of substitution	116
2.5.11.3	Comparison of direct and Allen elasticities of substitution	118
2.5.12	First-order conditions for profit maximization	119
2.5.13	First-order conditions for cost minimization	119

2.6	Translog production function	123
2.6.1	Specification	123
2.6.2	Estimation	123
2.6.3	Statistical significance of individual inputs	125
2.6.4	Properties	129
2.6.5	Predicted output quantities	130
2.6.6	Output elasticities	131
2.6.7	Marginal products	132
2.6.8	Elasticity of scale	133
2.6.9	Marginal rates of technical substitution	134
2.6.10	Relative marginal rates of technical substitution	136
2.6.11	Second partial derivatives	138
2.6.12	Quasiconcavity	138
2.6.13	Elasticities of substitution	140
2.6.13.1	Direct elasticities of substitution	140
2.6.13.2	Allen elasticities of substitution	143
2.6.13.3	Comparison of direct and Allen elasticities of substitution	146
2.6.14	Mean-scaled quantities	147
2.6.15	First-order conditions for profit maximization	150
2.6.16	First-order conditions for cost minimization	152
2.7	Evaluation of different functional forms	155
2.7.1	Goodness of fit	156
2.7.2	Test for functional form misspecification	157
2.7.3	Theoretical consistency	158
2.7.4	Plausible estimates	159
2.7.5	Summary	160
2.8	Non-parametric production function	161
3	Dual Approach: Cost Functions	167
3.1	Theory	167
3.1.1	Cost function	167
3.1.2	Properties of the cost function	167
3.1.3	Cost flexibility and elasticity of size	167
3.1.4	Homotheticity of cost functions	168
3.1.5	Short-run cost functions	168
3.1.6	Cost functions with multiple outputs	169
3.1.7	Suitability of the cost function for econometric applications	170
3.2	Cobb-Douglas cost function	171
3.2.1	Specification	171
3.2.2	Estimation	171

3.2.3	Properties	172
3.2.4	Estimation with linear homogeneity in input prices imposed	173
3.2.5	Checking concavity in input prices	177
3.2.6	Homotheticity	182
3.2.7	Optimal input quantities	183
3.2.8	Optimal cost shares	184
3.2.9	Derived input demand functions	184
3.2.10	Derived input demand elasticities	186
3.2.11	Cost flexibility and elasticity of size	188
3.2.12	Marginal costs, average costs, and total costs	188
3.3	Cobb-Douglas short-run cost function	192
3.3.1	Specification	192
3.3.2	Estimation	192
3.3.3	Properties	193
3.3.4	Estimation with linear homogeneity in input prices imposed	193
3.4	Cobb-Douglas cost function with multiple outputs	195
3.4.1	Specification	195
3.4.2	Estimation	195
3.4.3	Properties	196
3.4.4	Estimation with linear homogeneity in input prices imposed	196
3.4.5	Cost flexibility and elasticity of size	197
3.5	Translog cost function	197
3.5.1	Specification	197
3.5.2	Estimation	197
3.5.3	Linear homogeneity in input prices	199
3.5.4	Estimation with linear homogeneity in input prices imposed	201
3.5.5	Cost flexibility and elasticity of size	206
3.5.6	Homotheticity	208
3.5.7	Marginal costs and average costs	210
3.5.8	Derived input demand functions	213
3.5.9	Derived input demand elasticities	216
3.5.10	Theoretical consistency	221
3.6	Translog cost function with multiple outputs	223
3.6.1	Specification	223
3.6.2	Properties	223
4	Dual Approach: Profit Function	224
4.1	Theory	224
4.1.1	Profit functions	224
4.1.2	Short-run profit functions	224

4.2	Graphical illustration of profit and gross margin	224
4.3	Cobb-Douglas profit function	226
4.3.1	Specification	226
4.3.2	Estimation	227
4.3.3	Properties	227
4.3.4	Estimation with linear homogeneity in all prices imposed	228
4.3.5	Checking convexity in all prices	230
4.3.6	Predicted profit	236
4.3.7	Optimal profit shares	237
4.3.8	Derived output supply and input demand functions	239
4.3.9	Derived output supply and input demand elasticities	239
4.4	Cobb-Douglas short-run profit function	241
4.4.1	Specification	241
4.4.2	Estimation	241
4.4.3	Properties	242
4.4.4	Estimation with linear homogeneity in all prices imposed	242
4.4.5	Returns to scale	244
4.4.6	Shadow prices of quasi-fixed inputs	244
5	Efficiency Measures	246
5.1	Technical efficiency	246
5.1.1	Output-oriented technical efficiency with one output	246
5.1.2	Input-oriented technical efficiency with one input	246
5.1.3	Output-oriented technical efficiency with two or more outputs	246
5.1.4	Input-oriented technical efficiency with two or more inputs	247
5.2	Allocative efficiency, revenue efficiency, cost efficiency	247
5.2.1	Output-oriented allocative efficiency and revenue efficiency	247
5.2.2	Input-oriented allocative efficiency and cost efficiency	248
5.3	Profit efficiency	249
5.4	Scale efficiency	249
6	Stochastic Frontier Analysis	250
6.1	Stochastic production frontiers	250
6.1.1	Specification	250
6.1.1.1	Marginal products and output elasticities in SFA models	252
6.1.2	Skewness of residuals from OLS estimations	252
6.1.3	Cobb-Douglas stochastic production frontier	253
6.1.3.1	Estimation	253
6.1.3.2	Variance parameters	254
6.1.3.3	Statistical tests for inefficiencies	256

6.1.3.4	Obtaining technical efficiency estimates	257
6.1.3.5	Truncated normal distribution of the inefficiency term	259
6.1.4	Translog production frontier	262
6.1.4.1	Estimation	262
6.1.4.2	Statistical test for inefficiencies	263
6.1.4.3	Testing against the Cobb-Douglas functional form	263
6.1.4.4	Obtaining technical efficiency estimates	264
6.1.4.5	Truncated normal distribution of the inefficiency term	265
6.1.5	Translog production frontier with mean-scaled variables	268
6.2	Stochastic cost frontiers	269
6.2.1	Specification	269
6.2.2	Skewness of residuals from OLS estimations	270
6.2.3	Estimation of a Cobb-Douglas stochastic cost frontier	271
6.2.4	Short-run cost frontiers	273
6.2.5	Profit frontiers	275
6.3	Analyzing the effects of z variables	275
6.3.1	Production functions with z variables	276
6.3.2	Production frontiers with z variables	277
6.3.3	Efficiency effects production frontiers	280
6.4	Decomposition of cost efficiency	284
7	Data Envelopment Analysis (DEA)	288
7.1	Preparations	288
7.2	DEA with input-oriented efficiencies	288
7.3	DEA with output-oriented efficiencies	291
7.4	DEA with “super efficiencies”	291
7.5	DEA with graph hyperbolic efficiencies	291
8	Distance Functions	292
8.1	Theory	292
8.1.1	Output distance functions	292
8.1.1.1	Properties	292
8.1.1.2	Distance elasticities	298
8.1.1.3	Elasticity of scale	299
8.1.1.4	Optimising behaviour and distance elasticities	299
8.1.1.5	Obtaining unobserved output prices	301
8.1.2	Input distance functions	302
8.1.2.1	Properties	302
8.1.2.2	Distance elasticities	308
8.1.2.3	Elasticity of scale	309

8.2	Cobb-Douglas output distance function	309
8.2.1	Specification	309
8.2.2	Estimation	310
8.2.3	Distance elasticities	312
8.2.4	Elasticity of scale	313
8.2.5	Properties	313
8.2.5.1	Non-increasing in input quantities	313
8.2.5.2	Non-decreasing in output quantities	313
8.2.5.3	Quasiconvex in input quantities	314
8.2.5.4	Convex in output quantities	316
8.2.6	Efficiency estimates	318
8.3	Translog output distance function	319
8.3.1	Specification	319
8.3.2	Estimation	319
8.3.3	Distance elasticities	324
8.3.4	Elasticity of scale	326
8.3.5	Properties	327
8.3.5.1	Non-increasing in input quantities	327
8.3.5.2	Non-decreasing in output quantities	327
8.3.5.3	Quasiconvex in input quantities	328
8.3.5.4	Convex in output quantities	332
8.3.6	Efficiency estimates	334
8.4	Cobb-Douglas input distance function	335
8.4.1	Specification	335
8.4.2	Estimation	336
8.4.3	Distance elasticities	338
8.4.4	Elasticity of scale	339
8.4.5	Properties	339
8.4.5.1	Non-decreasing in input quantities	339
8.4.5.2	Non-increasing in output quantities	339
8.4.5.3	Concave in input quantities	340
8.4.5.4	Quasiconcave in output quantities	342
8.4.6	Efficiency estimates	344
8.5	Translog input distance function	345
8.5.1	Specification	345
8.5.2	Estimation	345
8.5.3	Distance elasticities	350
8.5.4	Elasticity of scale	352
8.5.5	Efficiency estimates	352

8.5.6	Imposing monotonicity	353
9	Panel Data and Technological Change	359
9.1	Average production functions with technological change	359
9.1.1	Cobb-Douglas production function with technological change	359
9.1.1.1	Pooled estimation of the Cobb-Douglas production function with technological change	360
9.1.1.2	Panel data estimations of the Cobb-Douglas production function with technological change	361
9.1.2	Translog production function with constant and neutral technological change	366
9.1.2.1	Pooled estimation of the Translog production function with constant and neutral technological change	366
9.1.2.2	Panel-data estimations of the Translog production function with constant and neutral technological change	368
9.1.3	Translog production function with non-constant and non-neutral technological change	373
9.1.3.1	Pooled estimation of a translog production function with non-constant and non-neutral technological change	374
9.1.3.2	Panel-data estimations of a Translog production function with non-constant and non-neutral technological change	379
9.2	Frontier production functions with technological change	385
9.2.1	Cobb-Douglas production frontier with technological change	385
9.2.1.1	Time-invariant individual efficiencies	386
9.2.1.2	Time-variant individual efficiencies	389
9.2.1.3	Observation-specific efficiencies	393
9.2.1.4	Fixed effects in the production frontier	395
9.2.2	Translog production frontier with constant and neutral technological change	398
9.2.2.1	Observation-specific efficiencies	398
9.2.3	Translog production frontier with non-constant and non-neutral technological change	401
9.2.3.1	Observation-specific efficiencies	401
9.2.4	Decomposition of productivity growth	404
9.3	Analyzing productivity growths with Data Envelopment Analysis (DEA)	405
9.4	Cost functions with technical change	406
9.4.1	Translog cost function with constant and neutral technical change	406
9.4.2	Translog cost function with non-constant and non-neutral technical change	406

1 Introduction

1.1 Objectives of the course and the lecture notes

Knowledge about production technologies and producer behavior is important for politicians, business organizations, government administrations, financial institutions, the EU, and other national and international organizations who desire to know how contemplated policies and market conditions can affect production, prices, income, and resource utilization in agriculture as well as in other industries. The same knowledge is relevant in consultancy of single firms who also want to compare themselves with other firms and their technology with the best practice technology.

The participants of my courses in the field of econometric production analysis will obtain relevant theoretical knowledge and practical skills so that they can contribute to the knowledge about production technologies and producer behavior. After completing my courses in the field of econometric production analysis, the students should be able to:

- use econometric production analysis and efficiency analysis to analyze various real-world questions,
- interpret the results of econometric production analyses and efficiency analyses,
- choose a relevant approach for econometric production and efficiency analysis, and
- critically evaluate the appropriateness of a specific econometric production analysis or efficiency analysis for analyzing a specific real-world question.

These lecture notes focus on practical applications of econometrics and microeconomic production theory. Hence, they complement textbooks in microeconomic production theory (rather than substituting them).

1.2 An extremely short introduction to R

Many tutorials for learning R are freely available on-line, e.g. the official “Introduction to R” (<http://cran.r-project.org/doc/manuals/r-release/R-intro.pdf>) or the many tutorials listed in the category “Contributed Documentation” (<http://cran.r-project.org/other-docs.html>). Furthermore, many good books are available, e.g. “A Beginner’s Guide to R” (Zuur, Ieno, and Meesters, 2009), “R Cookbook” (Teetor, 2011), or “Applied Econometrics with R” (Kleiber and Zeileis, 2008).

1.2.1 Some commands for simple calculations

R is my favorite “pocket calculator”...

```
> 2 + 3
```

```
[1] 5
```

```
> 2 - 3
```

```
[1] -1
```

```
> 2 * 3
```

```
[1] 6
```

```
> 2 / 3
```

```
[1] 0.6666667
```

```
> 2^3
```

```
[1] 8
```

R uses the standard order of evaluation (as in mathematics). One can use parenthesis (round brackets) to change the order of evaluation.

```
> 2 + 3 * 4^2
```

```
[1] 50
```

```
> 2 + ( 3 * ( 4^2 ) )
```

```
[1] 50
```

```
> ( ( 2 + 3 ) * 4 )^2
```

```
[1] 400
```

In R, the hash symbol (#) can be used to add comments to the code, because the hash symbol and all following characters in the same line are ignored by R.

```
> sqrt(2) # square root
```

```
[1] 1.414214
```

```
> 2^(1/2) # the same
```

```
[1] 1.414214

> 2^0.5    # also the same

[1] 1.414214

> log(3)   # natural logarithm

[1] 1.098612

> exp(3)   # exponential function

[1] 20.08554
```

The commands can span multiple lines. They are executed as soon as the command can be considered as complete.

```
> 2 +
+   3

[1] 5

> ( 2
+   +
+   3 )

[1] 5
```

1.2.2 Creating objects and assigning values

```
> a <- 2
> a

[1] 2

> b <- 3
> b

[1] 3

> a * b

[1] 6
```

Initially, the arrow symbol (`<-`, consistent of a “smaller than” sign and a dash) was used to assign values to objects. However, in recent versions of R, also the equality sign (`=`) can be used for this.


```
> a = 4
```

```
> a
```

```
[1] 4
```

```
> b = 5
```

```
> b
```

```
[1] 5
```

```
> a * b
```

```
[1] 20
```

In these lecture notes, I stick to the traditional assignment operator, i.e. the arrow symbol (`<-`).

Please note that R is case-sensitive, i.e. R distinguishes between upper-case and lower-case letters. Therefore, the following commands return error messages:

```
> A # NOT the same as "a"
```

```
> B # NOT the same as "b"
```

```
> Log(3) # NOT the same as "log(3)"
```

```
> LOG(3) # NOT the same as "log(3)"
```

1.2.3 Vectors

```
> v <- 1:4 # create a vector with 4 elements: 1, 2, 3, and 4
```

```
> v
```

```
[1] 1 2 3 4
```

```
> 2 + v # adding 2 to each element
```

```
[1] 3 4 5 6
```

```
> 2 * v # multiplying each element by 2
```

```
[1] 2 4 6 8
```

```
> log( v ) # the natural logarithm of each element
```

```
[1] 0.0000000 0.6931472 1.0986123 1.3862944
```

```
> w <- c( 2, 4, 8, 16 ) # concatenate 4 numbers to a vector
```

```
> w
```

```
[1] 2 4 8 16

> v + w # element-wise addition

[1] 3 6 11 20

> v * w # element-wise multiplication

[1] 2 8 24 64

> v %*% w # scalar product (inner product)

      [,1]
[1,]    98

> w[2] # select the second element

[1] 4

> w[c(1,3)] # select the first and the third element

[1] 2 8

> w[2:4] # select the second, third, and fourth element

[1] 4 8 16

> w[-2] # select all but the second element

[1] 2 8 16

> length( w )

[1] 4
```

1.2.4 Simple functions

```
> sum( w )

[1] 30

> mean( w )

[1] 7.5

> median( w )
```

```
[1] 6
> min( w )
[1] 2
> max( w )
[1] 16
> which.min( w )
[1] 1
> which.max( w )
[1] 4
```

1.2.5 Comparing values and Boolean values

```
> a == 2
[1] FALSE
> a != 2
[1] TRUE
> a > 4
[1] FALSE
> a >= 4
[1] TRUE
> w > 3
[1] FALSE TRUE TRUE TRUE
> w == 2^(1:4)
[1] TRUE TRUE TRUE TRUE
> all.equal( w, 2^(1:4) )
[1] TRUE
> w > 3 & w < 6 # ampersand = and
[1] FALSE TRUE FALSE FALSE
> w < 3 / w > 6 # vertical line = or
[1] TRUE FALSE TRUE TRUE
```

1.2.6 Data sets (“data frames”)

The data set “women” is included in R.

```
> data( "women" )  # load the data set into the workspace
> women
```

	height	weight
1	58	115
2	59	117
3	60	120
4	61	123
5	62	126
6	63	129
7	64	132
8	65	135
9	66	139
10	67	142
11	68	146
12	69	150
13	70	154
14	71	159
15	72	164

```
> names( women )  # display the variable names

[1] "height" "weight"

> dim( women )  # dimension of the data set (rows and columns)

[1] 15  2

> nrow( women )  # number of rows (observations)

[1] 15

> ncol( women )  # number of columns (variables)

[1] 2

> women[[ "height" ]]  # display the values of variable "height"

[1] 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72

> women$height  # short-cut for the previous command
```

1 Introduction

```
[1] 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72

> women$height[ 3 ] # height of the third observation

[1] 60

> women[ 3, "height" ] # the same

[1] 60

> women[ 3, 1 ] # also the same

[1] 60

> women[ 1:3, 1 ] # height of the first three observations

[1] 58 59 60

> women[ 1:3, ] # all variables of the first three observations

  height weight
1     58    115
2     59    117
3     60    120

> women$cmHeight <- 2.54 * women$height # new variable: height in cm
> women$kgWeight <- women$weight / 2.205 # new variable: weight in kg
> women$bmi <- women$kgWeight / ( women$cmHeight / 100 )^2 # new variable: BMI
> women

  height weight cmHeight kgWeight    bmi
1     58    115   147.32  52.15420 24.03067
2     59    117   149.86  53.06122 23.62685
3     60    120   152.40  54.42177 23.43164
4     61    123   154.94  55.78231 23.23643
5     62    126   157.48  57.14286 23.04152
6     63    129   160.02  58.50340 22.84718
7     64    132   162.56  59.86395 22.65364
8     65    135   165.10  61.22449 22.46110
9     66    139   167.64  63.03855 22.43112
10    67    142   170.18  64.39909 22.23631
11    68    146   172.72  66.21315 22.19520
12    69    150   175.26  68.02721 22.14711
13    70    154   177.80  69.84127 22.09269
14    71    159   180.34  72.10884 22.17198
15    72    164   182.88  74.37642 22.23836
```

1.2.7 Functions

In order to execute a function in R, the function name has to be followed by a pair of parenthesis (round brackets). The documentation of a function (if available) can be obtained by, e.g., typing at the R prompt a question mark followed by the name of the function.

```
> ?log
```

One can read in the documentation of the function `log`, e.g., that this function has a second optional argument `base`, which can be used to specify the base of the logarithm. By default, the base is equal to the Euler number (e , `exp(1)`). A different base can be chosen by adding a second argument, either with or without specifying the name of the argument.

```
> log( 100, base = 10 )
```

```
[1] 2
```

```
> log( 100, 10 )
```

```
[1] 2
```

1.2.8 Simple graphics

Histograms can be created with the command `hist`. The optional argument `breaks` can be used to specify the approximate number of cells:

```
> hist( women$bmi )
```

```
> hist( women$bmi, breaks = 10 )
```

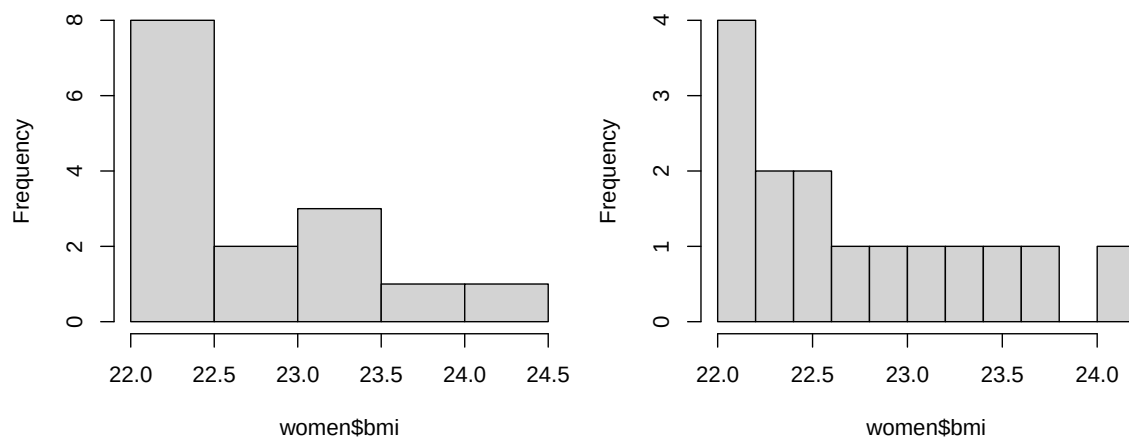


Figure 1.1: Histogram of BMIs

The resulting histogram is shown in figure 1.1.

Scatter plots can be created with the command `plot`:

```
> plot( women$height, women$weight )
```

The resulting scatter plot is shown in figure 1.2.

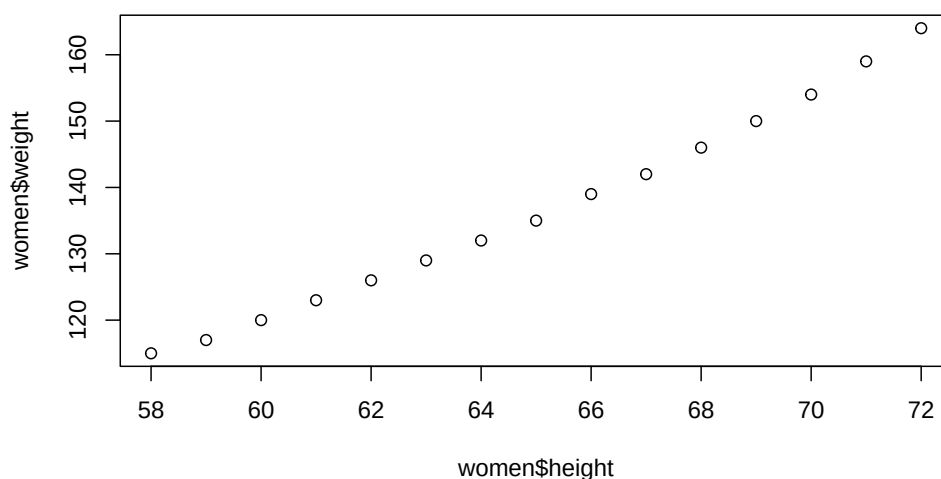


Figure 1.2: Scatter plot of heights and weights

1.2.9 Other useful commands

```
> class( a )

[1] "numeric"

> class( women )

[1] "data.frame"

> class( women$height )

[1] "numeric"

> ls() # list all objects in the workspace

[1] "a"      "b"      "v"      "w"      "women"

> rm(w) # remove an object
> ls()

[1] "a"      "b"      "v"      "women"
```

1.2.10 Extension packages

Currently (June 12, 2013, 2pm GMT), 4611 extension packages for R are available on CRAN (Comprehensive R Archive Network, <http://cran.r-project.org>). When an extension package is installed, it can be loaded with the command `library`. The following command loads the R package *foreign* that includes function for reading data in various formats.

```
> library( "foreign" )
```

Please note that you should cite scientific software packages in your publications if you used them for obtaining your results (as any other scientific works). You can use the command `citation` to find out how an R package should be cited, e.g.:

```
> citation( "frontier" )
```

To cite package 'frontier' in publications use:

```
Tim Coelli and Arne Henningsen (2020). frontier: Stochastic Frontier  
Analysis. R package version 1.1-8.  
https://CRAN.R-Project.org/package=frontier.
```

A BibTeX entry for LaTeX users is

```
@Manual{,  
  title = {frontier: Stochastic Frontier Analysis},  
  author = {Tim Coelli and Arne Henningsen},  
  year = {2020},  
  note = {R package version 1.1-8},  
  url = {https://CRAN.R-Project.org/package=frontier},  
}
```

1.2.11 Reading data into R

R can read and import data from many different file formats. This is described in the official R manual “R Data Import/Export” (<http://cran.r-project.org/doc/manuals/r-release/R-data.pdf>). I usually read my data into R from files in CSV (comma separated values) format. This can be done by the function `read.csv`. The command `read.csv2` can read files in the “European CSV format” (values separated by semicolons, comma as decimal separator). The functions `read.spss` and `read.xport` (both in package *foreign*) can read SPSS data files and SAS “XPORT” files, respectively. While the add-on package *readstata13* can be used to read binary data files from all STATA versions (including versions 13 and 14), function `read.dta` (in package *foreign*) can only read binary data files from STATA versions 5–12. Functions for reading MS-Excel files are available, e.g., in the packages *XLConnect* and *xlsx*.

1.2.12 Linear regression

The command for estimating linear models in R is `lm`. The first argument of the command `lm` specifies the model that should be estimated. This must be a formula object that consists of the name of the dependent variable, followed by a tilde (`~`) and the name of the explanatory variable. Argument `data` can be used to specify the data set:

1 Introduction

```
> olsWeight <- lm( weight ~ height, data = women )
> olsWeight
```

Call:

```
lm(formula = weight ~ height, data = women)
```

Coefficients:

(Intercept)	height
-87.52	3.45

The `summary` method can be used to display summary statistics of the regression:

```
> summary( olsWeight )
```

Call:

```
lm(formula = weight ~ height, data = women)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.7333	-1.1333	-0.3833	0.7417	3.1167

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-87.51667	5.93694	-14.74	1.71e-09 ***
height	3.45000	0.09114	37.85	1.09e-14 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.525 on 13 degrees of freedom

Multiple R-squared: 0.991, Adjusted R-squared: 0.9903

F-statistic: 1433 on 1 and 13 DF, p-value: 1.091e-14

The command `abline` can be used to add a linear (regression) line to a (scatter) plot:

```
> plot( women$height, women$weight )
> abline( olsWeight )
```

The resulting plot is shown in figure 1.3. This figure indicates that the relationship between the height and the corresponding average weights of the women is slightly nonlinear. Therefore, we add the squared height as additional explanatory regressor. When specifying more than one explanatory variable, the names of the explanatory variables must be separated by plus signs (+):

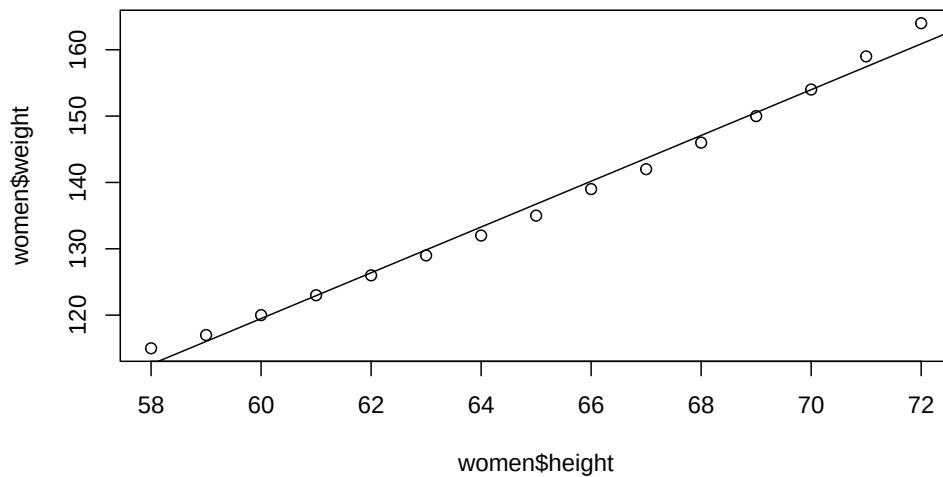


Figure 1.3: Scatter plot of heights and weights with estimated regression line

```
> women$heightSquared <- women$height^2
> olsWeight2 <- lm( weight ~ height + heightSquared, data = women )
> summary( olsWeight2 )
```

Call:

```
lm(formula = weight ~ height + heightSquared, data = women)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.50941	-0.29611	-0.00941	0.28615	0.59706

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	261.87818	25.19677	10.393	2.36e-07 ***
height	-7.34832	0.77769	-9.449	6.58e-07 ***
heightSquared	0.08306	0.00598	13.891	9.32e-09 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3841 on 12 degrees of freedom

Multiple R-squared: 0.9995, Adjusted R-squared: 0.9994

F-statistic: 1.139e+04 on 2 and 12 DF, p-value: < 2.2e-16

One can use the function `I()` to calculate explanatory variables directly in the formula:

```
> olsWeight3 <- lm( weight ~ height + I(height^2), data = women )
> summary( olsWeight3 )
```

1 Introduction

Call:

```
lm(formula = weight ~ height + I(height^2), data = women)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-0.50941	-0.29611	-0.00941	0.28615	0.59706

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	261.87818	25.19677	10.393	2.36e-07 ***
height	-7.34832	0.77769	-9.449	6.58e-07 ***
I(height^2)	0.08306	0.00598	13.891	9.32e-09 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3841 on 12 degrees of freedom

Multiple R-squared: 0.9995, Adjusted R-squared: 0.9994

F-statistic: 1.139e+04 on 2 and 12 DF, p-value: < 2.2e-16

The `coef` method for `lm` objects can be used to extract the vector of the estimated coefficients:

```
> coef( olsWeight2 )
```

	(Intercept)	height	heightSquared
	261.87818358	-7.34831933	0.08306399

When the `coef` method is applied to the object returned by the `summary` method for `lm` objects, the matrix of the estimated coefficients, their standard errors, their t -values, and their P -values is returned:

```
> coef( summary( olsWeight2 ) )
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	261.87818358	25.196770820	10.393323	2.356879e-07
height	-7.34831933	0.777692280	-9.448878	6.584476e-07
heightSquared	0.08306399	0.005979642	13.891131	9.322439e-09

The variance covariance matrix of the estimated coefficients can be obtained by the `vcov` method:

```
> vcov( olsWeight2 )
```

```

              (Intercept)          height heightSquared
(Intercept)  634.8772597 -19.586524729  1.504022e-01
height       -19.5865247   0.604805283 -4.648296e-03
heightSquared  0.1504022  -0.004648296  3.575612e-05

```

The `residuals` method for `lm` objects can be used to obtain the residuals:

```

> residuals( olsWeight2 )

      1      2      3      4      5      6
-0.102941176 -0.473109244 -0.009405301  0.288170653  0.419618617  0.384938591
      7      8      9     10     11     12
 0.184130575 -0.182805430  0.284130575 -0.415061409 -0.280381383 -0.311829347
     13     14     15
-0.509405301  0.126890756  0.597058824

```

The `fitted` method for `lm` objects can be used to obtain the fitted values:

```

> fitted( olsWeight2 )

      1      2      3      4      5      6      7      8
115.1029 117.4731 120.0094 122.7118 125.5804 128.6151 131.8159 135.1828
      9     10     11     12     13     14     15
138.7159 142.4151 146.2804 150.3118 154.5094 158.8731 163.4029

```

We can evaluate the “fit” of the model by plotting the fitted values against the observed values of the dependent variable and adding a 45-degree line:

```

> plot( women$weight, fitted( olsWeight2 ) )
> abline(0,1)

```

The resulting scatter plot is shown in figure 1.4.

The `plot` method for `lm` objects can be used to generate diagnostic plots

```

> plot( olsWeight2 )

```

The resulting diagnostic plots are shown in figure 1.5.

1.3 R packages

The following R packages are particularly relevant for empirical production, efficiency, and productivity analysis using R (in alphabetical order):

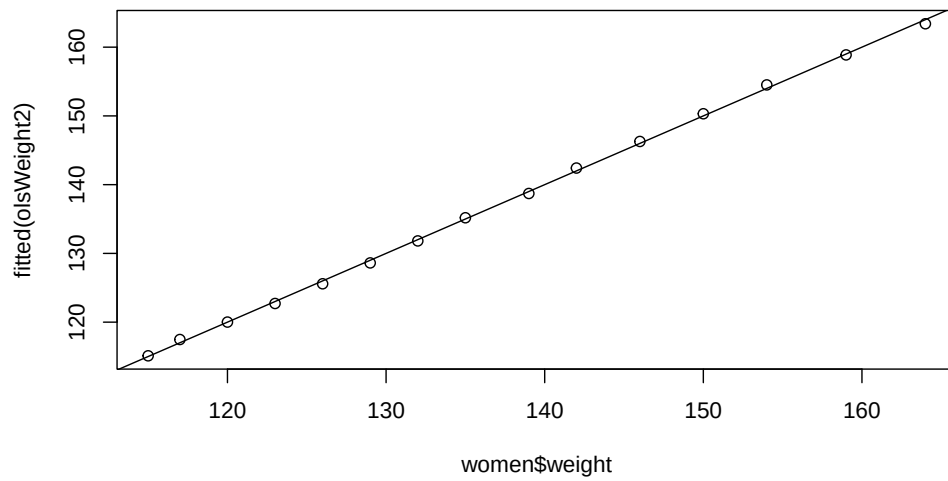


Figure 1.4: Observed and fitted values of the dependent variable

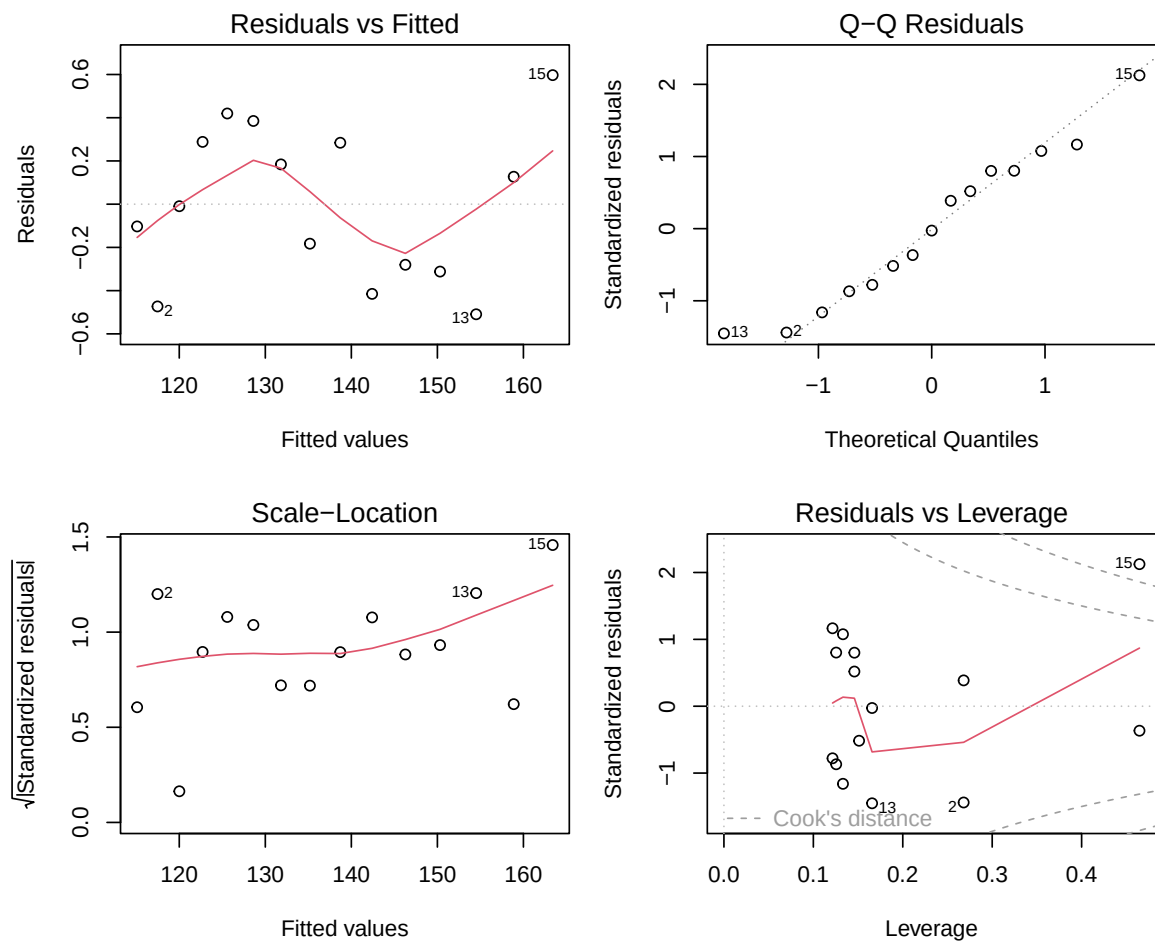


Figure 1.5: Diagnostic plots

Benchmarking

- various DEA models
- rudimentary SFA models
- extensively described in the book of Bogetoft and Otto (2011)
- available at <http://cran.r-project.org/package=Benchmarking>

FEAR

- various DEA models
- bootstrapping
- closed source
- non-academic use requires payment of a license fee
- available at: <http://www.clemson.edu/economics/faculty/wilson/Software/FEAR/fear.html>

frontiles

- alpha-quantile and order-m efficiency
- available at <http://cran.r-project.org/package=frontiles>

frontier

- cross-sectional and panel data stochastic frontier models as suggested by Battese and Coelli (1992)
- stochastic frontier models incorporating a model for inefficiency effects as suggested by Battese and Coelli (1995)
- based on FRONTIER 4.1 (Coelli, 1996)
- available at <http://cran.r-project.org/package=frontier>

nonparaeff

- DEA
- available at <http://cran.r-project.org/package=nonparaeff>

np

- nonparametric kernel regression
- available at <http://cran.r-project.org/package=np>

plm

- estimation and testing of various specifications of linear panel data models
- available at <http://cran.r-project.org/package=plm>

rDEA

- robust DEA
- available at <http://cran.r-project.org/package=rDEA>

semsfa

- semiparametric SFA
- available at <http://cran.r-project.org/package=semsfa>

sfa

- SFA
- available at <http://cran.r-project.org/package=sfa>

spfrontier

- spatial SFA
- available at <http://cran.r-project.org/package=spfrontier>

ssfa

- spatial SFA
- available at <http://cran.r-project.org/package=ssfa>

TFDEA

- technology forecasting using DEA
- available at <http://cran.r-project.org/package=TFDEA>

1.4 Data sets

In my courses in the field of econometric production analysis, I usually use two data sets: one cross-sectional data set of French apple producers and a panel data set of rice producers on the Philippines.

1.4.1 French apple producers

1.4.1.1 Description of the data set

In this course, we will predominantly use a cross-sectional production data set of 140 French apple producers from the year 1986. These data are extracted from a panel data set that has been used in an article published by Ivaldi et al. (1996) in the *Journal of Applied Econometrics*.

The full panel data set is available in the journal's data archive: <http://www.econ.queensu.ca/jae/1996-v11.6/ivaldi-ladoux-ossard-simioni/>.¹

The cross-sectional data set that we will predominantly use in the course is available in the R package *micEcon*. It has the name `appleProdFr86` and can be loaded by the command:

```
> data( "appleProdFr86", package = "micEcon" )
```

The names of the variables in the data set can be obtained by the command `names`:

```
> names( appleProdFr86 )
```

```
[1] "vCap"      "vLab"      "vMat"      "qApples"   "qOtherOut" "qOut"
[7] "pCap"      "pLab"      "pMat"      "pOut"      "adv"
```

The data set includes following variables:²

vCap costs of capital (including land)

vLab costs of labor (including remuneration of unpaid family labor)

vMat costs of intermediate materials (e.g. seedlings, fertilizer, pesticides, fuel)

qOut quantity index of all outputs (apples and other outputs)

pCap price index of capital goods

pLab price index of labor

pMat price index of materials

pOut price index of the aggregate output*

adv use of advisory service*

Please note that variables indicated by * are not in the original data set but are artificially generated in order to be able to conduct some further analyses with this data set. Variable names starting with **v** indicate volumes (values), variable names starting with **q** indicate quantities, and variable names starting with **p** indicate prices.

1.4.1.2 Abbreviating name of data set

In order to avoid too much typing, give the data set a much shorter name (`dat`) by creating a copy of the data set and removing the original data set:

```
> dat <- appleProdFr86
```

```
> rm( appleProdFr86 )
```

¹In order to focus on the microeconomic analysis rather than on econometric issues in panel data analysis, we only use a single year from this panel data set.

²This information is also available in the documentation of this data set, which can be obtained by the command: `help( "appleProdFr86", package = "micEcon" )`.

1.4.1.3 Calculation of input quantities

Our data set does not contain input quantities but prices and costs (volumes) of the inputs. As we will need to know input quantities for many of our analyses, we calculate input quantity indices based on following identity:

$$v_i = x_i \cdot w_i, \quad (1.1)$$

where w_i is the price, x_i is the quantity and v_i is the volume of the i th input. In R, we can calculate the input quantities with the following commands:

```
> dat$qCap <- dat$vCap / dat$pCap
> dat$qLab <- dat$vLab / dat$pLab
> dat$qMat <- dat$vMat / dat$pMat
```

1.4.1.4 Calculation of total costs, variable costs, and cost shares

Total costs are defined as:

$$c = \sum_{i=1}^N w_i x_i, \quad (1.2)$$

where N denotes the number of inputs. We can calculate the apple producers' total costs by following command:

```
> dat$cost <- with( dat, vCap + vLab + vMat )
```

Alternatively, we can calculate the costs by summing up the products of the quantities and the corresponding prices over all inputs:

```
> all.equal( dat$cost, with( dat, pCap * qCap + pLab * qLab + pMat * qMat ) )
```

```
[1] TRUE
```

Variable costs are defined as:

$$c^v = \sum_{i \in N^1} w_i x_i, \quad (1.3)$$

where N^1 is a vector of the indices of the variable inputs. If capital is a quasi-fixed input and labor and materials are variable inputs, the apple producers' variable costs can be calculated by following command:

```
> dat$vCost <- with( dat, vLab + vMat )
```

The cost shares of the individual inputs in total costs can be calculated as:

```
> dat$sCap <- dat$vCap / dat$cost
> dat$sLab <- dat$vLab / dat$cost
> dat$sMat <- dat$vMat / dat$cost
```

1.4.1.5 Calculation of profit and gross margin

Profit is defined as:

$$\pi = p y - \sum_{i=1}^N w_i x_i = p y - c, \quad (1.4)$$

where all variables are defined as above. We can calculate the apple producers' profits by:

```
> dat$profit <- with( dat, pOut * qOut - cost )
```

Alternatively, we can calculate the profit by subtracting the products of the quantities and the corresponding prices of all inputs from the revenues:

```
> all.equal( dat$cost, with( dat, pCap * qCap + pLab * qLab + pMat * qMat ) )
```

```
[1] TRUE
```

The gross margin ("variable profit") is defined as:

$$\pi^v = p y - \sum_{i \in N^1} w_i x_i = p y - c^v, \quad (1.5)$$

where all variables are defined as above. If capital is a quasi-fixed input and labor and materials are variable inputs, the apple producers' gross margins can be calculated by following command:

```
> dat$vProfit <- with( dat, pOut * qOut - vLab - vMat )
```

1.4.2 Rice producers on the Philippines

1.4.2.1 Description of the data set

In the last part of this course, we will use a balanced panel data set of annual data collected from 43 smallholder rice producers in the Tarlac region of the Philippines between 1990 and 1997. This data set has the name `riceProdPhil` and is available in the R package *frontier*. Detailed information about these data is available in the documentation of this data set. We can load this data set with following command:

```
> data( "riceProdPhil", package = "frontier" )
```

The names of the variables in the data set can be obtained by the command `names`:

```
> names( riceProdPhil )
```

```
[1] "YEARDUM" "FMERCODE" "PROD"      "AREA"      "LABOR"     "NPK"
[7] "OTHER"   "PRICE"     "AREAP"     "LABORP"    "NPKP"      "OTHERP"
[13] "AGE"     "EDYRS"     "HHSIZE"    "NADULT"    "BANRAT"
```

The following variables are of particular importance for our analysis:

PROD output (tonnes of freshly threshed rice)
AREA area planted (hectares).
LABOR labor used (man-days of family and hired labor)
NPK fertilizer used (kg of active ingredients)
YEARDUM time period (1 = 1990, ..., 8 = 1997)

In our analysis of the production technology of the rice producers we will use variable **PROD** as output quantity and variables **AREA**, **LABOR**, and **NPK** as input quantities.

1.4.2.2 Mean-scaling quantities

In some model specifications, it is an advantage to use mean-scaled quantities. Therefore, we create new variables with mean-scaled input and output quantities:

```
> riceProdPhil$area <- riceProdPhil$AREA / mean( riceProdPhil$AREA )
> riceProdPhil$labor <- riceProdPhil$LABOR / mean( riceProdPhil$LABOR )
> riceProdPhil$npk <- riceProdPhil$NPK / mean( riceProdPhil$NPK )
> riceProdPhil$prod <- riceProdPhil$PROD / mean( riceProdPhil$PROD )
```

As expected, the sample means of the mean-scaled variables are all one so that their logarithms are all zero (except for negligible very small rounding errors):

```
> colMeans( riceProdPhil[ , c( "prod", "area", "labor", "npk" ) ] )

prod  area labor  npk
    1    1    1    1

> log( colMeans( riceProdPhil[ , c( "prod", "area", "labor", "npk" ) ] ) )

      prod      area      labor      npk
0.000000e+00 -1.110223e-16 0.000000e+00 0.000000e+00
```

1.4.2.3 Logarithmic mean-scaled quantities

As we use logarithmic input and output quantities in the Cobb-Douglas and Translog specifications, we can reduce our typing work by creating variables with logarithmic (mean-scaled) input and output quantities:

```
> riceProdPhil$lArea <- log( riceProdPhil$area )
> riceProdPhil$lLabor <- log( riceProdPhil$labor )
> riceProdPhil$lNpk <- log( riceProdPhil$npk )
> riceProdPhil$lProd <- log( riceProdPhil$prod )
```

Please note that the (arithmetic) mean values of the logarithmic mean-scaled variables are not equal to zero:

```
> colMeans( riceProdPhil[ , c( "lProd", "lArea", "lLabor", "lNpk" ) ] )

      lProd      lArea      lLabor      lNpk
-0.3263075 -0.2718549 -0.2772354 -0.4078492
```

1.4.2.4 Mean-adjusting the time trend

In some model specifications, it is an advantage to have a time trend variable that is zero at the sample mean. If we subtract the sample mean from our time trend variable, the sample mean of the adjusted time trend is zero:

```
> riceProdPhil$mYear <- riceProdPhil$YEARDUM - mean( riceProdPhil$YEARDUM )
> mean( riceProdPhil$mYear )

[1] 0
```

1.4.2.5 Total costs and cost shares

The following code calculates total costs and the cost shares (ignoring “other inputs”):

```
> riceProdPhil$cost <- with( riceProdPhil,
+   AREA * AREAP + LABOR * LABORP + NPK * NPKP )
> riceProdPhil$sArea <- with( riceProdPhil, AREA * AREAP / cost )
> riceProdPhil$sLabor <- with( riceProdPhil, LABOR * LABORP / cost )
> riceProdPhil$sNpk <- with( riceProdPhil, NPK * NPKP / cost )
```

Now we check whether these cost shares sum up to one:

```
> all.equal( with( riceProdPhil, sArea + sLabor + sNpk ),
+   rep( 1, nrow( riceProdPhil ) ) )

[1] TRUE
```

1.4.2.6 Specifying panel structure

This data set does not include any information about its panel structure. Hence, R would ignore the panel structure and treat this data set as cross-sectional data collected from 352 different producers. The command `pdata.frame` of the *plm* package (Croissant and Millo, 2008) can be used to create data sets that include the information on its panel structure. The following commands creates a new data set of the rice producers from the Philippines that includes information on the panel structure, i.e. variable `FMERCODE` indicates the individual (farmer), and variable `YEARDUM` indicated the time period (year):³

```
> library( "plm" )
> pdat <- pdata.frame( riceProdPhil, c( "FMERCODE", "YEARDUM" ) )
```

³Please note that the specification of variable `YEARDUM` as the time dimension in the panel data set `pdat` converts this variable to a categorical variable. If a numeric time variable is needed, it can be created, e.g., by the command `pdat$year <- as.numeric( pdat$YEARDUM )`.

1.5 Mathematical and statistical methods

1.5.1 Exponentiation

Exponentiation is a calculation with two numbers, e.g., b^a , where one number (in this example b) is the base and the other number (in this example a) is the exponent. If the base is equal to the Euler number $e \approx 2.71828$, the exponentiation e^a is called exponential function. In R, exponentiation can be done with the symbol “ $\wedge$ ”, while values of the exponential function can be obtained by function `exp()`. The most important rules for calculating with exponentiation (including exponential functions) are:

- $b^0 = 1$, e.g., $e^0 = 1$
- $b^1 = b$, e.g., $e^1 = e$
- $b^{-x} = 1/b^x$, e.g., $e^{-x} = 1/e^x$
- $b^{x+y} = b^x \cdot b^y$, e.g., $e^{x+y} = e^x \cdot e^y$
- $b^{x-y} = b^x / b^y$, e.g., $e^{x-y} = e^x / e^y$
- $b^{x \cdot y} = (b^x)^y$, e.g., $e^{x \cdot y} = (e^x)^y$
- $b^{1/x} = \sqrt[x]{b}$
- $(x \cdot y)^a = x^a \cdot y^a$
- $(x/y)^a = x^a / y^a$

1.5.2 Logarithms

$\log_b(x)$ indicates the logarithm of x to the base b . If $y = \log_b(x)$, then $b^y = x$. In economics, we often use the natural logarithm $\ln(x) \equiv \log_e(x)$, i.e., the logarithm of x to the base of the Euler number $e \approx 2.71828$. Some economists write $\log(x)$ even if they mean $\ln(x)$. In R, function `log()` calculates logarithms with the base being the Euler number by default, i.e., `log(x)` returns the natural logarithm of x , whereas, e.g., `log(x, base = 10)` returns the logarithm of x to the base 10. The most important rules for calculating with logarithms (including natural logarithms) are:

- $\log_b(1) = 0$, e.g., $\ln(1) = 0$
- $\log_b(b) = 1$, e.g., $\ln(e) = 1$
- $\log_b(b^n) = n$, e.g., $\ln(e^n) = n$
- $\lim_{x \rightarrow 0+} \log_b(x) = -\infty$, e.g., $\lim_{x \rightarrow 0+} \ln(x) = -\infty$
- $\log_b(x)$ and also $\ln(x)$ are undefined for $x \leq 0$
- $\log_b(1/x) = -\log_b(x)$, e.g., $\ln(1/x) = -\ln(x)$

- $\log_b(x \cdot y) = \log_b(x) + \log_b(y)$, e.g., $\ln(x \cdot y) = \ln(x) + \ln(y)$
- $\log_b(x/y) = \log_b(x) - \log_b(y)$, e.g., $\ln(x/y) = \ln(x) - \ln(y)$
- $\log_b(x^y) = y \log_b(x)$, e.g., $\ln(x^y) = y \ln(x)$
- $\log_b(x) = \ln(x)/\ln(b)$
- $\log_b(x + y)$ cannot be reasonably transformed

1.5.3 Partial derivatives

The most important partial derivatives in microeconomic and econometric production analysis are:

- $\frac{\partial a}{\partial x} = 0$
- $\frac{\partial x^a}{\partial x} = a \cdot x^{a-1}$
- $\frac{\partial e^x}{\partial x} = e^x$
- $\frac{\partial \ln(x)}{\partial x} = \frac{1}{x}$
- $\frac{\partial (a \cdot f(x))}{\partial x} = a \frac{\partial f(x)}{\partial x}$
- $\frac{\partial (f(x) + g(x))}{\partial x} = \frac{\partial f(x)}{\partial x} + \frac{\partial g(x)}{\partial x}$
- $\frac{\partial (f(x) - g(x))}{\partial x} = \frac{\partial f(x)}{\partial x} - \frac{\partial g(x)}{\partial x}$
- $\frac{\partial (f(x) \cdot g(x))}{\partial x} = \frac{\partial f(x)}{\partial x} \cdot g(x) + \frac{\partial g(x)}{\partial x} \cdot f(x)$
- $\frac{\partial (f(x)/g(x))}{\partial x} = \left(\frac{\partial f(x)}{\partial x} \cdot g(x) - \frac{\partial g(x)}{\partial x} \cdot f(x) \right) / (g(x))^2$
- $\frac{\partial f(g(x))}{\partial x} = \frac{\partial f(g(x))}{\partial g(x)} \cdot \frac{\partial g(x)}{\partial x}$

1.5.4 Aggregating quantities

Sometimes, it is desirable to aggregate quantities of different goods to an aggregate quantity. This can be done by a quantity index, e.g. the Laspeyres or Paasche quantity index

$$X_j^L = \frac{\sum_i x_{ij} \cdot p_{i0}}{\sum_i x_{i0} \cdot p_{i0}} \quad X_j^P = \frac{\sum_i x_{ij} \cdot p_{ij}}{\sum_i x_{i0} \cdot p_{ij}}, \quad (1.6)$$

where subscript i indicates the good, subscript j indicates the observation, x_{i0} is the “base” quantity, and p_{i0} is the “base” price of the i th good, e.g. the sample means.

The Paasche and Laspeyres quantity indices of all three inputs in the data set of French apple producers can be calculated by:

```
> dat$XP <- with( dat,
+   ( vCap + vLab + vMat ) /
+   ( mean( qCap ) * pCap + mean( qLab ) * pLab + mean( qMat ) * pMat ) )
> dat$XL <- with( dat,
+   ( qCap * mean( pCap ) + qLab * mean( pLab ) + qMat * mean( pMat ) ) /
+   ( mean( qCap ) * mean( pCap ) + mean( qLab ) * mean( pLab ) +
+     mean( qMat ) * mean( pMat ) ) )
```

In many cases, the choice of the formula for calculating quantity indices does not have a major influence on the result. We demonstrate this with two scatter plots, where we set argument `log` of the second `plot` command to the character string "xy" so that both axes are measured in logarithmic terms and the dots (firms) are more equally spread:

```
> plot( dat$XP, dat$XL )
> plot( dat$XP, dat$XL, log = "xy" )
```

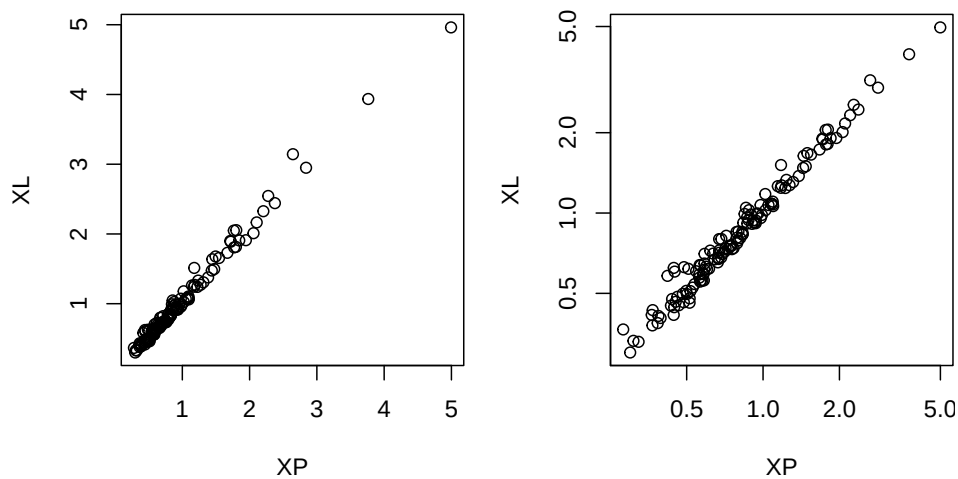


Figure 1.6: Comparison of Paasche and Laspeyres quantity indices

The resulting scatter plots are shown in figure 1.6.

As a compromise, one can use the Fisher quantity index, which is the geometric mean of the Paasche quantity index and the Laspeyres quantity index:

```
> dat$X <- sqrt( dat$XP * dat$XL )
```

We can also use function `quantityIndex` from the *micEconIndex* package to calculate the quantity index:

```
> library( "micEconIndex" )
> dat$XP2 <- quantityIndex( c( "pCap", "pLab", "pMat" ),
```

```

+      c( "qCap", "qLab", "qMat" ), data = dat, method = "Paasche" )
> all.equal( dat$XP, dat$XP2, check.attributes = FALSE )

[1] TRUE

> dat$XL2 <- quantityIndex( c( "pCap", "pLab", "pMat" ),
+      c( "qCap", "qLab", "qMat" ), data = dat, method = "Laspeyres" )
> all.equal( dat$XL, dat$XL2, check.attributes = FALSE )

[1] TRUE

> dat$X2 <- quantityIndex( c( "pCap", "pLab", "pMat" ),
+      c( "qCap", "qLab", "qMat" ), data = dat, method = "Fisher" )
> all.equal( dat$X, dat$X2, check.attributes = FALSE )

[1] TRUE

```

1.5.5 Concave and convex functions

A function $f(x) : \mathbb{R}^N \rightarrow \mathbb{R}$ is $\begin{smallmatrix} \text{concave} \\ \text{convex} \end{smallmatrix}$ over a convex set D if and only if

$$\theta f(x_u) + (1 - \theta)f(x_v) \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} f(\theta x_u + (1 - \theta)x_v) \quad (1.7)$$

for all combinations of $x_u, x_v \in D$ and for all $0 \leq \theta \leq 1$, while it is *strictly* $\begin{smallmatrix} \text{concave} \\ \text{convex} \end{smallmatrix}$ over a convex set D if and only if

$$\theta f(x_u) + (1 - \theta)f(x_v) \begin{smallmatrix} < \\ > \end{smallmatrix} f(\theta x_u + (1 - \theta)x_v) \quad (1.8)$$

for all combinations of $x_u, x_v \in D$ and for all $0 < \theta < 1$ ([Chiang, 1984](#), p. 342).

A *continuous and twice continuously differentiable* function $f(x) : \mathbb{R}^N \rightarrow \mathbb{R}$ is $\begin{smallmatrix} \text{concave} \\ \text{convex} \end{smallmatrix}$ over a convex set D if and only if its Hessian matrix is $\begin{smallmatrix} \text{negative} \\ \text{positive} \end{smallmatrix}$ semidefinite at all $x \in D$, while it is *strictly* $\begin{smallmatrix} \text{concave} \\ \text{convex} \end{smallmatrix}$ over a convex set D if (but not only if) its Hessian matrix is $\begin{smallmatrix} \text{negative} \\ \text{positive} \end{smallmatrix}$ definite at all $x \in D$ ([Chiang, 1984](#), p. 347).

A symmetric quadratic $N \times N$ matrix H is negative semidefinite if and only if all its i th-order principal minors (not only its leading principal minors) are non-positive for i being odd and non-negative for i being even for all $i \in \{1, \dots, N\}$, while it is positive semidefinite if all its principal minors (not only its leading principal minors) are non-negative ([Gantmacher, 1959](#), p. 307–308). An i th-order principal minor of a symmetric $N \times N$ matrix H is the determinant of an $k \times k$ submatrix of H , whereas $N - k$ rows and the corresponding $N - k$ columns have been deleted. An $N \times N$ matrix has $\binom{N}{k}$ i th-order principal minors and in total $\sum_{k=1}^N \binom{N}{k}$ principal minors.⁴

⁴The binomial coefficient $\binom{N}{k}$ (“ N choose k ”) can be calculated in R with the command `choose( N, k )`.

A quadratic $N \times N$ matrix H is negative definite if and only if its first leading principal minor is strictly negative and the following leading principle minors alternate in sign, i.e. $|B_1| < 0$, $|B_2| > 0$, $|B_3| < 0$, $\dots$, $(-1)^N |B_N| > 0$, while it is positive definite if and only if all its N leading principal minors are positive, i.e. $|B_i| > 0 \forall i \in \{1, \dots, N\}$, where $|B_i|$ is the i th leading principal minor of matrix H (Chiang, 1984, p. 324). An i th-order *leading* principal minor of a symmetric $N \times N$ matrix H is the determinant of an $k \times k$ submatrix of H , whereas last $N - k$ rows and the last $N - k$ columns have been deleted. An $N \times N$ has N leading principal minors.

A quadratic $N \times N$ matrix H is $\begin{smallmatrix} \text{negative} \\ \text{positive} \end{smallmatrix}$ semidefinite if and only if all N eigenvalues of H are $\begin{smallmatrix} \text{non-positive} \\ \text{non-negative} \end{smallmatrix}$ and at least one of these eigenvalues is zero (Chiang, 1984, p. 330).

A quadratic $N \times N$ matrix H is $\begin{smallmatrix} \text{negative} \\ \text{positive} \end{smallmatrix}$ definite if and only if all N eigenvalues of H are strictly $\begin{smallmatrix} \text{negative} \\ \text{positive} \end{smallmatrix}$ (Chiang, 1984, p. 330).

1.5.6 Quasiconcave and quasiconvex functions

A function $f(x) : \mathbb{R}^N \rightarrow \mathbb{R}$ is $\begin{smallmatrix} \text{quasiconcave} \\ \text{quasiconvex} \end{smallmatrix}$ over a convex set D if and only if

$$f(\theta x_u + (1 - \theta)x_v) \begin{smallmatrix} \geq \\ \leq \end{smallmatrix} \min(f(x_u), f(x_v)) \quad (1.9)$$

for all combinations of $x_u, x_v \in D$ and for all $0 \leq \theta \leq 1$, while it is *strictly* $\begin{smallmatrix} \text{quasiconcave} \\ \text{quasiconvex} \end{smallmatrix}$ over a convex set D if and only if

$$f(\theta x_u + (1 - \theta)x_v) \begin{smallmatrix} > \\ < \end{smallmatrix} \min(f(x_u), f(x_v)) \quad (1.10)$$

for all combinations of $x_u, x_v \in D$ and for all $0 < \theta < 1$ (Chiang, 1984, p. 389; Chambers, 1988, p. 311).

An alternative definition of quasiconcavity and quasiconvexity is that a function $f(x) : \mathbb{R}^N \rightarrow \mathbb{R}$ is $\begin{smallmatrix} \text{quasiconcave} \\ \text{quasiconvex} \end{smallmatrix}$ over a convex set D if and only if

$$\begin{aligned} S^u(k) &= \{x | f(x) \geq k\} \\ S^l(k) &= \{x | f(x) \leq k\} \end{aligned} \text{ is a convex set} \quad (1.11)$$

for any constant k (Chiang, 1984, p. 391; Chambers, 1988, p. 311). The sets $S^u(k)$ and $S^l(k)$ are called upper contour set and lower contour set, respectively. There is no corresponding definition of strict quasiconcavity or strict quasiconvexity (Chiang, 1984, p. 391).

If $f(x) : \mathbb{R}^N \rightarrow \mathbb{R}$ is a *continuous and twice continuously differentiable* function, a $\begin{smallmatrix} \text{necessary} \\ \text{sufficient} \end{smallmatrix}$ condition for quasiconcavity is that $|B_1| \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} 0$, $|B_2| \begin{smallmatrix} \geq \\ \leq \end{smallmatrix} 0$, $|B_3| \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} 0$, $\dots$, $(-1)^N |B_N| \begin{smallmatrix} \geq \\ \leq \end{smallmatrix} 0$, while a $\begin{smallmatrix} \text{necessary} \\ \text{sufficient} \end{smallmatrix}$ condition for quasiconvexity is that $|B_1| \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} 0$, $|B_2| \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} 0$, $|B_3| \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} 0$, $\dots$, $|B_N| \begin{smallmatrix} \leq \\ \geq \end{smallmatrix} 0$, where $|B_i|$ is the i th

leading principal minor of the bordered Hessian matrix

$$B = \begin{bmatrix} 0 & f_1 & f_2 & \dots & f_N \\ f_1 & f_{11} & f_{12} & \dots & f_{1N} \\ f_2 & f_{12} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_N & f_{1N} & f_{2N} & \dots & f_{NN} \end{bmatrix}, \quad (1.12)$$

f_i denotes the partial derivative of $f(x)$ with respect to x_i , f_{ij} denotes the second partial derivative of $f(x)$ with respect to x_i and x_j , $|B_1|$ is the determinant of the upper left 2×2 sub-matrix of B , $|B_2|$ is the determinant of the upper left 3×3 sub-matrix of B , ..., and $|B_N|$ is the determinant of B (Chiang, 1984, p. 393f; Chambers, 1988, p. 312). While the first inequality of the necessary condition for quasiconcavity and quasiconvexity, i.e. $|B_1| \leq 0$, is always fulfilled given that $|B_1| = -f_1^2$, the first inequality of the sufficient condition for quasiconcavity and quasiconvexity, i.e. $|B_1| < 0$, is only fulfilled if $f_1 \neq 0$ (Chiang, 1984, p. 394f).

1.5.7 Delta method

If we have estimated a parameter vector β and its variance covariance matrix $Var(\beta)$ and we calculate a vector of measures (e.g. elasticities) based on the estimated parameters by $z = g(\beta)$, we can calculate the approximate variance covariance matrix of z by:

$$Var(z) \approx \left(\frac{\partial g(\beta)}{\partial \beta} \right)^\top Var(\beta) \frac{\partial g(\beta)}{\partial \beta}, \quad (1.13)$$

where $\partial g(\beta)/\partial \beta$ is the Jacobian matrix of $z = g(\beta)$ with respect to β and the superscript $\top$ is the transpose operator.

2 Primal Approach: Production Function

2.1 Theory

2.1.1 Production function

The production function

$$y = f(x) \quad (2.1)$$

indicates the maximum quantity of a single output (y) that can be obtained with a vector of given input quantities (x). It is usually assumed that production functions fulfill some properties (see [Chambers, 1988](#), p. 9).

2.1.2 Average products

Very simple measures to compare the (partial) productivities of different firms are the inputs' average products. The average product of the i th input is defined as:

$$AP_i = \frac{f(x)}{x_i} = \frac{y}{x_i} \quad (2.2)$$

The more output one firm produces per unit of input, the more productive is this firm and the higher is the corresponding average product. If two firms use identical input quantities, the firm with the larger output quantity is more productive (has a higher average product). And if two firms produce the same output quantity, the firm with the smaller input quantity is more productive (has a higher average product). However, if these two firms use different input combinations, one firm could be more productive regarding the average product of one input, while the other firm could be more productive regarding the average product of another input.

2.1.3 Total factor productivity

As average products measure just partial productivities, it is often desirable to calculate total factor productivities (TFP):

$$TFP = \frac{y}{X}, \quad (2.3)$$

where X is a quantity index of all inputs (see section [1.5.4](#)).

2.1.4 Marginal products

The marginal productivities of the inputs can be measured by their marginal products. The marginal product of the i th input is defined as:

$$MP_i = \frac{\partial f(x)}{\partial x_i} \quad (2.4)$$

2.1.5 Output elasticities

The marginal productivities of the inputs can also be measured by their output elasticities. The output elasticity of the i th input is defined as:

$$\varepsilon_i = \frac{\partial f(x)}{\partial x_i} \frac{x_i}{f(x)} = \frac{MP}{AP} \quad (2.5)$$

In contrast to the marginal products, the changes of the input and output quantities are measured in relative terms so that output elasticities are independent of the units of measurement. Output elasticities are sometimes also called partial output elasticities or partial production elasticities.

2.1.6 Elasticity of scale and most productive scale size

The returns of scale of the technology can be measured by the elasticity of scale:

$$\varepsilon = \sum_i \varepsilon_i \quad (2.6)$$

If the technology has increasing returns to scale ($\varepsilon > 1$), total factor productivity increases when all input quantities are proportionally increased, because the relative increase of the output quantity y is larger than the relative increase of the aggregate input quantity X in equation (2.3). If the technology has decreasing returns to scale ($\varepsilon < 1$), total factor productivity decreases when all input quantities are proportionally increased, because the relative increase of the output quantity y is less than the relative increase of the aggregate input quantity X . If the technology has constant returns to scale ($\varepsilon = 1$), total factor productivity remains constant when all input quantities change proportionally, because the relative change of the output quantity y is equal to the relative change of the aggregate input quantity X .

If the elasticity of scale (monotonically) decreases with firm size, the firm has the most productive scale size (MPSS) at the point, where the elasticity of scale is one. In order to prove this, we let x^0 indicate the vector of a firm's initial input quantities and $s > 0$ indicate a scalar factor by which the firm scales its initial input quantities up or down so that it uses input quantities $x = s \cdot x^0$. When the firm uses the input quantities $x = s \cdot x^0$, its total factor productivity is $TFP = f(x)/X(x)$, where $f(\cdot)$ is the production function and $X(\cdot)$ indicates a linearly homogeneous function that aggregates the individual input quantities to a quantity index. Linear homogeneity of $X(\cdot)$ implies that, e.g., $X(s \cdot x^0) = s X(x^0)$ and $\partial X(s \cdot x^0)/\partial s = X(x^0)$. In the

following, we want to find the value of the scale factor s that gives the input quantities $x = s \cdot x^0$ that maximize total factor productivity. A necessary condition for a maximum of the total factor productivity is that its partial derivative with respect to s is zero:

$$\frac{\partial TFP}{\partial s} = 0 \quad (2.7)$$

$$\frac{\partial \left(\frac{f(x)}{X(x)} \right)}{\partial s} = 0 \quad (2.8)$$

$$\frac{\frac{\partial f(x)}{\partial s} X(x) - f(x) \frac{\partial X(x)}{\partial s}}{(X(x))^2} = 0 \quad (2.9)$$

$$\frac{\partial f(x)}{\partial s} X(x) - f(x) \frac{\partial X(s \cdot x^0)}{\partial s} = 0 \quad (2.10)$$

$$\left(\sum_{i=1}^N \frac{\partial f(x)}{\partial x_i} \frac{\partial x_i}{\partial s} \right) X(s \cdot x^0) - f(x) X(x^0) = 0 \quad (2.11)$$

$$\left(\sum_{i=1}^N \frac{\partial f(x)}{\partial x_i} x_i^0 \right) s X(x^0) = f(x) X(x^0) \quad (2.12)$$

$$\sum_{i=1}^N \frac{\partial f(x)}{\partial x_i} \frac{s \cdot x_i^0}{f(x)} = 1 \quad (2.13)$$

$$\sum_{i=1}^N \frac{\partial f(x)}{\partial x_i} \frac{x}{f(x)} = 1 \quad (2.14)$$

$$\sum_{i=1}^N \epsilon_i = 1 \quad (2.15)$$

$$\epsilon = 1 \quad (2.16)$$

Hence, a necessary condition for a maximum of the total factor productivity at a point $x = s \cdot x^0$ is that the elasticity of scale is one ($\epsilon = 1$). The sufficient condition for a maximum of the total factor productivity is fulfilled if the second derivative of the total factor productivity with respect to the scale factor is negative at the point $x = s \cdot x^0$ with $\epsilon = 1$. This is the case if the elasticity of scale decreases with the scale factor ($\partial \epsilon / \partial s < 0$), i.e., firms that are smaller than the point where $\epsilon = 1$ have increasing returns to scale, while firms that are larger than the point where $\epsilon = 1$ have decreasing returns to scale.

2.1.7 Marginal rates of technical substitution

The marginal rate of technical substitution between input i and input j is (Chambers, 1988, p. 29):

$$MRTS_{i,j} = \frac{\partial x_i}{\partial x_j} = - \frac{\frac{\partial y}{\partial x_j}}{\frac{\partial y}{\partial x_i}} = - \frac{MP_j}{MP_i} \quad (2.17)$$

2.1.8 Relative marginal rates of technical substitution

The relative marginal rate of technical substitution between input i and input j is:

$$RMRTS_{i,j} = \frac{\frac{\partial x_i}{\partial x_j} \frac{x_j}{x_i}}{\frac{\partial y}{\partial x_i} \frac{x_i}{y}} = -\frac{\varepsilon_j}{\varepsilon_i} \quad (2.18)$$

2.1.9 Elasticities of substitution

The elasticity of substitution measures the substitutability between two inputs. It is defined as:

$$\sigma_{ij} = \frac{d\left(\frac{x_i}{x_j}\right)}{d\left(\frac{MP_j}{MP_i}\right)} \frac{\frac{MP_j}{MP_i}}{\frac{x_i}{x_j}} = \frac{d\left(\frac{x_i}{x_j}\right)}{d(-MRTS_{ij})} \frac{-MRTS_{ij}}{\frac{x_i}{x_j}} = \frac{d\left(\frac{x_i}{x_j}\right)}{d MRTS_{ij}} \frac{MRTS_{ij}}{\frac{x_i}{x_j}} \quad (2.19)$$

Thus, if input i is substituted for input j so that the input ratio x_i/x_j increases by $\sigma_{ij}\%$, the marginal rate of technical substitution between input i and input j will increase by 1%.

2.1.9.1 Direct elasticities of substitution

The *direct* elasticity of substitution measures the substitutability between two inputs when the quantities of all other inputs are held constant. Hence, the *direct* elasticities of substitution can be considered as short-run elasticities of substitution (Chambers, 1988, p. 33). It can be calculated by:

$$\sigma_{ij}^D = \frac{-f_i f_j (x_i f_i + x_j f_j)}{x_i x_j (f_{ii} f_j^2 - 2f_{ij} f_i f_j + f_{jj} f_i^2)}, \quad (2.20)$$

where $f_i \equiv \partial f(x)/\partial x_i$ is the partial derivative of the production function $f(x)$ with respect to the i th input quantity (x_i), and $f_{ij} \equiv \partial^2 f(x)/\partial x_i \partial x_j$ is the second partial derivative of the production function $f(x)$ with respect to the i th and j th input quantity (x_i, x_j) (Chambers, 1988, p. 31). As $f_{ij} = f_{ji}$ (Young's theorem), the direct elasticity of substitution is symmetric ($\sigma_{ij}^D = \sigma_{ji}^D$).

2.1.9.2 Allen elasticities of substitution

The *Allen* elasticity of substitution is another measure of the substitutability between two inputs. In contrast to the direct elasticity of substitution, the Allen elasticity of substitution takes into account that the quantities of the other inputs can be adjusted as well. Hence, in comparison to the *direct* elasticities of substitution, the *Allen* elasticities of substitution can be considered as long-run elasticities of substitution. It can be calculated by:

$$\sigma_{ij} = \frac{\sum_k f_k x_k}{x_i x_j} \frac{F_{ij}}{F}, \quad (2.21)$$

where F is the determinant of the bordered Hessian matrix B with

$$B = \begin{bmatrix} 0 & f_1 & f_2 & \dots & f_N \\ f_1 & f_{11} & f_{12} & \dots & f_{1N} \\ f_2 & f_{12} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_N & f_{1N} & f_{2N} & \dots & f_{NN} \end{bmatrix}, \quad (2.22)$$

and F_{ij} is the co-factor of f_{ij} , i.e.¹

$$F_{ij} = (-1)^{i+j} \cdot \begin{vmatrix} 0 & f_1 & f_2 & \dots & f_{j-1} & f_{j+1} & \dots & f_N \\ f_1 & f_{11} & f_{12} & \dots & f_{1,j-1} & f_{1,j+1} & \dots & f_{1N} \\ f_2 & f_{12} & f_{22} & \dots & f_{2,j-1} & f_{2,j+1} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ f_{i-1} & f_{1,i-1} & f_{2,i-1} & \dots & f_{i-1,j-1} & f_{i-1,j+1} & \dots & f_{i-1,N} \\ f_{i+1} & f_{1,i+1} & f_{2,i+1} & \dots & f_{i+1,j-1} & f_{i+1,j+1} & \dots & f_{i+1,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ f_N & f_{1N} & f_{2N} & \dots & f_{j-1,N} & f_{j+1,N} & \dots & f_{NN} \end{vmatrix}. \quad (2.23)$$

As the bordered Hessian matrix is symmetric, the co-factors are also symmetric ($F_{ij} = F_{ji}$) so that also the Allen elasticities of substitution are symmetric ($\sigma_{ij} = \sigma_{ji}$).

In case of two inputs ($x = (x_1, x_2)$), $\sum_k f_k x_k = f_1 x_1 + f_2 x_2$, $x_i x_j = x_1 x_2$, $F_{12} = f_1 f_2$, and $F = 2f_{12}f_1f_2 - f_{11}f_2^2 - f_{22}f_1^2$ so that the direct elasticity of substitution is identical to the Allen elasticity of substitution ($\sigma_{ij} = \sigma_{ij}^D$).

Following condition holds for Allen elasticities of substitution:

$$\sum_i K_i \sigma_{ij} = 0 \text{ with } K_i = \frac{f_i x_i}{\sum_k f_k x_k} \quad (2.24)$$

(see [Chambers, 1988](#), p. 35).

2.1.9.3 Morishima elasticities of substitution

The Morishima elasticity of substitution is a third measure of the substitutability between two inputs. It can be calculated by:

$$\sigma_{ij}^M = \frac{f_j}{x_i} \frac{F_{ij}}{F} - \frac{f_j}{x_j} \frac{F_{jj}}{F}, \quad (2.25)$$

where F_{ij} and F are defined as above. In contrast to the direct elasticity of substitution and the Allen elasticity of substitution, the Morishima elasticity of substitution is usually *not* symmetric

¹The exponent of (-1) usually is the sum of the number of the deleted row ($i+1$) and the number of the deleted column ($j+1$), i.e. $i+j+2$. In our case, we can simplify this to $i+j$, because $(-1)^{i+j+2} = (-1)^{i+j} \cdot (-1)^2 = (-1)^{i+j}$.

$(\sigma_{ij}^M \neq \sigma_{ji}^M)$.

From the above definition of the Morishima elasticities of substitution (2.25), we can derive the relationship between the Morishima elasticities of substitution and the Allen elasticities of substitution:

$$\sigma_{ij}^M = \frac{f_j x_j}{\sum_k f_k x_k} \frac{\sum_k f_k x_k}{x_i x_j} \frac{F_{ij}}{F} - \frac{f_j x_j}{\sum_k f_k x_k} \frac{\sum_k f_k x_k}{x_j^2} \frac{F_{jj}}{F} \quad (2.26)$$

$$= \frac{f_j x_j}{\sum_k f_k x_k} \sigma_{ij} - \frac{f_j x_j}{\sum_k f_k x_k} \sigma_{jj} \quad (2.27)$$

$$= \frac{f_j x_j}{\sum_k f_k x_k} (\sigma_{ij} - \sigma_{jj}), \quad (2.28)$$

where σ_{jj} can be calculated as the Allen elasticities of substitution with equation (2.21), but does not have an economic meaning.

In case of two inputs ($x = (x_1, x_2)$), the Morishima elasticities of substitution are symmetric and equal to the direct and Allen elasticities of substitution:

$$\sigma_{12}^M = \frac{f_2}{x_1} \frac{F_{12}}{F} - \frac{f_2}{x_2} \frac{F_{22}}{F} \quad (2.29)$$

$$= \frac{F_{12}}{F} \left(\frac{f_2}{x_1} - \frac{f_2}{x_2} \frac{F_{22}}{F_{12}} \right) \quad (2.30)$$

$$= \frac{F_{12}}{F} \left(\frac{f_2}{x_1} + \frac{f_2}{x_2} \frac{f_1^2}{f_1 f_2} \right) \quad (2.31)$$

$$= \frac{F_{12}}{F} \left(\frac{f_2}{x_1} + \frac{f_1}{x_2} \right) \quad (2.32)$$

$$= \frac{F_{12}}{F} \left(\frac{f_2 x_2}{x_1 x_2} + \frac{f_1 x_1}{x_1 x_2} \right) \quad (2.33)$$

$$= \frac{f_1 x_1 + f_2 x_2}{x_1 x_2} \frac{F_{12}}{F} \quad (2.34)$$

$$= \sigma_{12} = \sigma_{12}^D. \quad (2.35)$$

2.1.10 Profit maximization

We assume that the firms maximize their profit. The firm's profit is given by:

$$\pi = p y - \sum_i w_i x_i, \quad (2.36)$$

where p is the price of the output and w_i is the price of the i th input. If the firm faces output price p and input prices w_i , we can calculate the maximum profit that can be obtained by the firm by solving following optimization problem:

$$\max_{y, x} p y - \sum_i w_i x_i, \text{ s.t. } y = f(x) \quad (2.37)$$

This restricted maximization can be transformed into an unrestricted optimization by replacing y by the production function:

$$\max_x p f(x) - \sum_i w_i x_i \quad (2.38)$$

Hence, the first-order conditions are:

$$\frac{\partial \pi}{\partial x_i} = p \frac{\partial f(x)}{\partial x_i} - w_i = p MP_i - w_i = 0 \quad (2.39)$$

so that we get:

$$w_i = p MP_i = MVP_i \quad (2.40)$$

where $MVP_i = p(\partial y / \partial x_i)$ is the marginal value product of the i th input.

2.1.11 Cost minimization

Now, we assume that the firms take total output as given (e.g. because production is restricted by a quota) and try to produce this output quantity with minimal costs. The total cost is given by:

$$c = \sum_i w_i x_i, \quad (2.41)$$

where w_i is the price of the i th input.

If the firm faces input prices w_i and wants to produce y units of output, the minimum costs can be obtained by:

$$\min_x \sum_i w_i x_i, \text{ s.t. } y = f(x) \quad (2.42)$$

This restricted minimization can be solved by using the Lagrangian approach:

$$\mathcal{L} = \sum_i w_i x_i + \lambda (y - f(x)) \quad (2.43)$$

So that the first-order conditions are:

$$\frac{\partial \mathcal{L}}{\partial x_i} = w_i - \lambda \frac{\partial f(x)}{\partial x_i} = w_i - \lambda MP_i = 0 \quad (2.44)$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = y - f(x) = 0 \quad (2.45)$$

From the first-order conditions (2.44), we get:

$$w_i = \lambda MP_i \quad (2.46)$$

and

$$\frac{w_i}{w_j} = \frac{\lambda MP_i}{\lambda MP_j} = \frac{MP_i}{MP_j} = -MRTS_{ji} \quad (2.47)$$

2 Primal Approach: Production Function

As profit maximization implies producing the optimal output quantity with minimum costs, the first-order conditions for the optimal input combinations (2.47) can be obtained not only from cost minimization but also from the first-order conditions for profit maximization (2.40):

$$\frac{w_i}{w_j} = \frac{MVP_i}{MVP_j} = \frac{p MP_i}{p MP_j} = \frac{MP_i}{MP_j} = -MRTS_{ji} \quad (2.48)$$

Equations (2.47) and (2.48) can also be expressed in terms of the RMRTS instead of the MRTS:

$$-RMRTS_{ji} = -\frac{\partial x_j}{\partial x_i} \frac{x_i}{x_j} = -MRTS_{ji} \frac{x_i}{x_j} = \frac{w_i}{w_j} \frac{x_i}{x_j} = \frac{w_i}{w_j} \frac{x_i}{x_j} = \frac{\frac{w_i x_i}{c}}{\frac{w_j x_j}{c}} = \frac{s_i}{s_j}, \quad (2.49)$$

where $s_i = w_i x_i / c$ are the cost shares. Hence, for a profit maximising or cost minimizing producer, the ratio of the cost shares of two inputs must be equal to the negative value of the (reversed) relative marginal rate of technical substitution (RMRTS).

As the cost shares sum up to one, i.e., $\sum_{i=1}^N s_i = 1$, we can revise equation (2.49) to:

$$s_i = -RMRTS_{ji} s_j = \frac{\epsilon_i}{\epsilon_j} s_j \quad (2.50)$$

$$= \frac{\epsilon_i}{\epsilon_j} \left(1 - \sum_{k \in \{1, \dots, N\} \setminus j} s_k \right) \quad (2.51)$$

$$= \frac{\epsilon_i}{\epsilon_j} \left(1 - \sum_{k \in \{1, \dots, N\} \setminus j} \frac{\epsilon_k}{\epsilon_i} s_i \right) \quad (2.52)$$

$$= \frac{\epsilon_i}{\epsilon_j} \left(1 - \frac{s_i}{\epsilon_i} \sum_{k \in \{1, \dots, N\} \setminus j} \epsilon_k \right) \quad (2.53)$$

$$= \frac{\epsilon_i}{\epsilon_j} - \frac{s_i}{\epsilon_j} \sum_{k \in \{1, \dots, N\} \setminus j} \epsilon_k \quad (2.54)$$

$$s_i + \frac{s_i}{\epsilon_j} \sum_{k \in \{1, \dots, N\} \setminus j} \epsilon_k = \frac{\epsilon_i}{\epsilon_j} \quad (2.55)$$

$$s_i \left(1 + \frac{1}{\epsilon_j} \sum_{k \in \{1, \dots, N\} \setminus j} \epsilon_k \right) = \frac{\epsilon_i}{\epsilon_j} \quad (2.56)$$

$$s_i = \frac{\epsilon_i}{\epsilon_j \left(1 + \frac{1}{\epsilon_j} \sum_{k \in \{1, \dots, N\} \setminus j} \epsilon_k \right)} \quad (2.57)$$

$$= \frac{\epsilon_i}{\epsilon_j + \sum_{k \in \{1, \dots, N\} \setminus j} \epsilon_k} \quad (2.58)$$

$$= \frac{\epsilon_i}{\sum_{k=1}^N \epsilon_k} \quad (2.59)$$

$$= \frac{\epsilon_i}{\epsilon} \quad (2.60)$$

Hence, for a profit maximising or cost minimizing producer, the the cost share of each input must

be equal to the output elasticity of this input divided by the elasticity of scale.

2.1.12 Derived input demand functions and output supply functions

In this section, we will analyze how profit maximizing or cost minimizing firms react on changing prices and on changing output quantities.

2.1.12.1 Derived from profit maximization

If we replace the marginal products in the first-order conditions for profit maximization (2.40) by the equations for calculating these marginal products and then solve this system of equations for the input quantities, we get the input demand functions:

$$x_i = x_i(p, w), \quad (2.61)$$

where $w = [w_i]$ is the vector of all input prices. The input demand functions indicate the optimal input quantities (x_i) given the output price (p) and all input prices (w). We can obtain the output supply function from the production function by replacing all input quantities by the corresponding input demand functions:

$$y = f(x(p, w)) = y(p, w), \quad (2.62)$$

where $x(p, w) = [x_i(p, w)]$ is the set of all input demand functions. The output supply function indicates the optimal output quantity (y) given the output price (p) and all input prices (w). Hence, the input demand and output supply functions can be used to analyze the effects of prices on the (optimal) input use and output supply. In economics, the effects of price changes are usually measured in terms of price elasticities. These price elasticities can measure the effects of the input prices on the input quantities:

$$\epsilon_{ij}(p, w) = \frac{\partial x_i(p, w)}{\partial w_j} \frac{w_j}{x_i(p, w)}, \quad (2.63)$$

the effects of the input prices on the output quantity (expected to be non-positive):

$$\epsilon_{yj}(p, w) = \frac{\partial y(p, w)}{\partial w_j} \frac{w_j}{y(p, w)}, \quad (2.64)$$

the effects of the output price on the input quantities (expected to be non-negative):

$$\epsilon_{ip}(p, w) = \frac{\partial x_i(p, w)}{\partial p} \frac{p}{x_i(p, w)}, \quad (2.65)$$

and the effect of the output price on the output quantity (expected to be non-negative):

$$\epsilon_{yp}(p, w) = \frac{\partial y(p, w)}{\partial p} \frac{p}{y(p, w)}. \quad (2.66)$$

The effect of an input price on the optimal quantity of the same input is expected to be non-positive ($\epsilon_{ii}(p, w) \leq 0$). If the cross-price elasticities between two inputs i and j are positive ($\epsilon_{ij}(p, w) \geq 0$, $\epsilon_{ji}(p, w) \geq 0$), they are considered as gross substitutes. If the cross-price elasticities between two inputs i and j are negative ($\epsilon_{ij}(p, w) \leq 0$, $\epsilon_{ji}(p, w) \leq 0$), they are considered as gross complements.

2.1.12.2 Derived from cost minimization

If we replace the marginal products in the first-order conditions for cost minimization (2.46) by the equations for calculating these marginal products and solve this system of equations for the input quantities, we get the *conditional* input demand functions:

$$x_i = x_i(w, y) \quad (2.67)$$

These input demand functions are called “conditional,” because they indicate the optimal input quantities (x_i) given all input prices (w) and conditional on the fixed output quantity (y). The conditional input demand functions can be used to analyze the effects of input prices on the (optimal) input use if the output quantity is given. The effects of price changes on the optimal input quantities can be measured by conditional price elasticities:

$$\epsilon_{ij}(w, y) = \frac{\partial x_i(w, y)}{\partial w_j} \frac{w_j}{x_i(w, y)} \quad (2.68)$$

The effect of the output quantity on the optimal input quantities can also be measured in terms of elasticities (expected to be positive):

$$\epsilon_{iy}(w, y) = \frac{\partial x_i(w, y)}{\partial y} \frac{y}{x_i(w, y)}. \quad (2.69)$$

The conditional effect of an input price on the optimal quantity of the same input is expected to be non-positive ($\epsilon_{ii}(w, y) \leq 0$). If the conditional cross-price elasticities between two inputs i and j are positive ($\epsilon_{ij}(w, y) \geq 0$, $\epsilon_{ji}(w, y) \geq 0$), they are considered as net substitutes. If the conditional cross-price elasticities between two inputs i and j are negative ($\epsilon_{ij}(w, y) \leq 0$, $\epsilon_{ji}(w, y) \leq 0$), they are considered as net complements.

2.1.13 Suitability of the production function for econometric applications

Given the microeconomic theory of the production function and the assumptions of standard econometric methods such as ordinary least squares (OLS), it is appropriate to estimate a pro-

duction function with these econometric methods if several conditions are fulfilled. The most important and relevant conditions are:

1. The inputs and outputs are rather similar across all firms in the data set (within each input or output category).
2. The production conditions are rather similar for all firms in the data set (unless the empirical specification appropriately accounts for these differences).
3. The data set contains variables that indicate—in reasonably precise ways—the quantities of all relevant inputs, the (overall) output quantity, and all relevant ‘environmental’ variables that can be used for controlling for differences in production conditions.
4. Random / representative sample of the firms to be analysed.
5. All firms in the data set produce the maximum obtainable output quantity given the input quantities.
6. No firm in the data set has all input quantities equal to zero but produces a strictly positive output quantity (i.e., weak essentiality must be fulfilled at all observations).
7. The functional form specification of the estimated production function is sufficiently similar to the ‘true’ production function in the ‘real world’.
8. No perfect multicollinearity (i.e., no Leontief production function).
9. Homoscedasticity (often fulfilled to a higher degree when the logarithm of the output quantity instead of the non-transformed output quantity is used as dependent variable).
10. All input quantities (i.e., all explanatory variables) are uncorrelated with the error term, e.g., unobserved heterogeneity between firms (e.g., differences in management quality or production conditions such as soil quality or weather in agricultural production) is unrelated to the input quantities. This assumption is usually not fulfilled if the output quantity is exogenously given so that producers must adjust the input quantities to account for unobserved heterogeneity.

2.2 Productivity measures

2.2.1 Average products

We calculate the average products of the three inputs for each firm in the data set by equation 2.2:

```
> dat$apCap <- dat$qOut / dat$qCap  
> dat$apLab <- dat$qOut / dat$qLab  
> dat$apMat <- dat$qOut / dat$qMat
```

We can visualize these average products with histograms that can be created with the command `hist`.

```
> hist( dat$apCap )
> hist( dat$apLab )
> hist( dat$apMat )
```

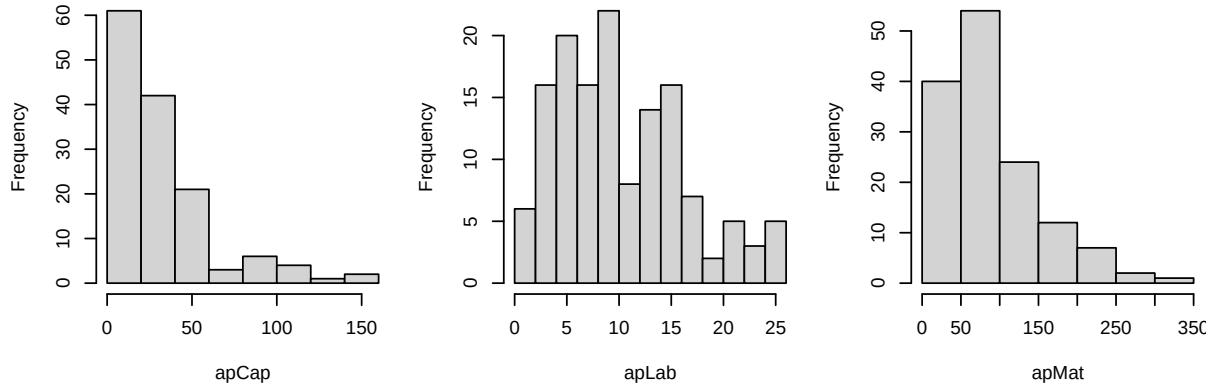


Figure 2.1: Average products

The resulting graphs are shown in figure 2.1. These graphs show that average products (partial productivities) vary considerably between firms. Most firms in our data set produce on average between 0 and 40 units of output per unit of capital, between 2 and 16 units of output per unit of labor, and between 0 and 100 units of output per unit of materials. Looking at each average product separately, There are usually many firms with medium to low partial productivity measures and only a few firms with high partial productivity measures.

We can find the firms with the highest partial productivities with the commands:

```
> which.max( dat$apCap )

[1] 132

> which.max( dat$apLab )

[1] 7

> which.max( dat$apMat )

[1] 83
```

Firm number 132 has the highest capital productivity, firm number 7 has the highest labor productivity, and firm number 83 has the highest materials productivity.

The relationships between the average products can be visualized by scatter plots:

2 Primal Approach: Production Function

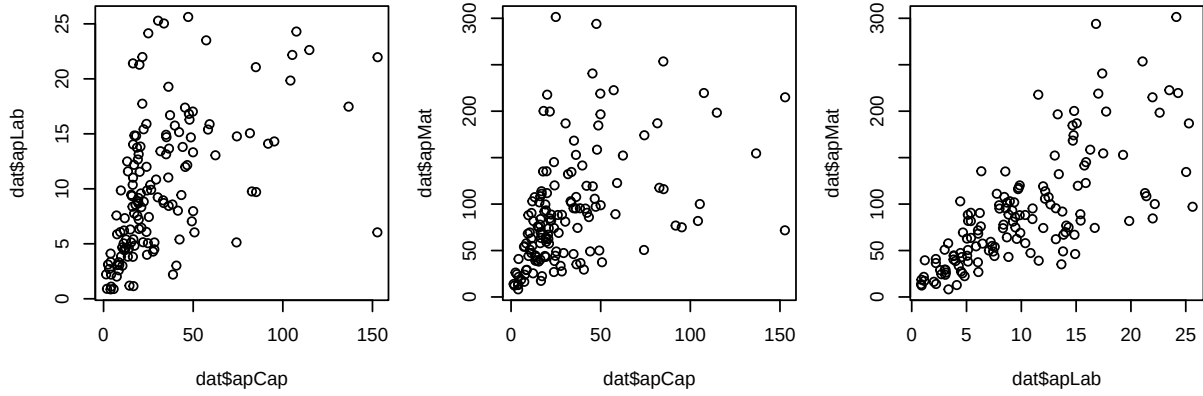


Figure 2.2: Relationships between average products

```
> plot( dat$apCap, dat$apLab )
> plot( dat$apCap, dat$apMat )
> plot( dat$apLab, dat$apMat )
```

The resulting graphs are shown in figure 2.2. They show that the average products of the three inputs are positively correlated.

As the units of measurements of the input and output quantities in our data set cannot be interpreted in practical terms, the interpretation of the size of the average products is practically not useful. However, they can be used to make comparisons between firms. For instance, the interrelation between average products and firm size can be analyzed. A possible (although not perfect) measure of size of the firms in our data set is the total output.

```
> plot( dat$qOut, dat$apCap, log = "x" )
> plot( dat$qOut, dat$apLab, log = "x" )
> plot( dat$qOut, dat$apMat, log = "x" )
```

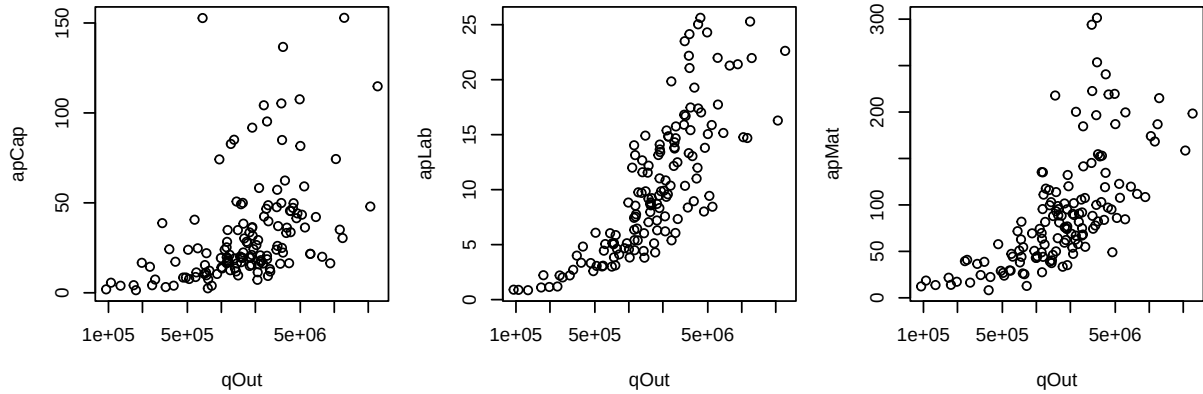


Figure 2.3: Average products for different firm sizes

The resulting graphs are shown in figure 2.3. These graphs show that the larger firms (i.e. firms with larger output quantities) produce also a larger output quantity per unit of each input. This

is not really surprising, because the output quantity is in the numerator of equation (2.2) so that the average products are necessarily positively related to the output quantity for a given input quantity.

2.2.2 Total factor productivity

After calculating a quantity index of all inputs (see section 1.5.4), we can use equation 2.3 to calculate the total factor productivity, where we arbitrarily choose the Fisher quantity index:

```
> dat$tfp <- dat$qOut / dat$X
```

The variation of the total factor productivities can be visualized as before in a histogram:

```
> hist( dat$tfp )
```

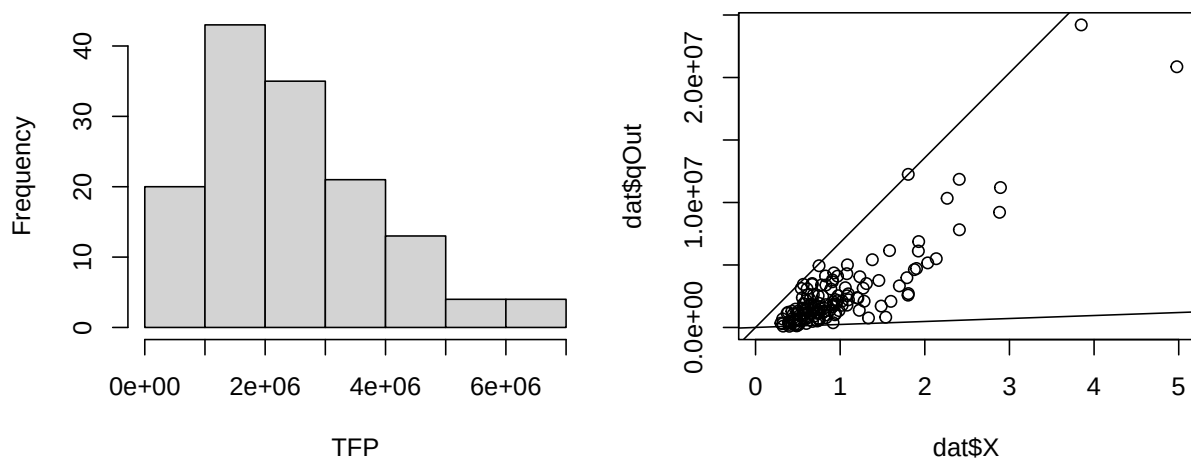


Figure 2.4: Total factor productivities

The resulting histogram is shown in the left panel of figure 2.4.

As the total factor productivity is the ratio between the aggregate output quantity and the aggregate input quantity, it can be illustrated by a scatter plot between the aggregate input quantity and the aggregate output quantity, whereas the slope of a line from the origin to a respective point in the scatter plot indicates the total factor productivity. The following commands create a scatter plot between the aggregate input quantity and the aggregate output quantity and add a line through the origin with the slope equal to the maximum total factor productivity as well as a line through the origin with the slope equal to the minimum total factor productivity in the data set:

```
> plot( dat$X, dat$qOut,
+       xlim = c( 0, max( dat$X ) ), ylim = c( 0, max( dat$qOut ) ) )
> abline( 0, max( dat$tfp ) )
> abline( 0, min( dat$tfp ) )
```


The resulting scatter plot is shown in the right panel of figure 2.4. Both parts of figure 2.4 indicate that total factor productivity considerably varies between firms.

Where do these large differences in (total factor) productivity come from? We can check the relation between total factor productivity and firm size by a scatter plot. We use two different measures of firm size: total output and aggregate input. The following commands produce scatter plots, where we set argument `log` of the `plot` command to the character string `"x"` so that the horizontal axis is measured in logarithmic terms and the dots (firms) are more equally spread:

```
> plot( dat$qOut, dat$tfp, log = "x" )
> plot( dat$X, dat$tfp, log = "x" )
```

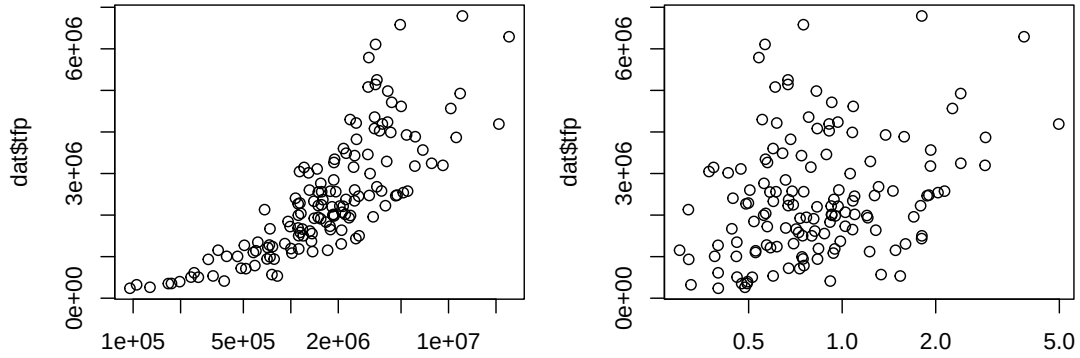


Figure 2.5: Firm size and total factor productivity

The resulting scatter plots are shown in figure 2.5. The scatter plot in the left panel clearly shows that the firms with larger output quantities also have a larger total factor productivity. This is not really surprising, because the output quantity is in the numerator of equation (2.3) so that the total factor productivity is necessarily positively related to the output quantity for given input quantities. The total factor productivity is only slightly positively related to the measure of aggregate input use.

We can also analyze whether the firms that use an advisory service have a higher total factor productivity than firms that do not use an advisory service. We can visualize and compare the total factor productivities of the two different groups of firms (with and without advisory service) using boxplot diagrams:

```
> boxplot( tfp ~ adv, data = dat )
> boxplot( log(qOut) ~ adv, data = dat )
> boxplot( log(X) ~ adv, data = dat )
```

The resulting boxplot graphic is shown on the left panel of figure 2.6. It suggests that the firms that use advisory service are slightly more productive than firms that do not use advisory service (at least when looking at the 25th percentile and the median).

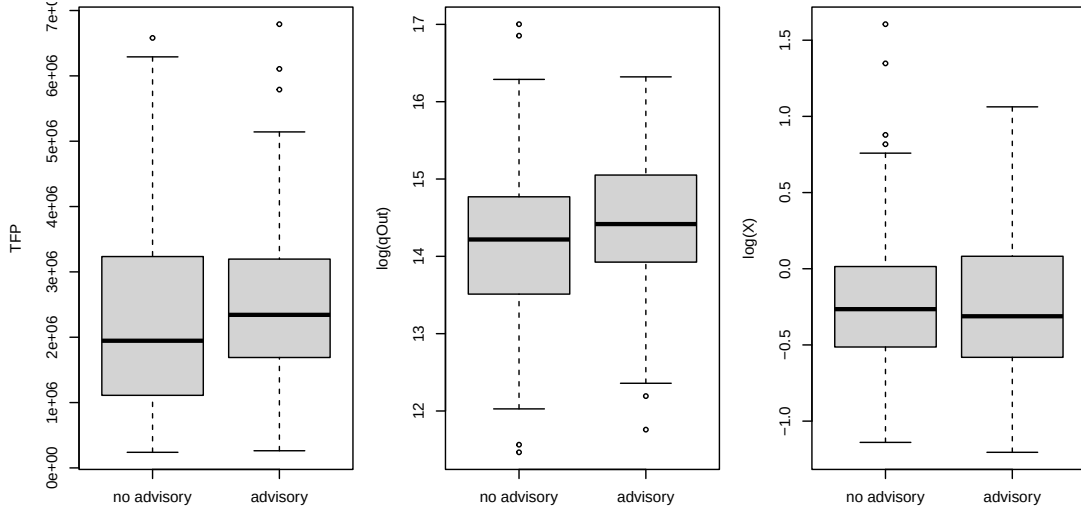


Figure 2.6: Total factor productivities and advisory service

However, these boxplots can only indicate a relationship between using advisory service and total factor productivity but they cannot indicate whether using an advisory service increases productivity (i.e. a causal effect). For instance, if larger firms are more likely to use an advisory service than smaller firms and larger firms have a higher total factor productivity than smaller firms, we expect that firms that use an advisory service have a higher productivity than smaller firms even if using an advisory service does not affect total factor productivity. However, this is not the case in our data set, because farms with and without advisory service use rather similar input quantities (see right panel of figure 2.6). As farms that use advisory service use similar input quantities but have a higher total factor productivity than farms without advisory service (see left panel of figure 2.6), they also have larger output quantities than corresponding farms without advisory service (see middle panel of figure 2.6). Furthermore, the causal effect of advisory service on total factor productivity might not be equal to the productivity difference between farms with and without advisory service, because it might be that the firms that anyway were the most productive were more (or less) likely to use advisory service than the firms that anyway were the least productive.

2.3 Linear production function

2.3.1 Specification

A linear production function with N inputs is defined as:

$$y = \beta_0 + \sum_{i=1}^N \beta_i x_i \quad (2.70)$$

2.3.2 Estimation

We can add a stochastic error term to this linear production function and estimate it for our data set using the command `lm`:

```
> prodLin <- lm( qOut ~ qCap + qLab + qMat, data = dat )
> summary( prodLin )
```

Call:

```
lm(formula = qOut ~ qCap + qLab + qMat, data = dat)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-3888955	-773002	86119	769073	7091521

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-1.616e+06	2.318e+05	-6.972	1.23e-10 ***
qCap	1.788e+00	1.995e+00	0.896	0.372
qLab	1.183e+01	1.272e+00	9.300	3.15e-16 ***
qMat	4.667e+01	1.123e+01	4.154	5.74e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1541000 on 136 degrees of freedom

Multiple R-squared: 0.7868, Adjusted R-squared: 0.7821

F-statistic: 167.3 on 3 and 136 DF, p-value: < 2.2e-16

2.3.3 Properties

As the coefficients of all three input quantities are positive, the monotonicity condition is (globally) fulfilled. However, the coefficient of the capital quantity is statistically not significantly different from zero. Therefore, we cannot be sure that the capital quantity has a positive effect on the output quantity. However, the result that the capital quantity does not have a statistically significant effect on the output quantity does not mean that the capital quantity in reality does not affect the output quantity—we just do not have sufficient (statistical) evidence to rule out that the capital quantity does not affect the output quantity. If the capital input in reality affects the output quantity but we remove it from our regression model, the estimation results will be biased and inconsistent. On the other hand, if the capital input in reality does not affect the output quantity but we keep it as explanatory variable in our regression model, the estimation results are inefficient but they are still unbiased and consistent. Furthermore, based on our knowledge about apple production, we expect that the capital input (consisting of land, fruit

trees, machinery, ...) clearly affects the output quantity. For these reasons, we do not remove the capital quantity from the regression model but keep it as explanatory variable.

As every linear function is concave (and convex), also our estimated linear production is concave and hence, also quasiconcave. As the isoquants of linear productions functions are linear, the input requirement sets are always convex (and concave).

Our estimated linear production function does not fulfill the weak essentiality assumption, because the intercept is different from zero. The production technology described by a linear production function with more than one (relevant) input never shows strict essentiality.

The input requirement sets derived from linear production functions are always closed and non-empty for $y > 0$ if weak essentiality is fulfilled ($\alpha_0 = 0$) and strict monotonicity is fulfilled for at least one input ($\exists i \in \{1, \dots, N\} : \beta_i > 0$), as the input quantities must be non-negative ($x_i \geq 0 \forall i$).

The linear production function always returns finite, real, and single values for all non-negative and finite x . However, as the intercept of our estimated production function is negative, the non-negativity assumption is not fulfilled. A linear production function would return non-negative values for all non-negative and finite x if $\beta_0 \geq 0$ and the monotonicity condition is fulfilled ($\beta_i \geq 0 \forall i = 1, \dots, N$).

All linear production functions are continuous and twice-continuously differentiable.

2.3.4 Predicted output quantities

We can calculate the predicted (“fitted”) output quantities manually by taking the linear production function (2.70), the observed input quantities, and the estimated parameters, but it is easier to use the `fitted` method to obtain the predicted values of the dependent variable from an estimated model:

```
> dat$qOutLin <- fitted( prodLin )
> all.equal( dat$qOutLin, coef( prodLin )[ "(Intercept)" ] +
+   coef( prodLin )[ "qCap" ] * dat$qCap +
+   coef( prodLin )[ "qLab" ] * dat$qLab +
+   coef( prodLin )[ "qMat" ] * dat$qMat )

[1] TRUE
```

We can evaluate the “fit” of the model by comparing the observed with the fitted output quantities using the command `compPlot` (package *miscTools*):

```
> library( "miscTools" )
> compPlot( dat$qOut, dat$qOutLin )
> compPlot( dat$qOut[ dat$qOutLin > 0 ], dat$qOutLin[ dat$qOutLin > 0 ],
+   log = "xy" )
```

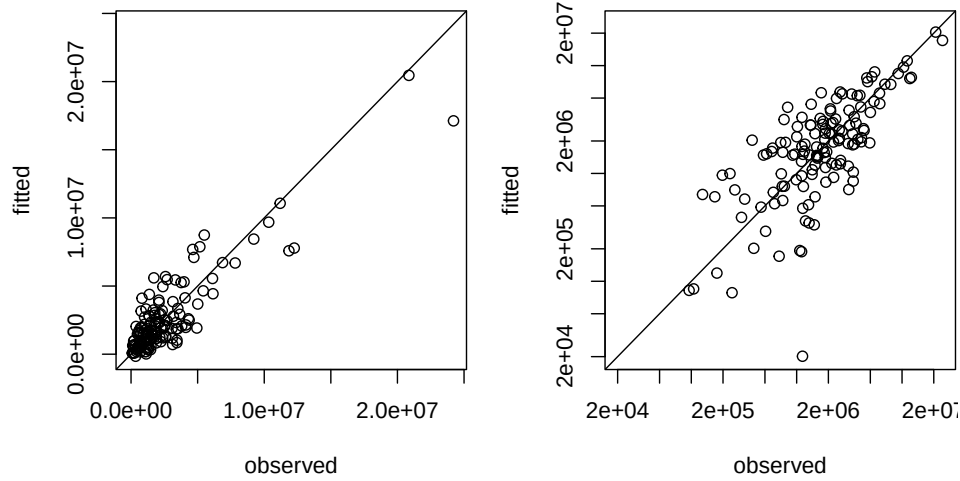


Figure 2.7: Linear production function: fit of the model

The resulting graphs are shown in figure 2.7. While the graph in the left panel uses a linear scale for the axes, the graph in the right panel uses a logarithmic scale for both axes. Hence, the deviations from the 45°-line illustrate the *absolute* deviations in the left panel and the *relative* deviations in the right panel. As the logarithm of non-positive values is undefined, we have to exclude observations with non-positive predicted output quantities in the graphs with logarithmic axes. The fit of the model looks okay in both scatter plots.

As negative output quantities would render the corresponding output elasticities useless, we have to carefully check the sign of the predicted output quantities:

```
> table( dat$qOutLin >= 0 )

FALSE  TRUE
    1    139
```

One predicted output quantity is negative.

2.3.5 Marginal products

In the linear production function, the marginal products are equal to coefficients of the corresponding input quantities.

$$MP_i = \frac{\partial y}{\partial x_i} = \alpha_i \quad (2.71)$$

Hence, if a firm increases capital input by one unit, the output will increase by 1.79 units; if a firm increases labor input by one unit, the output will increase by 11.83 units; and if a firm increases materials input by one unit, the output will increase by 46.67 units.

2.3.6 Output elasticities

As we do not know the units of measurements of the input and output quantities, the interpretation of the marginal products is practically not very useful. Therefore, we calculate the output elasticities (partial production elasticities) of the three inputs.

$$\epsilon_i = \frac{\partial y}{\partial x_i} \frac{x_i}{y} = MP_i \frac{x_i}{y} = \frac{MP_i}{AP_i} \quad (2.72)$$

As the output elasticities depend on the input and output quantities and these quantities generally differ between firms, also the output elasticities differ between firms. Hence, we can calculate them for each firm in the sample:

```
> dat$eCap <- coef(prodLin)["qCap"] * dat$qCap / dat$qOut
> dat$eLab <- coef(prodLin)["qLab"] * dat$qLab / dat$qOut
> dat$eMat <- coef(prodLin)["qMat"] * dat$qMat / dat$qOut
```

We can obtain their mean values by:

```
> colMeans( subset( dat, , c( "eCap", "eLab", "eMat" ) ) )
```

```
      eCap      eLab      eMat
0.1202721 2.0734793 0.8631936
```

However, these mean values are distorted by outliers (see figure 2.8). Therefore, we calculate the median values of the the output elasticities:

```
> colMedians( subset( dat, , c( "eCap", "eLab", "eMat" ) ) )
```

```
      eCap      eLab      eMat
0.08063406 1.28627208 0.58741460
```

Hence, if a firm increases capital input by one percent, the output will usually increase by around 0.08 percent; if the firm increases labor input by one percent, the output will often increase by around 1.29 percent; and if the firm increases materials input by one percent, the output will often increase by around 0.59 percent.

We can visualize (the variation of) these output elasticities with histograms. The user can modify the desired number of bars in the histogram by adding an integer number as additional argument:

```
> hist( dat$eCap )
> hist( dat$eLab, 20 )
> hist( dat$eMat, 20 )
```

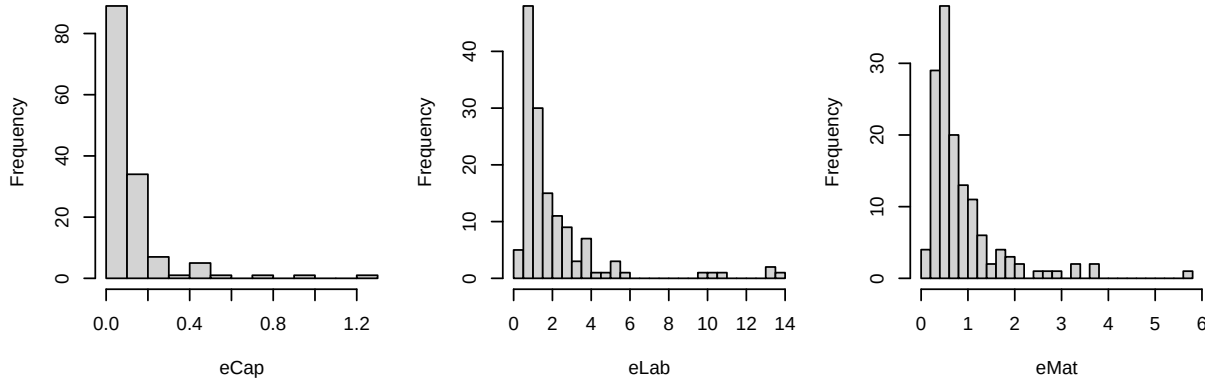


Figure 2.8: Linear production function: output elasticities

The resulting graphs are shown in figure 2.8. If the firms increase capital input by one percent, the output of most firms will increase by between 0 and 0.2 percent; if the firms increase labor input by one percent, the output of most firms will increase by between 0.5 and 3 percent; and if the firms increase materials input by one percent, the output of most firms will increase by between 0.2 and 1.2 percent. While the marginal effect of capital on the output is rather small for most firms, there are many firms with implausibly high output elasticities of labor and materials ($\epsilon_i > 1$). This might indicate that the true production technology cannot be reasonably approximated by a linear production function.

In contrast to a pure theoretical microeconomic model, our empirically estimated model includes a stochastic error term so that the observed output quantities (y) are not necessarily equal to the output quantities that are predicted by the model ($\hat{y} = f(x)$). This error term comes from, e.g., measurement errors, omitted explanatory variables, (good or bad) luck, or unusual(ly) (good or bad) weather conditions. The better the fit of our model, i.e. the higher the R^2 value, the smaller is the difference between the observed and the predicted output quantities. If we “believe” in our estimated model, it would be more consistent with microeconomic theory, if we use the *predicted* output quantities and disregard the stochastic error term.

We can calculate the output elasticities based on the predicted output quantities (see section 2.3.4) rather than the observed output quantities:

```
> dat$eCapFit <- coef(prodLin)["qCap"] * dat$qCap / dat$qOutLin
> dat$eLabFit <- coef(prodLin)["qLab"] * dat$qLab / dat$qOutLin
> dat$eMatFit <- coef(prodLin)["qMat"] * dat$qMat / dat$qOutLin
> colMeans( subset( dat, , c( "eCapFit", "eLabFit", "eMatFit" ) ) )

eCapFit  eLabFit  eMatFit
0.1421941 2.2142092 0.9784056

> colMedians( subset( dat, , c( "eCapFit", "eLabFit", "eMatFit" ) ) )
```

```
eCapFit    eLabFit    eMatFit
0.07407719 1.21044421 0.58821500
```

```
> hist( dat$eCapFit, 20 )
> hist( dat$eLabFit, 20 )
> hist( dat$eMatFit, 20 )
```

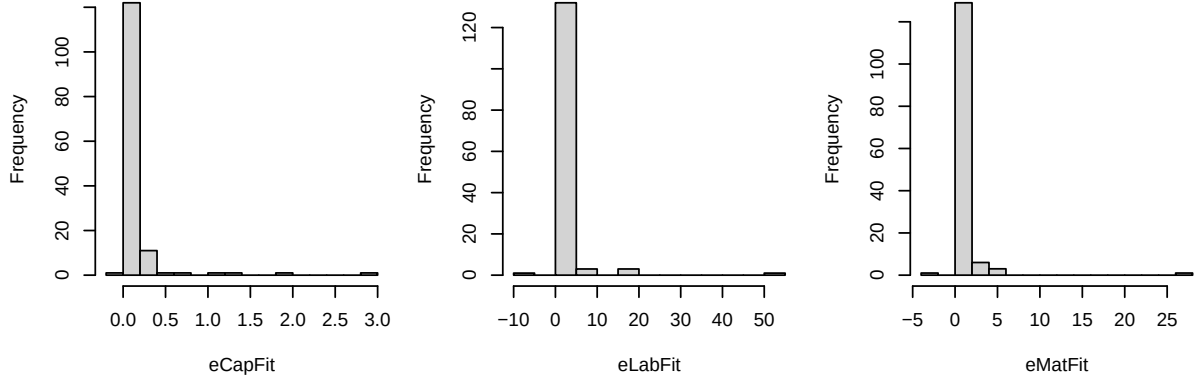


Figure 2.9: Linear production function: output elasticities based on predicted output quantities

The resulting graphs are shown in figure 2.9. While the choice of the variable for the output quantity (observed vs. predicted) only has a minor effect on the mean and median values of the output elasticities, the ranges of the output elasticities that are calculated from the predicted output quantities are much larger than the ranges of the output elasticities that are calculated from the observed output quantities. Due to 1 negative predicted output quantity, the output elasticities of this observation are also negative.

2.3.7 Elasticity of scale

The elasticity of scale is the sum of all output elasticities

$$\epsilon = \sum_i \epsilon_i \quad (2.73)$$

Hence, the elasticities of scale of all firms in the sample can be calculated by:

```
> dat$eScale <- with( dat, eCap + eLab + eMat )
> dat$eScaleFit <- with( dat, eCapFit + eLabFit + eMatFit )
```

The mean and median values of the elasticities of scale can be calculated by

```
> colMeans( subset( dat, , c( "eScale", "eScaleFit" ) ) )

eScale eScaleFit
3.056945 3.334809
```



```
> colMedians( subset( dat, , c( "eScale", "eScaleFit" ) ) )

eScale eScaleFit
1.941536 1.864253
```

Hence, if a firm increases all input quantities by one percent, the output quantity will usually increase by around 1.9 percent. This means that most firms have increasing returns to scale and hence, the firms could increase productivity by increasing the firm size (i.e. increasing all input quantities).

The (variation of the) elasticities of scale can be visualized with histograms:

```
> hist( dat$eScale, 30 )
> hist( dat$eScaleFit, 50 )
> hist( dat$eScaleFit[ dat$eScaleFit > 0 & dat$eScaleFit < 15 ], 30 )
```

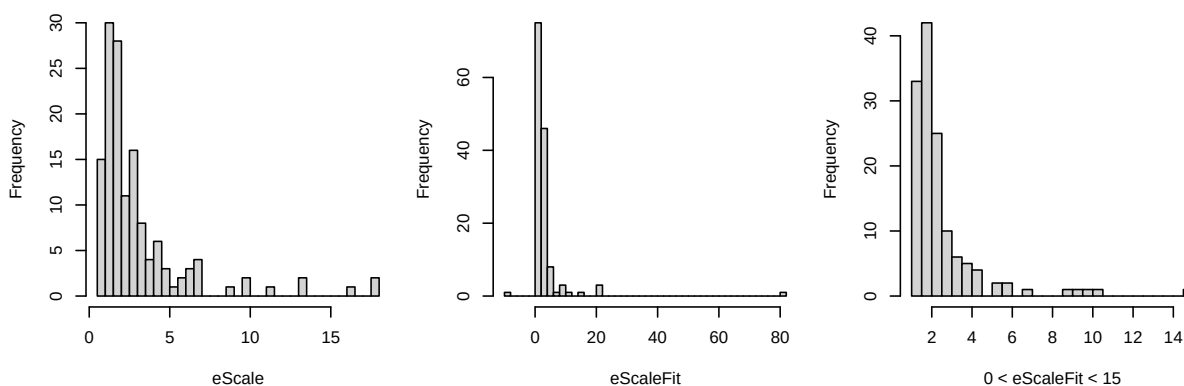


Figure 2.10: Linear production function: elasticities of scale

The resulting graphs are shown in figure 2.10. As the predicted output quantity of 1 firm is negative, the elasticity of scale of this observation also is negative, if the predicted output quantities are used for the calculation. However, all remaining elasticities of scale that are based on the predicted output quantities are larger than one, which indicates increasing returns to scale. In contrast, 15 (out of 140) elasticities of scale that are calculated with the observed output quantities indicate decreasing returns to scale. However, both approaches indicate that most firms have an elasticity of scale between one and two. Hence, if these firms increase all input quantities by one percent, the output of most firms will increase by between 1 and 2 percent. Some firms even have an elasticity of scale larger than five, which is very implausible and might indicate that the true production technology cannot be reasonably approximated by a linear production function.

Information on the optimal firm size can be obtained by analyzing the interrelationship between firm size and the elasticity of scale:

```
> plot( dat$qOut, dat$eScale, log = "x" )
> abline( 1, 0 )
```

```
> plot( dat$X, dat$eScale, log = "x" )
> abline( 1, 0 )
> plot( dat$qOut, dat$eScaleFit, log = "x", ylim = c( 0, 15 ) )
> abline( 1, 0 )
> plot( dat$X, dat$eScaleFit, log = "x", ylim = c( 0, 15 ) )
> abline( 1, 0 )
```

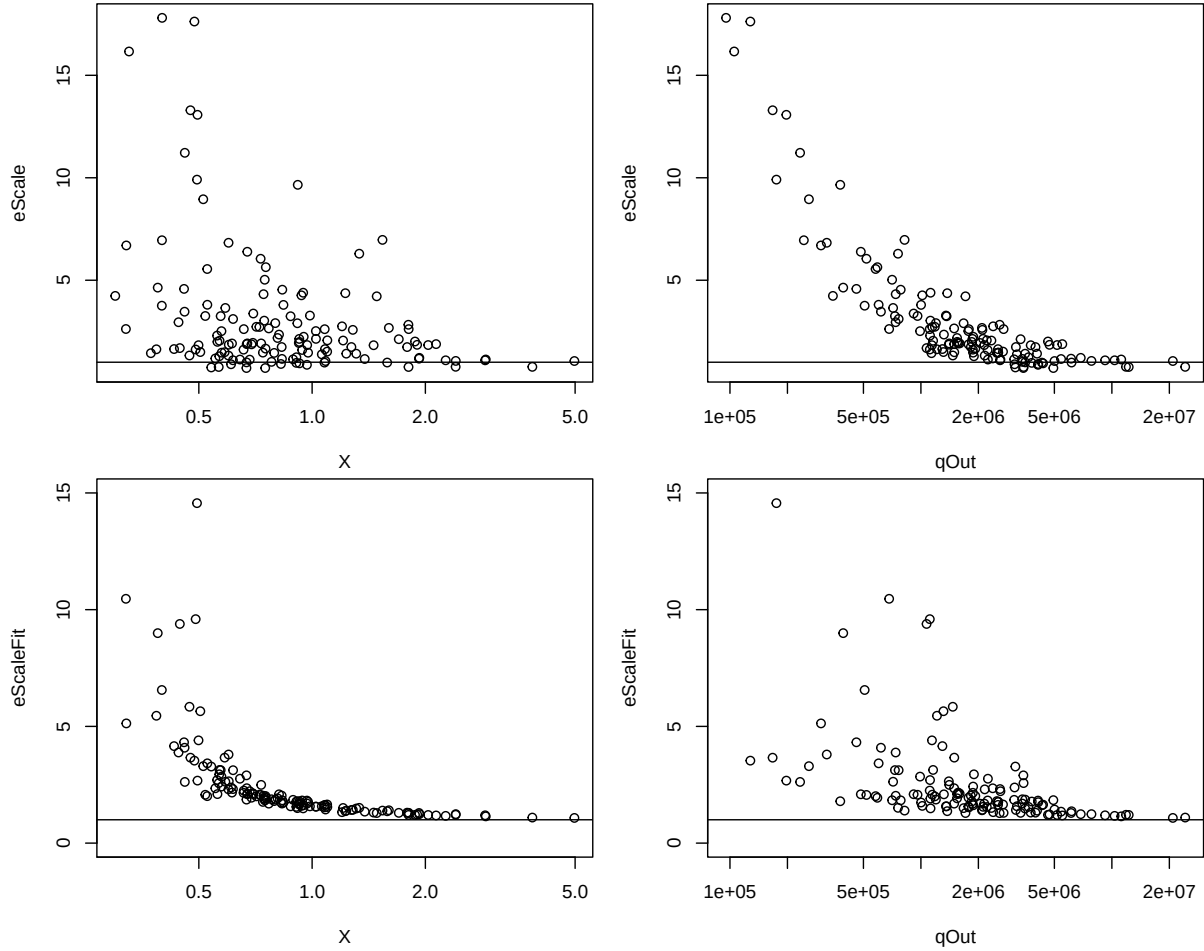


Figure 2.11: Linear production function: elasticities of scale for different firm sizes

The resulting graphs are shown in figure 2.11. They indicate that very small firms could enormously gain from increasing their size, while the benefits from increasing firm size decrease with size. Only a few elasticities of scale that are calculated with the observed output quantities indicate decreasing returns to scale so that productivity would decline when these firms increase their size. For all firms that use at least 2.1 times the input quantities of the average firm or produces more than 6,000,000 quantity units (approximately 6,000,000 Euros), the elasticities of scale that are based on the observed input quantities are very close to one. From this observation we could conclude that firms have their optimal size when they use at least 2.1 times the input quantities

of the average firm or produce at least 6,000,000 quantity units (approximately 6,000,000 Euros turn over). In contrast, the elasticities of scale that are based on the predicted output quantities are larger one even for the largest firms in the data set. From this observation, we could conclude that the even the largest firms in the sample would gain from growing in size and thus, the most productive scale size is larger than the size of the largest firms in the sample.

The high elasticities of scale explain why we found much higher partial productivities (average products) and total factor productivities for larger firms than for smaller firms.

2.3.8 Marginal rates of technical substitution

As the marginal products based on a linear production function are equal to the coefficients, we can calculate the MRTS (2.17) as follows:

```
> mrtsCapLab <- - coef(prodLin)["qLab"] / coef(prodLin)["qCap"]

      qLab
-6.615934

> mrtsLabCap <- - coef(prodLin)["qCap"] / coef(prodLin)["qLab"]

      qCap
-0.1511502

> mrtsCapMat <- - coef(prodLin)["qMat"] / coef(prodLin)["qCap"]

      qMat
-26.09666

> mrtsMatCap <- - coef(prodLin)["qCap"] / coef(prodLin)["qMat"]

      qCap
-0.03831908

> mrtsLabMat <- - coef(prodLin)["qMat"] / coef(prodLin)["qLab"]

      qMat
-3.944516

> mrtsMatLab <- - coef(prodLin)["qLab"] / coef(prodLin)["qMat"]

      qLab
-0.2535165
```

Hence, if a firm wants to reduce the use of labor by one unit, he/she has to use 6.62 additional units of capital in order to produce the same output as before. Alternatively, the firm can replace the unit of labor by using 0.25 additional units of materials. If the firm increases the use of labor by one unit, he/she can reduce capital by 6.62 units whilst still producing the same output as before. Alternatively, the firm can reduce materials by 0.25 units.

2.3.9 Relative marginal rates of technical substitution

We can calculate the RMRTS (2.18) derived from the linear production function as follows:

```
> dat$rmrtsCapLab <- - dat$eLab / dat$eCap
> dat$rmrtsLabCap <- - dat$eCap / dat$eLab
> dat$rmrtsCapMat <- - dat$eMat / dat$eCap
> dat$rmrtsMatCap <- - dat$eCap / dat$eMat
> dat$rmrtsLabMat <- - dat$eMat / dat$eLab
> dat$rmrtsMatLab <- - dat$eLab / dat$eMat
```

We can visualize (the variation of) these RMRTSs with histograms:

```
> hist( dat$rmrtsCapLab, 20 )
> hist( dat$rmrtsLabCap )
> hist( dat$rmrtsCapMat )
> hist( dat$rmrtsMatCap )
> hist( dat$rmrtsLabMat )
> hist( dat$rmrtsMatLab )
```

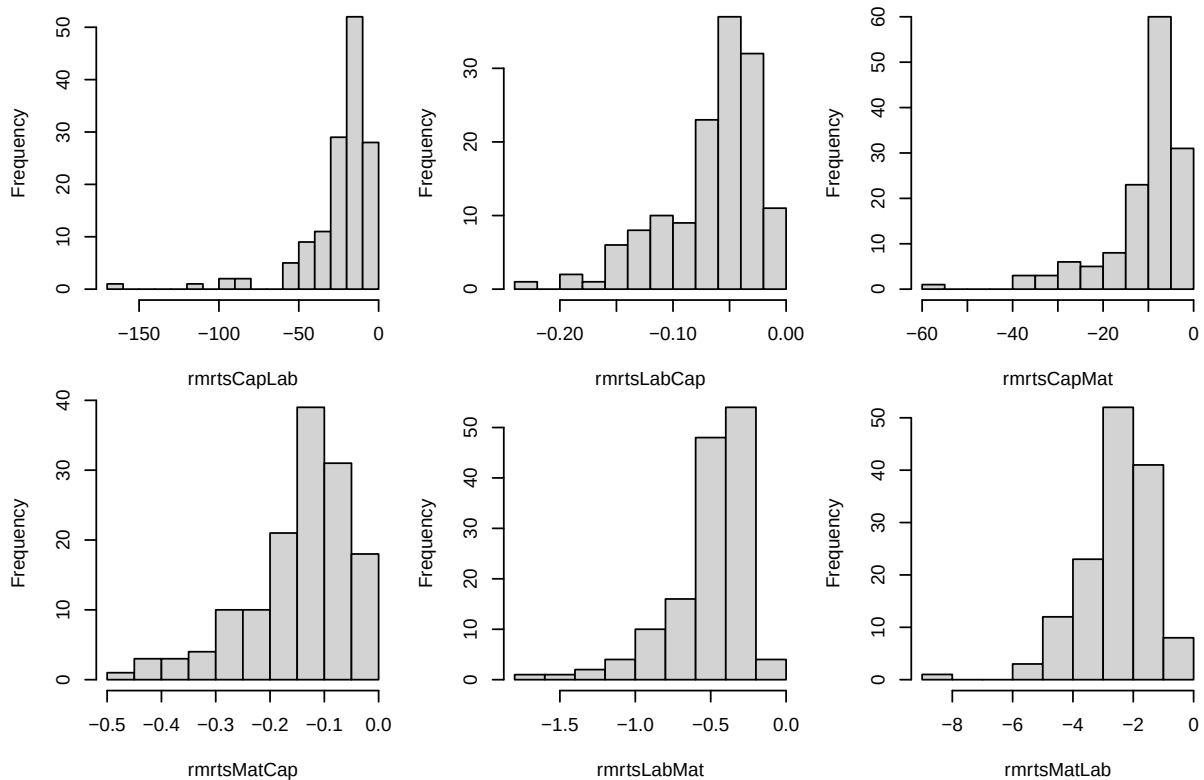


Figure 2.12: Linear production function: relative marginal rates of technical substitution (RMRTS)

The resulting graphs are shown in figure 2.12. According to the RMRTS based on the linear production function, most firms need between 20% more capital or around 2% more materials to compensate a 1% reduction of labor.

The mean and median values of these RMRTS are:

```
> colMeans( dat[, c( "rmrtsCapLab", "rmrtsLabCap", "rmrtsCapMat",
+   "rmrtsMatCap", "rmrtsLabMat", "rmrtsMatLab" ) ] )

rmrtsCapLab  rmrtsLabCap  rmrtsCapMat  rmrtsMatCap  rmrtsLabMat  rmrtsMatLab
-24.24019545  -0.06533753 -10.70239406  -0.14805675  -0.49845972  -2.50782286

> colMedians( dat[, c( "rmrtsCapLab", "rmrtsLabCap", "rmrtsCapMat",
+   "rmrtsMatCap", "rmrtsLabMat", "rmrtsMatLab" ) ] )

rmrtsCapLab  rmrtsLabCap  rmrtsCapMat  rmrtsMatCap  rmrtsLabMat  rmrtsMatLab
-17.70447629  -0.05648301  -7.95275036  -0.12574680  -0.43240915  -2.31262549
```

2.3.10 First-order conditions for profit maximization

In this section, we will check to what extent the first-order conditions for profit maximization (2.40) are fulfilled, i.e. to what extent the firms use the optimal input quantities. We do this by comparing the marginal value products of the inputs with the corresponding input prices. We can calculate the marginal value products by multiplying the marginal products by the output price:

```
> dat$mvpCap <- dat$pOut * coef(prodLin)["qCap"]
> dat$mvpLab <- dat$pOut * coef(prodLin)["qLab"]
> dat$mvpMat <- dat$pOut * coef(prodLin)["qMat"]
```

The command `compPlot` (package *miscTools*) can be used to compare the marginal value products with the corresponding input prices:

```
> compPlot( dat$pCap, dat$mvpCap )
> compPlot( dat$pLab, dat$mvpLab )
> compPlot( dat$pMat, dat$mvpMat )
> compPlot( dat$pCap, dat$mvpCap, log = "xy" )
> compPlot( dat$pLab, dat$mvpLab, log = "xy" )
> compPlot( dat$pMat, dat$mvpMat, log = "xy" )
```

The resulting graphs are shown in figure 2.13. The graphs on the left side indicate that the marginal value products of capital are sometimes lower but more often higher than the capital prices. The four other graphs indicate that the marginal value products of labor and materials are always higher than the labor prices and the materials prices, respectively. This indicates that some firms could increase their profit by using more capital and all firms could increase their

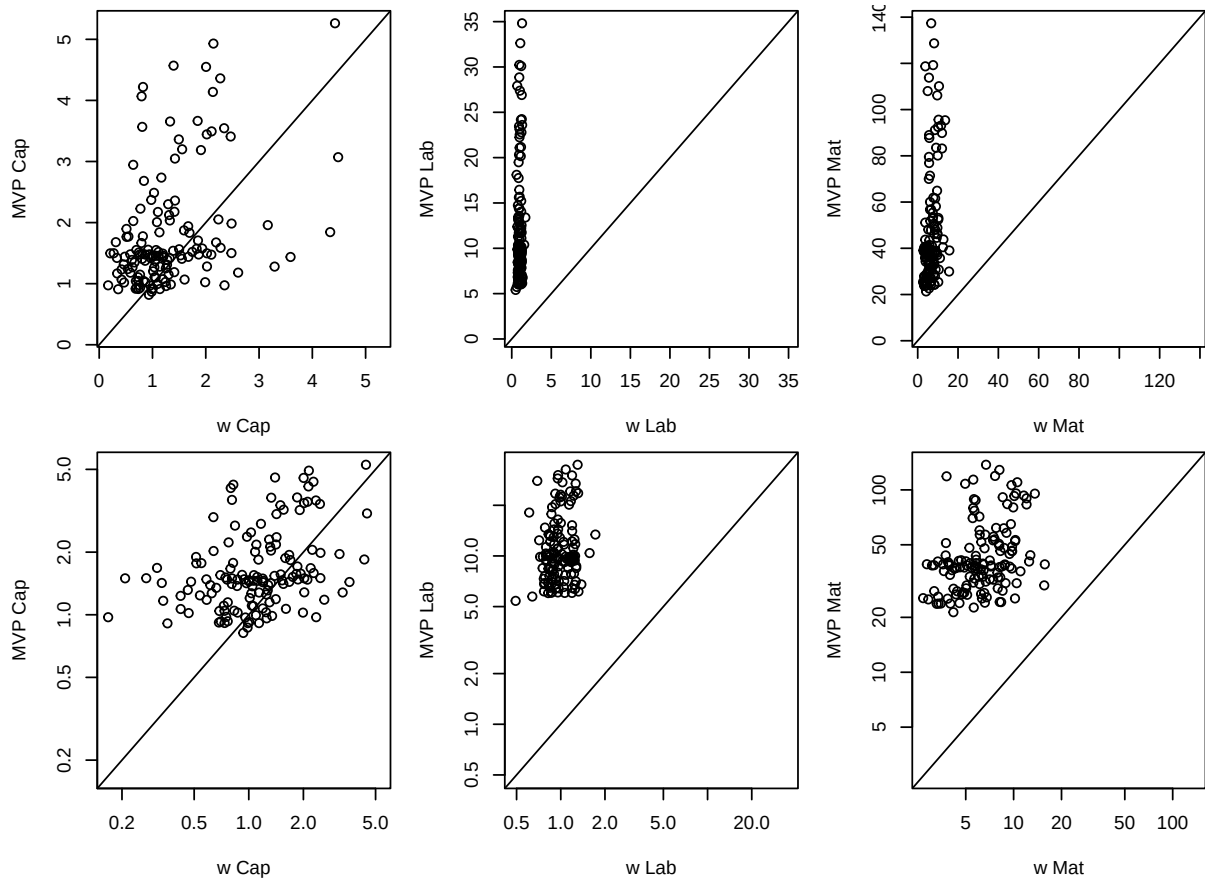


Figure 2.13: Marginal value products and corresponding input prices

profit by using more labor and more materials. Given that most firms operate under increasing returns to scale, it is not surprising that most firms would gain from increasing most—or even all—input quantities. Therefore, the question arises why the firms in the sample did not do this. There are many possible reasons for not increasing the input quantities until the predicted optimal input levels, e.g. legal restrictions, environmental regulations, market imperfections, credit (liquidity) constraints, and/or risk aversion. Furthermore, market imperfections might cause that the (observed) average prices are lower than the marginal costs of obtaining these inputs (e.g. [Henning and Henningsen, 2007](#)), particularly for labor and capital.

2.3.11 First-order conditions for cost minimization

As the marginal rates of technical substitution are constant for linear production functions, we compare the input price ratios with the negative inverse marginal rates of technical substitution by creating a histogram for each input price ratio and drawing a vertical line at the corresponding negative marginal rate of technical substitution:

```
> hist( dat$pCap / dat$pLab )
> abline( v = - mrtsLabCap, lwd = 3 )
> hist( dat$pCap / dat$pMat )
> abline( v = - mrtsMatCap, lwd = 3 )
> hist( dat$pLab / dat$pMat )
> abline( v = - mrtsMatLab, lwd = 3 )
> hist( dat$pLab / dat$pCap )
> abline( v = - mrtsCapLab, lwd = 3 )
> hist( dat$pMat / dat$pCap )
> abline( v = - mrtsCapMat, lwd = 3 )
> hist( dat$pMat / dat$pLab )
> abline( v = - mrtsLabMat, lwd = 3 )
```

The resulting graphs are shown in figure 2.14. The upper left graph shows that the ratio between the capital price and the labor price is larger than the absolute value of the marginal rate of technical substitution between labor and capital (0.151) for the most firms in the sample:

$$\frac{w_{cap}}{w_{lab}} > -MRTS_{lab,cap} = \frac{MP_{cap}}{MP_{lab}} \quad (2.74)$$

Or taken the other way round, the lower left graph shows that the ratio between the labor price and the capital price is smaller than the absolute value of the marginal rate of technical substitution between capital and labor (6.616) for the most firms in the sample:

$$\frac{w_{lab}}{w_{cap}} < -MRTS_{cap,lab} = \frac{MP_{lab}}{MP_{cap}} \quad (2.75)$$

Hence, the firm can get closer to the minimum of the costs by substituting labor for capital,

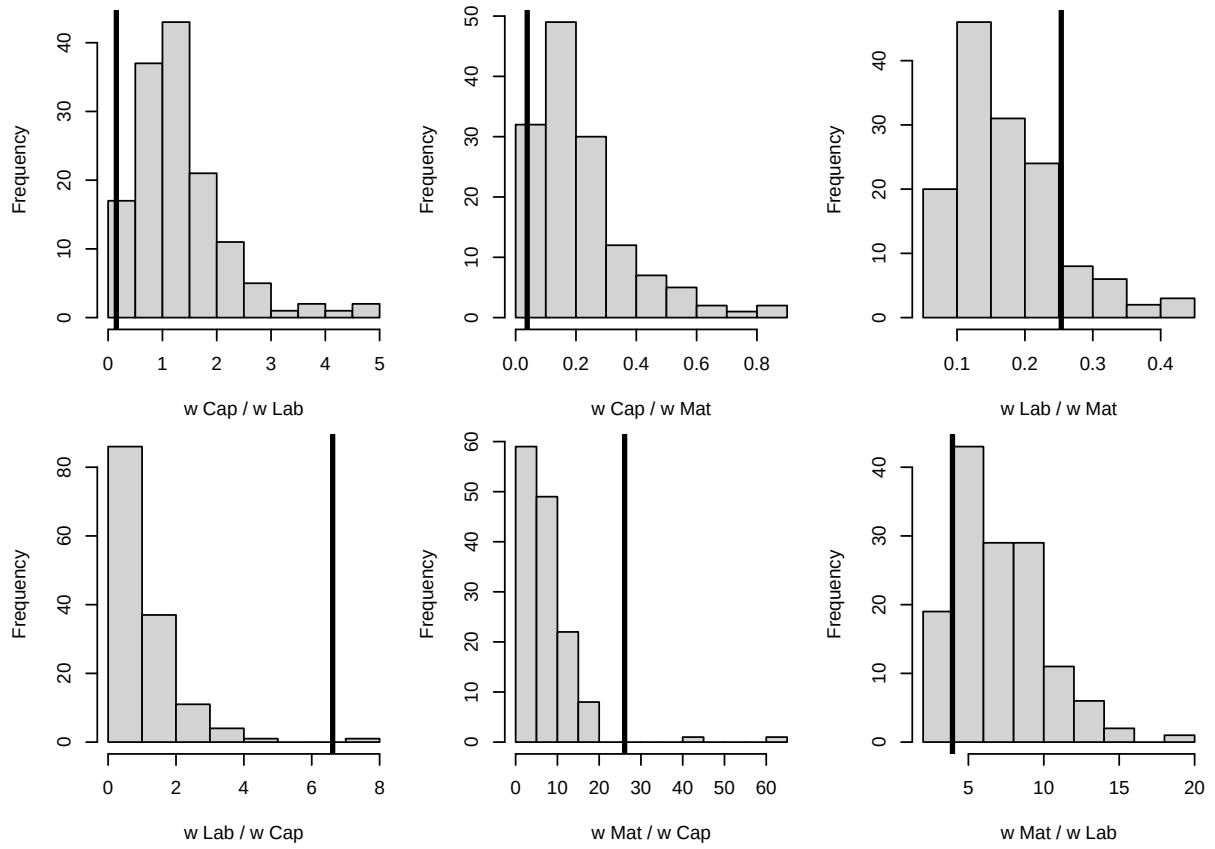


Figure 2.14: First-order conditions for costs minimization

because this will decrease the marginal product of labor and increase the marginal product of capital so that the absolute value of the MRTS between labor and capital increases, the absolute value of the MRTS between capital and labor decreases, and both of the MRTS get closer to the corresponding input price ratios. Similarly, the graphs in the middle column indicate that almost all firms should substitute materials for capital and the graphs on the right indicate that most of the firms should substitute labor for materials. Hence, the firms could reduce production costs particularly by using less capital and more labor.

2.3.12 Derived input demand functions and output supply functions

Given a linear production function (2.70), the input quantities chosen by a profit maximizing producer are either zero, indeterminate, or infinity:

$$x_i(p, w) = \begin{cases} 0 & \text{if } MVP_i < w_i \\ \text{indeterminate} & \text{if } MVP_i = w_i \\ \infty & \text{if } MVP_i > w_i \end{cases} \quad (2.76)$$

If all input quantities are zero, the output quantity is equal to the intercept, which is zero in case of weak essentiality. Otherwise, the output quantity is indeterminate or infinity:

$$y(p, w) = \begin{cases} \beta_0 & \text{if } MVP_i < w_i \forall i \\ \infty & \text{if } MVP_i > w_i \exists i \\ \text{indeterminate} & \text{otherwise} \end{cases} \quad (2.77)$$

A cost minimizing producer will use only a single input, i.e. the input with the lowest cost per unit of produced output (w_i/MVP_i). If the lowest cost per unit of produced output can be obtained by two or more inputs, these input quantities are indeterminate.

$$x_i(w, y) = \begin{cases} 0 & \text{if } \frac{\beta_i}{w_i} < \frac{\beta_j}{w_j} \exists j \\ \frac{y - \beta_0}{\beta_i} & \text{if } \frac{\beta_i}{w_i} > \frac{\beta_j}{w_j} \forall j \neq i \\ \text{indeterminate} & \text{otherwise} \end{cases} \quad (2.78)$$

Given that the unconditional and conditional input demand functions and the output supply functions based on the linear production function are non-continuous and often return either zero or infinite values, it does not make much sense to use this functional form to predict the effects of price changes when the true technology implies that firms always use non-zero finite input quantities.

2.4 Cobb-Douglas production function

2.4.1 Specification

A Cobb-Douglas production function with N inputs is defined as:

$$y = A \prod_{i=1}^N x_i^{\alpha_i}. \quad (2.79)$$

This function can be linearized by taking the (natural) logarithm on both sides:

$$\ln y = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i, \quad (2.80)$$

where α_0 is equal to $\ln A$.

2.4.2 Estimation

We can estimate this Cobb-Douglas production function for our data set using the command `lm`:

```
> prodCD <- lm( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ),
+   data = dat )
> summary( prodCD )
```

Call:

```
lm(formula = log(qOut) ~ log(qCap) + log(qLab) + log(qMat), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.67239	-0.28024	0.00667	0.47834	1.30115

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-2.06377	1.31259	-1.572	0.1182
log(qCap)	0.16303	0.08721	1.869	0.0637 .
log(qLab)	0.67622	0.15430	4.383	2.33e-05 ***
log(qMat)	0.62720	0.12587	4.983	1.87e-06 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.656 on 136 degrees of freedom

Multiple R-squared: 0.5943, Adjusted R-squared: 0.5854

F-statistic: 66.41 on 3 and 136 DF, p-value: < 2.2e-16

2.4.3 Properties

The monotonicity condition is (globally) fulfilled, as the estimated coefficients of all three (logarithmic) input quantities are positive and the output quantity as well as all input quantities are non-negative (see equation 2.82). However, the coefficient of the (logarithmic) capital quantity is only statistically significantly different from zero at the 10% level. Therefore, we cannot be sure that the capital quantity has a positive effect on the output quantity.²

The quasiconcavity of our estimated Cobb-Douglas production function is checked in section 2.4.12.

The production technology described by a Cobb-Douglas production function always shows weak and strict essentiality, because the output quantity becomes zero, as soon as a single input quantity becomes zero (see equation 2.79).

The input requirement sets derived from Cobb-Douglas production functions are always closed and non-empty for $y > 0$ if strict monotonicity is fulfilled for at least one input ($\exists i \in \{1, \dots, N\} : \beta_i > 0$), as the input quantities must be non-negative ($x_i \geq 0 \forall i$).

The Cobb-Douglas production function always returns finite, real, and single values if the input quantities are non-negative and finite. The predicted output quantity is non-negative as long as A and the input quantities are non-negative, where $A = \exp(\alpha_0)$ is positive even if α_0 is negative.

All Cobb-Douglas production functions are continuous and twice-continuously differentiable.

2.4.4 Predicted output quantities

We can calculate the predicted (“fitted”) output quantities manually by taking the Cobb-Douglas function (2.79), the observed input quantities, and the estimated parameters, but it is easier to use the `fitted` method to obtain the predicted values of the dependent variable from an estimated model. As we estimated the Cobb-Douglas function in logarithms, we have to use the exponential function to obtain the predicted values in levels (non-logarithms):

```
> dat$qOutCD <- exp( fitted( prodCD ) )
> all.equal( dat$qOutCD,
+   with( dat, exp( coef( prodCD )[ "(Intercept)" ] ) *
+     qCap^coef( prodCD )[ "log(qCap)" ] *
+     qLab^coef( prodCD )[ "log(qLab)" ] *
+     qMat^coef( prodCD )[ "log(qMat)" ] ) )

[1] TRUE
```

We can evaluate the “fit” of the Cobb-Douglas production function by comparing the observed with the fitted output quantities:

²In section 2.3.3, we discuss whether input quantities that do not have a statistically significant effect on the output should be removed from the model.

```
> compPlot( dat$qOut, dat$qOutCD )
> compPlot( dat$qOut, dat$qOutCD, log = "xy" )
```

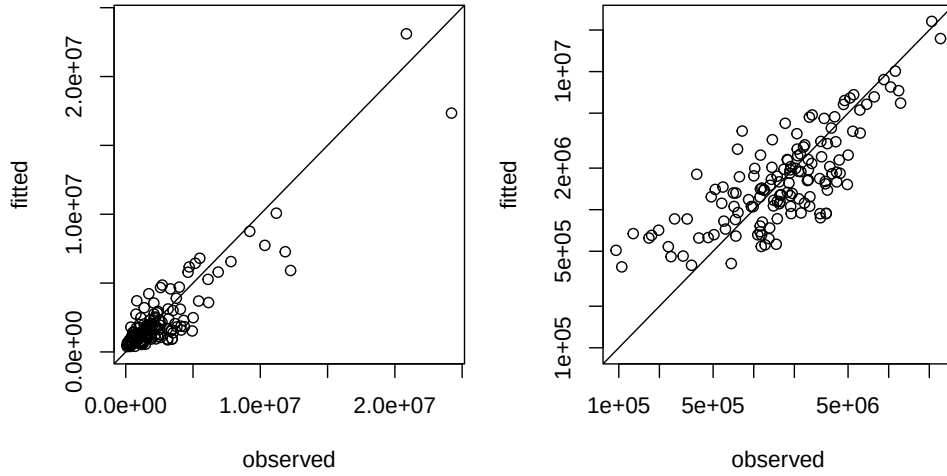


Figure 2.15: Cobb-Douglas production function: fit of the model

The resulting graphs are shown in figure 2.15. While the graph in the left panel uses a linear scale for the axes, the graph in the right panel uses a logarithmic scale for both axes. Hence, the deviations from the 45°-line illustrate the *absolute* deviations in the left panel and the *relative* deviations in the right panel. The fit of the model looks okay in the scatter plot on the left-hand side, but if we use a logarithmic scale on both axes (as in the graph on the right-hand side), we can see that the output quantity is generally over-estimated if the observed output quantity is small.

2.4.5 Output elasticities

In the Cobb-Douglas function, the output elasticities of the inputs are equal to the corresponding coefficients:

$$\epsilon_i = \frac{\partial y}{\partial x_i} \frac{x_i}{y} = \frac{\partial \ln y}{\partial \ln x_i} \frac{\partial y}{\partial \ln y} \frac{\partial \ln x_i}{\partial x_i} \frac{x_i}{y} = \frac{\partial \ln y}{\partial \ln x_i} \frac{1}{1} \frac{x_i}{x_i} \frac{1}{y} = \frac{\partial \ln y}{\partial \ln x_i} = \alpha_i \quad (2.81)$$

Hence, if a firm increases capital input by one percent, the output will increase by 0.16 percent; if a firm increases labor input by one percent, the output will increase by 0.68 percent; and if a firm increases materials input by one percent, the output will increase by 0.63 percent. The output elasticity of capital is somewhat larger and the output elasticity of labor is considerably smaller when estimated by a Cobb-Douglas production function than when estimated by a linear production function. Indeed, the output elasticities of all three inputs are in the reasonable range, i.e. between zero one one, now.

2.4.6 Marginal products

The marginal products of the inputs in a Cobb-Douglas production function can be derived in different ways. One possibility is to derive the marginal products from equation (2.79):

$$MP_i = \frac{\partial y}{\partial x_i} = \alpha_i x_i^{-1} A \prod_{j=1}^N x_j^{\alpha_j} = \alpha_i \frac{y}{x_i} \quad (2.82)$$

Another possibility is to derive the marginal products by re-arranging equation (2.81):

$$MP_i = \frac{\partial y}{\partial x_i} = \frac{\partial \ln y}{\partial \ln x_i} \frac{y}{x_i} = \alpha_i \frac{y}{x_i} \quad (2.83)$$

As the marginal products depend on the input and output quantities and these quantities generally differ between firms, the marginal products based on Cobb-Douglas production functions also differ between firms. Hence, we can calculate them for each firm in the sample:

```
> dat$mpCapCD <- coef(prodCD)["log(qCap)"] * dat$qOut / dat$qCap
> dat$mpLabCD <- coef(prodCD)["log(qLab)"] * dat$qOut / dat$qLab
> dat$mpMatCD <- coef(prodCD)["log(qMat)"] * dat$qOut / dat$qMat
```

While microeconomic theory usually ignores statistical “noise,” empirical estimations have a noise term so that the observed output quantities are generally unequal to the output quantities predicted by the estimated production function. Hence, alternatively, we can calculate the marginal products based on the predicted output quantities:

```
> dat$mpCapCDFit <- coef(prodCD)["log(qCap)"] * dat$qOutCD / dat$qCap
> dat$mpLabCDFit <- coef(prodCD)["log(qLab)"] * dat$qOutCD / dat$qLab
> dat$mpMatCDFit <- coef(prodCD)["log(qMat)"] * dat$qOutCD / dat$qMat
```

We can obtain summary statistics of all calculated marginal products by:

```
> summary( dat[, c( "mpCapCD", "mpCapCDFit", "mpLabCD", "mpLabCDFit",
+   "mpMatCD", "mpMatCDFit" ) ] )
```

mpCapCD	mpCapCDFit	mpLabCD	mpLabCDFit
Min. : 0.2358	Min. : 0.8782	Min. : 0.581	Min. : 1.870
1st Qu.: 2.4923	1st Qu.: 2.5344	1st Qu.: 3.465	1st Qu.: 4.250
Median : 3.6158	Median : 3.4891	Median : 6.220	Median : 5.397
Mean : 5.3207	Mean : 4.4904	Mean : 6.905	Mean : 5.909
3rd Qu.: 6.7871	3rd Qu.: 5.4819	3rd Qu.: 9.738	3rd Qu.: 7.114
Max. :24.9217	Max. :14.8856	Max. :17.329	Max. :13.757
mpMatCD	mpMatCDFit		
Min. : 5.157	Min. : 21.89		
1st Qu.: 28.750	1st Qu.: 35.49		

Median : 49.850	Median : 45.16
Mean : 56.851	Mean : 47.45
3rd Qu.: 70.450	3rd Qu.: 56.60
Max. :189.056	Max. :110.13

Hence, if a firm increases capital input by one unit, the output will on average increase by around 5.32 or 4.49 units. If the firm increases labor input by one unit, the output will on average increase by around 6.91 or 5.91 units. And if the firm increases materials input by one unit, the output will on average increase by around 56.85 or 47.45 units.

If the “noise” term mainly comprises measurement errors in the output quantity, the “true” output quantity is probably closer to the predicted output quantity than to the observed output quantity so that it seems to be more appropriate to calculate the marginal products based on the predicted output quantity than based on the observed output quantity. On the other hand, if the “noise” term mainly comprises the effects of omitted variables, the “true” output quantity is probably closer to the observed output quantity than to the predicted output quantity so that it seems to be more appropriate to calculate the marginal products based on the observed output quantity than based on the predicted output quantity.

However, in the second case, the output quantities used in the calculations are generally unequal to those predicted by the estimated production function, while the derivation of the marginal products in equation (2.82) assumes that the output quantities y are equal to the right-hand side of the production function (2.79). Therefore, using observed (rather than predicted) output quantities can result in inconsistencies in microeconomic analyses.

We can visualize (the variation of) these marginal products with histograms:

```
> hist( dat$mpCapCD )
> hist( dat$mpLabCD )
> hist( dat$mpMatCD )
> hist( dat$mpCapCDFit )
> hist( dat$mpLabCDFit )
> hist( dat$mpMatCDFit )
```

The resulting graphs are shown in figure 2.16. If the firms increase capital input by one unit, the output of most firms will increase by between 0 and 8 units; if the firms increase labor input by one unit, the output of most firms will increase by between 2 and 12 units; and if the firms increase materials input by one unit, the output of most firms will increase by between 20 and 80 units. Not surprisingly, a comparison of these marginal effects with the marginal effects from the linear production function confirms the results from the comparison based on the output elasticities: the marginal products of capital are generally larger than the marginal product estimated by the linear production function and the marginal products of labor are generally smaller than the marginal product estimated by the linear production function, while the marginal products of

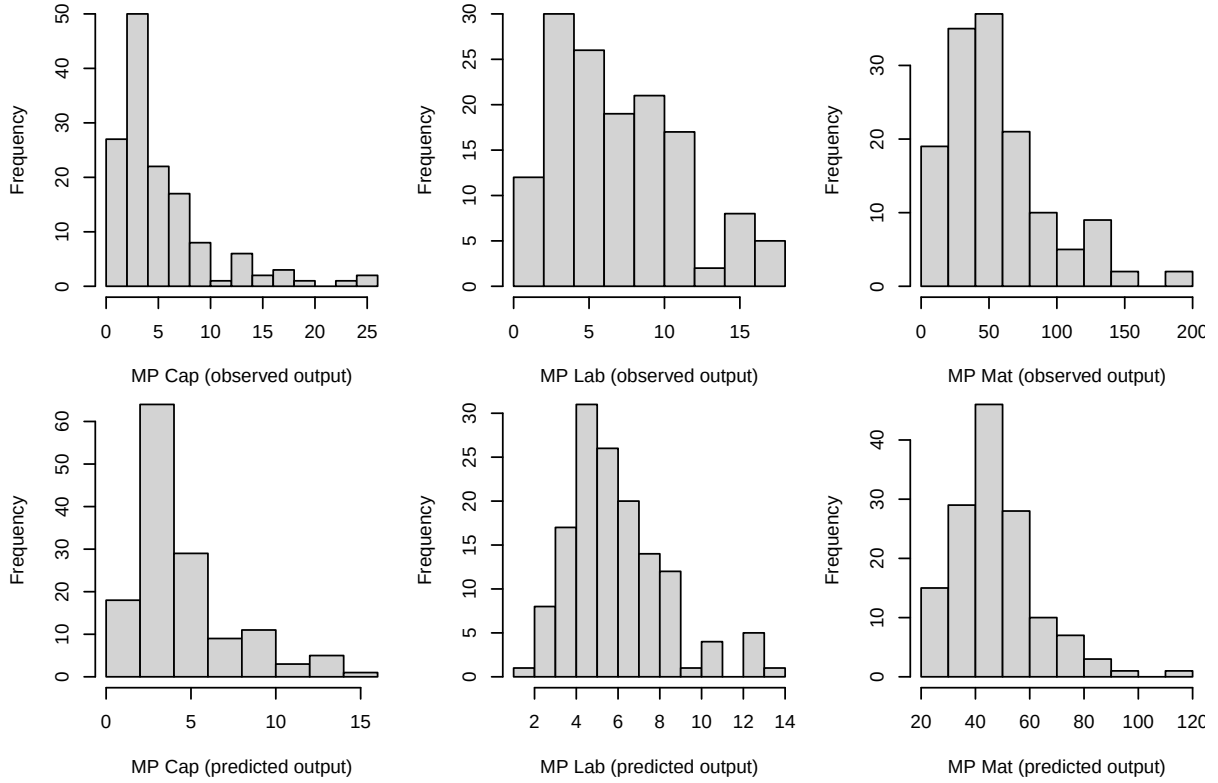


Figure 2.16: Cobb-Douglas production function: marginal products

fuel are (on average) rather similar to the marginal product estimated by the linear production function.

2.4.7 Elasticity of scale

As the elasticity of scale is the sum of all output elasticities (see equation 2.73), we can calculate it simply by summing up all coefficients except for the intercept:

```
> sum( coef( prodCD )[ -1 ] )
[1] 1.466442
```

Hence, if the firm increases all input quantities by one percent, output will increase by 1.47 percent. This means that the technology has strong increasing returns to scale. However, in contrast to the results of the linear production function, the elasticity of scale based on the Cobb-Douglas production function is (globally) constant. Hence, it does not decrease (or increase), e.g., with the size of the firm. This means that the optimal firm size would be infinity.

We can use the delta method (see section 1.5.7) to calculate the variance and the standard error of the elasticity of scale. Given that the first derivatives of the elasticity of scale with respect to the estimated coefficients are $\partial\varepsilon/\partial\alpha_0 = 0$ and $\partial\varepsilon/\partial\alpha_{Cap} = \partial\varepsilon/\partial\alpha_{Lab} = \partial\varepsilon/\partial\alpha_{Mat} = 1$, we can do this by following commands:

2 Primal Approach: Production Function

```
> ESCD <- sum( coef(prodCD)[-1] )  
[1] 1.466442  
  
> dESCD <- c( 0, 1, 1, 1 )  
[1] 0 1 1 1  
  
> varESCD <- t(dESCD) %*% vcov(prodCD) %*% dESCD  
[,1]  
[1,] 0.0118237  
  
> seESCD <- sqrt( varESCD )  
[,1]  
[1,] 0.1087369
```

Now, we can apply a t test to test whether the elasticity of scale significantly differs from one. The following commands calculate the t value and the critical value for a two-sided t test based on a 5% significance level:

```
> tESCD <- (ESCD - 1) / seESCD  
[,1]  
[1,] 4.289645  
  
> cvESCD <- qt( 0.975, 136 )  
[1] 1.977561
```

Given that the t value is larger than the critical value, we can reject the null hypothesis of constant returns to scale and conclude that the technology has significantly increasing returns to scale. The P value for this two-sided t test is:

```
> pESCD <- 2 * ( 1 - pt( tESCD, 136 ) )  
[,1]  
[1,] 3.372264e-05
```

Given that the P value is close to zero, we can be very sure that the technology has increasing returns to scale. The 95% confidence interval for the elasticity of scale is:

```
> c( ESCD - cvESCD * seESCD, ESCD + cvESCD * seESCD )  
[1] 1.251409 1.681476
```


2.4.8 Marginal rates of technical substitution

The MRTS based on the Cobb-Douglas production function differ between firms. They can be calculated as follows:

```
> dat$mrtsCapLabCD <- - dat$mpLabCD / dat$mpCapCD
> dat$mrtsLabCapCD <- - dat$mpCapCD / dat$mpLabCD
> dat$mrtsCapMatCD <- - dat$mpMatCD / dat$mpCapCD
> dat$mrtsMatCapCD <- - dat$mpCapCD / dat$mpMatCD
> dat$mrtsLabMatCD <- - dat$mpMatCD / dat$mpLabCD
> dat$mrtsMatLabCD <- - dat$mpLabCD / dat$mpMatCD
```

Given that the marginal rates of technical substitution are ratios between marginal products, the output quantities in the marginal products cancel out so that it does not matter whether one uses the marginal products that were calculated based on the observed output quantities or the marginal products that were calculated based on the predicted output quantities:

```
> all.equal( dat$mrtsCapLabCD, - dat$mpLabCDFit / dat$mpCapCDFit )
[1] TRUE
> all.equal( dat$mrtsLabCapCD, - dat$mpCapCDFit / dat$mpLabCDFit )
[1] TRUE
> all.equal( dat$mrtsCapMatCD, - dat$mpMatCDFit / dat$mpCapCDFit )
[1] TRUE
> all.equal( dat$mrtsMatCapCD, - dat$mpCapCDFit / dat$mpMatCDFit )
[1] TRUE
> all.equal( dat$mrtsLabMatCD, - dat$mpMatCDFit / dat$mpLabCDFit )
[1] TRUE
> all.equal( dat$mrtsMatLabCD, - dat$mpLabCDFit / dat$mpMatCDFit )
[1] TRUE
```

We can visualize (the variation of) these MRTSs with histograms:

```
> hist( dat$mrtsCapLabCD )
> hist( dat$mrtsLabCapCD )
> hist( dat$mrtsCapMatCD )
> hist( dat$mrtsMatCapCD )
> hist( dat$mrtsLabMatCD )
> hist( dat$mrtsMatLabCD )
```

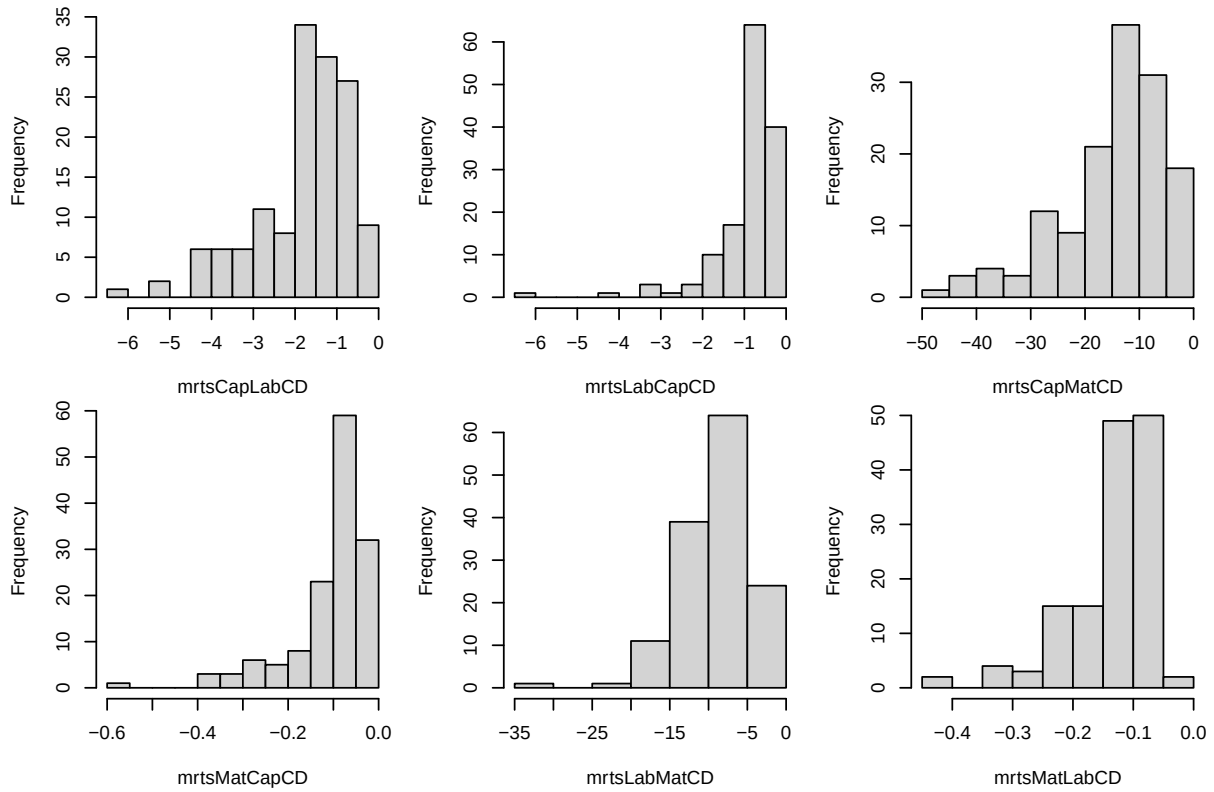


Figure 2.17: Cobb-Douglas production function: marginal rates of technical substitution (MRTS)

The resulting graphs are shown in figure 2.17. According to the MRTS based on the Cobb-Douglas production function, most firms only need between 0.5 and 2 additional units of capital or between 0.05 and 0.15 additional units of materials to replace one unit of labor.

2.4.9 Relative marginal rates of technical substitution

As we do not know the units of measurements of the input quantities, the interpretation of the MRTSs is practically not very useful. To overcome this problem, we calculate the relative marginal rates of technical substitution (RMRTS) by equation (2.18). As the output elasticities based on a Cobb-Douglas production function are equal to the coefficients, we can calculate the RMRTS as follows:

```
> rmrtsCapLabCD <- - coef(prodCD)["log(qLab)"] / coef(prodCD)["log(qCap)"]
```

```
log(qLab)
```

```
-4.147897
```

```
> rmrtsLabCapCD <- - coef(prodCD)["log(qCap)"] / coef(prodCD)["log(qLab)"]
```

```
log(qCap)
```

```
-0.241086
```

```
> rmrtsCapMatCD <- - coef(prodCD)["log(qMat)"] / coef(prodCD)["log(qCap)"]

log(qMat)
-3.847203

> rmrtsMatCapCD <- - coef(prodCD)["log(qCap)"] / coef(prodCD)["log(qMat)"]

log(qCap)
-0.2599291

> rmrtsLabMatCD <- - coef(prodCD)["log(qMat)"] / coef(prodCD)["log(qLab)"]

log(qMat)
-0.9275069

> rmrtsMatLabCD <- - coef(prodCD)["log(qLab)"] / coef(prodCD)["log(qMat)"]

log(qLab)
-1.078159
```

Hence, if a firm wants to reduce the use of labor by one percent, it has to use 4.15 percent more capital in order to produce the same output as before. Alternatively, the firm can replace one percent of labor by using 1.08 percent more materials. If the firm increases the use of labor by one percent, it can reduce capital by 4.15 percent whilst still producing the same output as before. Alternatively, the firm can reduce materials by 1.08 percent.

2.4.10 First and second partial derivatives

For the Cobb-Douglas production function with three inputs (2.79), the first derivatives (marginal products) are:

$$f_1 = \frac{\partial y}{\partial x_1} = \alpha_1 A x_1^{\alpha_1-1} x_2^{\alpha_2} x_3^{\alpha_3} = \alpha_1 \frac{y}{x_1} \quad (2.84)$$

$$f_2 = \frac{\partial y}{\partial x_2} = \alpha_2 A x_1^{\alpha_1} x_2^{\alpha_2-1} x_3^{\alpha_3} = \alpha_2 \frac{y}{x_2} \quad (2.85)$$

$$f_3 = \frac{\partial y}{\partial x_3} = \alpha_3 A x_1^{\alpha_1} x_2^{\alpha_2} x_3^{\alpha_3-1} = \alpha_3 \frac{y}{x_3} \quad (2.86)$$

and the second derivatives are:

$$f_{11} = \frac{\partial f_1}{\partial x_1} = \alpha_1 \frac{f_1}{x_1} - \alpha_1 \frac{y}{x_1^2} = \frac{f_1^2}{y} - \frac{f_1}{x_1} \quad (2.87)$$

$$f_{22} = \frac{\partial f_2}{\partial x_2} = \alpha_2 \frac{f_2}{x_2} - \alpha_2 \frac{y}{x_2^2} = \frac{f_2^2}{y} - \frac{f_2}{x_2} \quad (2.88)$$

$$f_{33} = \frac{\partial f_3}{\partial x_3} = \alpha_3 \frac{f_3}{x_3} - \alpha_3 \frac{y}{x_3^2} = \frac{f_3^2}{y} - \frac{f_3}{x_3} \quad (2.89)$$

2 Primal Approach: Production Function

$$f_{12} = \frac{\partial f_1}{\partial x_2} = \alpha_1 \frac{f_2}{x_1} = \frac{f_1 f_2}{y} \quad (2.90)$$

$$f_{13} = \frac{\partial f_1}{\partial x_3} = \alpha_1 \frac{f_3}{x_1} = \frac{f_1 f_3}{y} \quad (2.91)$$

$$f_{23} = \frac{\partial f_2}{\partial x_3} = \alpha_2 \frac{f_3}{x_2} = \frac{f_2 f_3}{y}. \quad (2.92)$$

Generally, for an N -input Cobb-Douglas function, the first and second derivatives are

$$f_i = \alpha_i \frac{y}{x_i} \quad (2.93)$$

$$f_{ij} = \frac{f_i f_j}{y} - \delta_{ij} \frac{f_i}{x_i}, \quad (2.94)$$

where δ_{ij} denotes Kronecker's delta with

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (2.95)$$

In section 2.4.6, we have already computed the first partial derivatives (f_i) of the Cobb-Douglas production function, which are also called marginal products. In section 2.4.6, we also argue that using observed (rather than predicted) output quantities can result in inconsistencies in microeconomic analyses. Therefore, we use marginal products based on predicted output quantities in the following derivations. In order to simplify further computations, we create variables with shorter names for the first derivatives (marginal products):

```
> dat$fCap <- dat$mpCapCDFit
> dat$fLab <- dat$mpLabCDFit
> dat$fMat <- dat$mpMatCDFit
```

Based on these first derivatives, we can also calculate the second derivatives:

```
> dat$fCapCap <- with( dat, fCap^2 / qOutCD - fCap / qCap )
> dat$fLabLab <- with( dat, fLab^2 / qOutCD - fLab / qLab )
> dat$fMatMat <- with( dat, fMat^2 / qOutCD - fMat / qMat )
> dat$fCapLab <- with( dat, fCap * fLab / qOutCD )
> dat$fCapMat <- with( dat, fCap * fMat / qOutCD )
> dat$fLabMat <- with( dat, fLab * fMat / qOutCD )
```

2.4.11 Elasticities of substitution

2.4.11.1 Direct elasticities of substitution

We can calculate the direct elasticities of substitution using equation (2.20):

```

> dat$esdCapLab <- with( dat,
+   - ( fCap * fLab * ( qCap * fCap + qLab * fLab ) ) /
+   ( qCap * qLab *
+     ( fCapCap * fLab^2 - 2 * fCapLab * fCap * fLab + fLabLab * fCap^2 ) ) )
> dat$esdCapMat <- with( dat,
+   - ( fCap * fMat * ( qCap * fCap + qMat * fMat ) ) /
+   ( qCap * qMat *
+     ( fCapCap * fMat^2 - 2 * fCapMat * fCap * fMat + fMatMat * fCap^2 ) ) )
> dat$esdLabMat <- with( dat,
+   - ( fLab * fMat * ( qLab * fLab + qMat * fMat ) ) /
+   ( qLab * qMat *
+     ( fLabLab * fMat^2 - 2 * fLabMat * fLab * fMat + fMatMat * fLab^2 ) ) )
> range( dat$esdCapLab )

[1] 1 1

> range( dat$esdCapMat )

[1] 1 1

> range( dat$esdLabMat )

[1] 1 1

```

All direct elasticities of substitution are exactly one for all observations and for all combinations of the inputs. This is no surprise and confirms that our calculations have been done correctly, because the Cobb-Douglas functional form implies that the direct elasticities of substitution are always equal to one, irrespective of the input and output quantities and the estimated parameters.

2.4.11.2 Allen elasticities of substitution

In order to calculate the Allen elasticities of substitution, we need to construct the bordered Hessian matrix. As the first and second derivatives of the Cobb-Douglas function differ between observations, also the bordered Hessian matrix differs between observations. As a starting point, we construct the bordered Hessian Matrix just for the first observation:

```

> bhm <- matrix( 0, nrow = 4, ncol = 4 )
> bhm[ 1, 2 ] <- bhm[ 2, 1 ] <- dat$fCap[ 1 ]
> bhm[ 1, 3 ] <- bhm[ 3, 1 ] <- dat$fLab[ 1 ]
> bhm[ 1, 4 ] <- bhm[ 4, 1 ] <- dat$fMat[ 1 ]
> bhm[ 2, 2 ] <- dat$fCapCap[ 1 ]
> bhm[ 3, 3 ] <- dat$fLabLab[ 1 ]
> bhm[ 4, 4 ] <- dat$fMatMat[ 1 ]

```

2 Primal Approach: Production Function

```
> bhm[ 2, 3 ] <- bhm[ 3, 2 ] <- dat$fCapLab[ 1 ]
> bhm[ 2, 4 ] <- bhm[ 4, 2 ] <- dat$fCapMat[ 1 ]
> bhm[ 3, 4 ] <- bhm[ 4, 3 ] <- dat$fLabMat[ 1 ]
> print(bhm)
```

```
      [,1]      [,2]      [,3]      [,4]
[1,] 0.000000 6.229014e+00 6.031225e+00 59.0909133861
[2,] 6.229014 -6.202845e-05 1.169835e-05 0.0001146146
[3,] 6.031225 1.169835e-05 -5.423455e-06 0.0001109752
[4,] 59.090913 1.146146e-04 1.109752e-04 -0.0006462733
```

Based on this bordered Hessian matrix, we can calculate the co-factors F_{ij} :

```
> FCapLab <- - det( bhm[ -2, -3 ] )
```

```
[1] -0.06512713
```

```
> FCapMat <- det( bhm[ -2, -4 ] )
```

```
[1] -0.006165438
```

```
> FLabMat <- - det( bhm[ -3, -4 ] )
```

```
[1] -0.02641227
```

Now, we can use equation (2.21) to calculate the Allen elasticities of substitution:

```
> esaCapLab <- with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) / ( qCap * qLab ) ) *
+   FCapLab / det( bhm )
```

```
[1] 1
```

```
> esaCapMat <- with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) / ( qCap * qMat ) ) *
+   FCapMat / det( bhm )
```

```
[1] 1
```

```
> esaLabMat <- with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) / ( qLab * qMat ) ) *
+   FLabMat / det( bhm )
```

```
[1] 1
```

All elasticities of substitution are exactly one at the first observation.

We can check whether condition (2.24) is fulfilled:

```
> FCapCap <- det( bhm[ -2, -2 ] )
> FLabLab <- det( bhm[ -3, -3 ] )
> FMatMat <- det( bhm[ -4, -4 ] )
> esaCapCap <- with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) / ( qCap * qCap ) ) *
+   FCapCap / det( bhm )
> esaLabLab <- with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) / ( qLab * qLab ) ) *
+   FLabLab / det( bhm )
> esaMatMat <- with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) / ( qMat * qMat ) ) *
+   FMatMat / det( bhm )
> k1 <- dat$qCap[ 1 ] * bhm[ 1, 2 ] / with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) )
> k2 <- dat$qLab[ 1 ] * bhm[ 1, 3 ] / with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) )
> k3 <- dat$qMat[ 1 ] * bhm[ 1, 4 ] / with( dat[1,],
+   ( qCap * fCap + qLab * fLab + qMat * fMat ) )
> k1 * esaCapCap + k2 * esaCapLab + k3 * esaCapMat

[1] -8.326673e-16

> k1 * esaCapLab + k2 * esaLabLab + k3 * esaLabMat

[1] 5.551115e-17

> k1 * esaCapMat + k2 * esaLabMat + k3 * esaMatMat

[1] 3.330669e-16
```

As all three values are virtually zero (the small deviations from zero are caused by rounding errors), we can conclude that condition (2.24) is fulfilled for all three inputs.

In order to calculate the Allen elasticities of substitution for all firms, we need to construct the bordered Hessian matrices for all observations. We can stack the bordered Hessian matrices on top of each other, which results in a three-dimensional array:

```
> bhmArray <- array( 0, c( 4, 4, nrow( dat ) ) )
> bhmArray[ 1, 2, ] <- bhmArray[ 2, 1, ] <- dat$fCap
> bhmArray[ 1, 3, ] <- bhmArray[ 3, 1, ] <- dat$fLab
```

```
> bhmArray[ 1, 4, ] <- bhmArray[ 4, 1, ] <- dat$fMat
> bhmArray[ 2, 2, ] <- dat$fCapCap
> bhmArray[ 3, 3, ] <- dat$fLabLab
> bhmArray[ 4, 4, ] <- dat$fMatMat
> bhmArray[ 2, 3, ] <- bhmArray[ 3, 2, ] <- dat$fCapLab
> bhmArray[ 2, 4, ] <- bhmArray[ 4, 2, ] <- dat$fCapMat
> bhmArray[ 3, 4, ] <- bhmArray[ 4, 3, ] <- dat$fLabMat
```

Based on these bordered Hessian matrices, we can calculate the co-factors and the determinants of the bordered Hessian matrices for each observation:

```
> dat$FCapLab <- - apply( bhmArray[ -2, -3, ], 3, det )
> dat$FCapMat <- apply( bhmArray[ -2, -4, ], 3, det )
> dat$FLabMat <- - apply( bhmArray[ -3, -4, ], 3, det )
> dat$bhmDet <- apply( bhmArray, 3, det )
```

Finally, we can calculate the Allen elasticities of substitution:

```
> dat$esaCapLab <- with( dat,
+      ( qCap * fCap + qLab * fLab + qMat * fMat ) /
+      ( qCap * qLab ) * FCapLab / bhmDet )
> dat$esaCapMat <- with( dat,
+      ( qCap * fCap + qLab * fLab + qMat * fMat ) /
+      ( qCap * qMat ) * FCapMat / bhmDet )
> dat$esaLabMat <- with( dat,
+      ( qCap * fCap + qLab * fLab + qMat * fMat ) /
+      ( qLab * qMat ) * FLabMat / bhmDet )
> range( dat$esaCapLab )
```

```
[1] 1 1
```

```
> range( dat$esaCapMat )
```

```
[1] 1 1
```

```
> range( dat$esaLabMat )
```

```
[1] 1 1
```

Also the Allen elasticities of substitution are equal to one for all firms and for all input combinations. This is again no surprise and confirms that our calculations have been done correctly, because the Cobb-Douglas functional form implies that the Allen elasticities of substitution are always equal to one, irrespective of the input and output quantities and the estimated parameters.

2.4.11.3 Morishima elasticities of substitution

In order to calculate the Morishima elasticities of substitution, we need to calculate the co-factors of the diagonal elements of the bordered Hessian matrix for each observation:

```
> dat$FCapCap <- apply( bhmArray[ -2, -2, ], 3, det )
> dat$FLabLab <- apply( bhmArray[ -3, -3, ], 3, det )
> dat$FMatMat <- apply( bhmArray[ -4, -4, ], 3, det )
```

We use equation (2.25) to calculate the Morishima elasticities of substitution for all observations in our data set:

```
> dat$esmCapLab <- with( dat, ( fLab / qCap ) * FCapLab / bhmDet -
+   ( fLab / qLab ) * FLabLab / bhmDet )
> range( dat$esmCapLab )
```

```
[1] 1 1
```

```
> dat$esmLabCap <- with( dat, ( fCap / qLab ) * FCapLab / bhmDet -
+   ( fCap / qCap ) * FCapCap / bhmDet )
> range( dat$esmLabCap )
```

```
[1] 1 1
```

```
> dat$esmCapMat <- with( dat, ( fMat / qCap ) * FCapMat / bhmDet -
+   ( fMat / qMat ) * FMatMat / bhmDet )
> range( dat$esmCapMat )
```

```
[1] 1 1
```

```
> dat$esmMatCap <- with( dat, ( fCap / qMat ) * FCapMat / bhmDet -
+   ( fCap / qCap ) * FCapCap / bhmDet )
> range( dat$esmMatCap )
```

```
[1] 1 1
```

```
> dat$esmLabMat <- with( dat, ( fMat / qLab ) * FLabMat / bhmDet -
+   ( fMat / qMat ) * FMatMat / bhmDet )
> range( dat$esmLabMat )
```

```
[1] 1 1
```

```
> dat$esmMatLab <- with( dat, ( fLab / qMat ) * FLabMat / bhmDet -
+   ( fLab / qLab ) * FLabLab / bhmDet )
> range( dat$esmMatLab )
```

[1] 1 1

As with the direct elasticities of substitution and the Allen elasticities of substitution, the Cobb-Douglas functional form implies that all Morishima elasticities of substitution are always equal to one, irrespective of the input and output quantities and the estimated parameters.

Using condition 2.24, we can show that all Morishima elasticities of substitution are always one ($\sigma_{ij}^M = 1 \forall i \neq j$), if all Allen elasticities of substitution are one ($\sigma_{ij} = 1 \forall i \neq j$):

$$\sigma_{ij}^M = K_j \sigma_{ij} - K_j \sigma_{jj} = K_j + \sum_{k \neq j} K_k \sigma_{kj} = \sum_k K_k = 1 \quad (2.96)$$

Hence, the Cobb-Douglas function cannot be used to analyze the substitutability of the inputs, because it will always return (direct, Allen and Morishima) elasticities of substitution that are equal to one, even if the true elasticities of substitution are close to zero or close to infinity.

2.4.12 Quasiconcavity

We start by checking whether our estimated Cobb-Douglas production function is quasiconcave at the first observation:

```
> bhm
```

```
      [,1]      [,2]      [,3]      [,4]
[1,] 0.000000 6.229014e+00 6.031225e+00 59.0909133861
[2,] 6.229014 -6.202845e-05 1.169835e-05 0.0001146146
[3,] 6.031225 1.169835e-05 -5.423455e-06 0.0001109752
[4,] 59.090913 1.146146e-04 1.109752e-04 -0.0006462733
```

```
> det( bhm[ 1:2, 1:2 ] )
```

```
[1] -38.80062
```

```
> det( bhm[ 1:3, 1:3 ] )
```

```
[1] 0.003345742
```

```
> det( bhm )
```

```
[1] -1.013458e-05
```

The first leading principal minor of the bordered Hessian matrix is negative, the second principal minor is positive, and the third principal minor is negative. This means that our estimated Cobb-Douglas production function is quasiconcave at the first observation.

Now we check quasiconcavity at each observation based on the three-dimensional array that consists of the stacked bordered Hessian matrices of all observations:

```
> dat$quasiConc <- apply( bhmArray[ 1:2, 1:2, ], 3, det ) < 0 &
+   apply( bhmArray[ 1:3, 1:3, ], 3, det ) > 0 &
+   apply( bhmArray, 3, det ) < 0
> table( dat$quasiConc )
```

TRUE

140

Our estimated Cobb-Douglas production function is quasiconcave at all of the 140 observations.

In fact, all Cobb-Douglas production functions are quasiconcave in inputs if $A \geq 0$, $\alpha_1 \geq 0$, $\dots$, $\alpha_N \geq 0$, while Cobb-Douglas production functions are concave in inputs if $A \geq 0$, $\alpha_1 \geq 0$, $\dots$, $\alpha_N \geq 0$, and the technology has decreasing or constant returns to scale ($\sum_{i=1}^N \alpha_i \leq 1$).³

2.4.13 First-order conditions for profit maximization

In this section, we will check to what extent the first-order conditions (2.40) for profit maximization are fulfilled, i.e. to what extent the firms use the optimal input quantities. We do this by comparing the marginal value products of the inputs with the corresponding input prices. We can calculate the marginal value products by multiplying the marginal products by the output price:

```
> dat$mvpCapCd <- dat$pOut * dat$fCap
> dat$mvpLabCd <- dat$pOut * dat$fLab
> dat$mvpMatCd <- dat$pOut * dat$fMat
```

The command `compPlot` (package *miscTools*) can be used to compare the marginal value products with the corresponding input prices:

```
> compPlot( dat$pCap, dat$mvpCapCd )
> compPlot( dat$pLab, dat$mvpLabCd )
> compPlot( dat$pMat, dat$mvpMatCd )
> compPlot( dat$pCap, dat$mvpCapCd, log = "xy" )
> compPlot( dat$pLab, dat$mvpLabCd, log = "xy" )
> compPlot( dat$pMat, dat$mvpMatCd, log = "xy" )
```

The resulting graphs are shown in figure 2.18. They indicate that the marginal value products are always nearly equal to or higher than the corresponding input prices. This indicates that (almost) all firms could increase their profit by using more of all inputs. Given that the estimated Cobb-Douglas technology exhibits increasing returns to scale, it is not surprising that (almost) all firms would gain from increasing all input quantities. Therefore, the question arises why the firms in the sample did not do this. This question has already been addressed in section 2.3.10.

³See, e.g., <http://econren.weebly.com/uploads/9/0/1/5/9015734/lecture16.pdf> or <http://web.mit.edu/14.102/www/ps/psisol.pdf>.

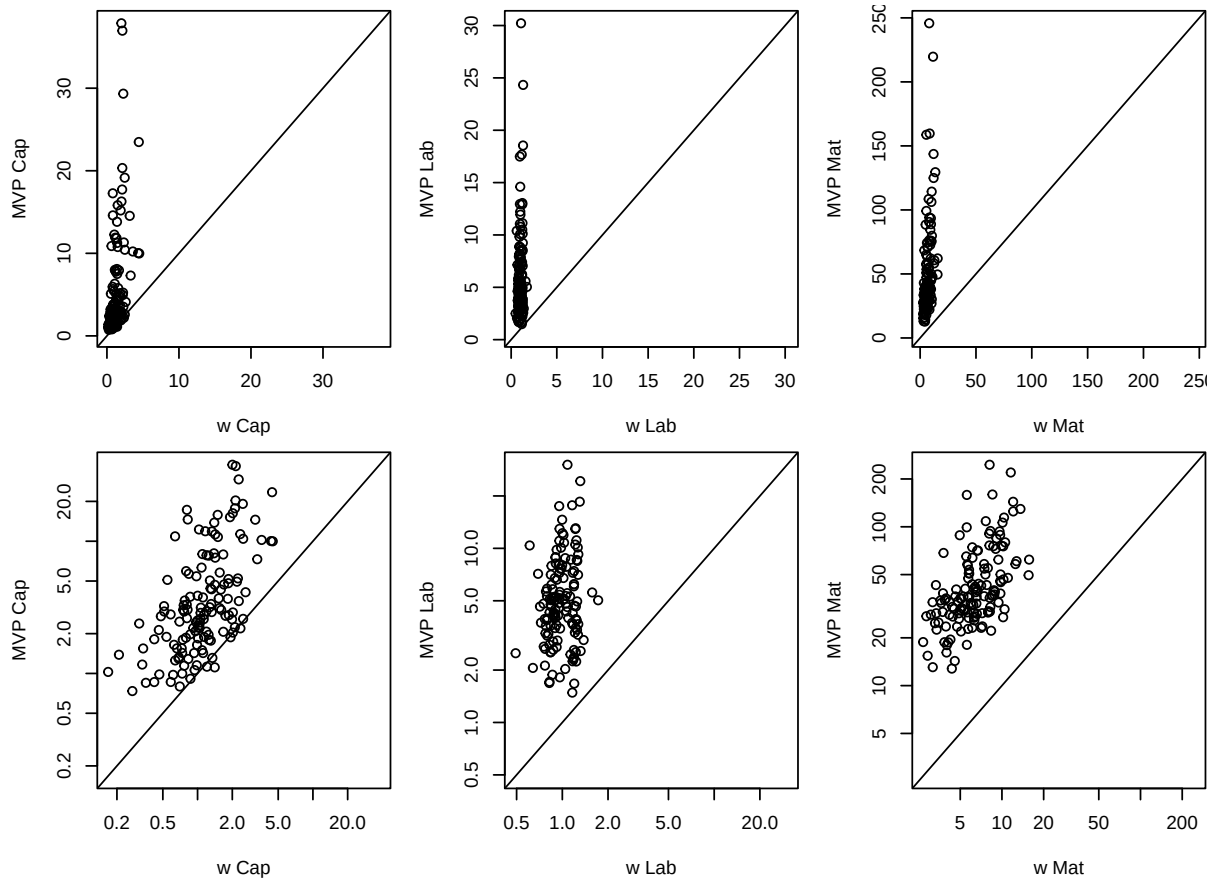


Figure 2.18: Marginal value products and corresponding input prices

2.4.14 First-order conditions for cost minimization

As the marginal rates of technical substitution differ between observations for the Cobb-Douglas functional form, we use scatter plots for visualizing the comparison of the input price ratios with the negative inverse marginal rates of technical substitution:

```
> compPlot( dat$pCap / dat$pLab, - dat$mrtsLabCapCD )
> compPlot( dat$pCap / dat$pMat, - dat$mrtsMatCapCD )
> compPlot( dat$pLab / dat$pMat, - dat$mrtsMatLabCD )
> compPlot( dat$pCap / dat$pLab, - dat$mrtsLabCapCD, log = "xy" )
> compPlot( dat$pCap / dat$pMat, - dat$mrtsMatCapCD, log = "xy" )
> compPlot( dat$pLab / dat$pMat, - dat$mrtsMatLabCD, log = "xy" )
```

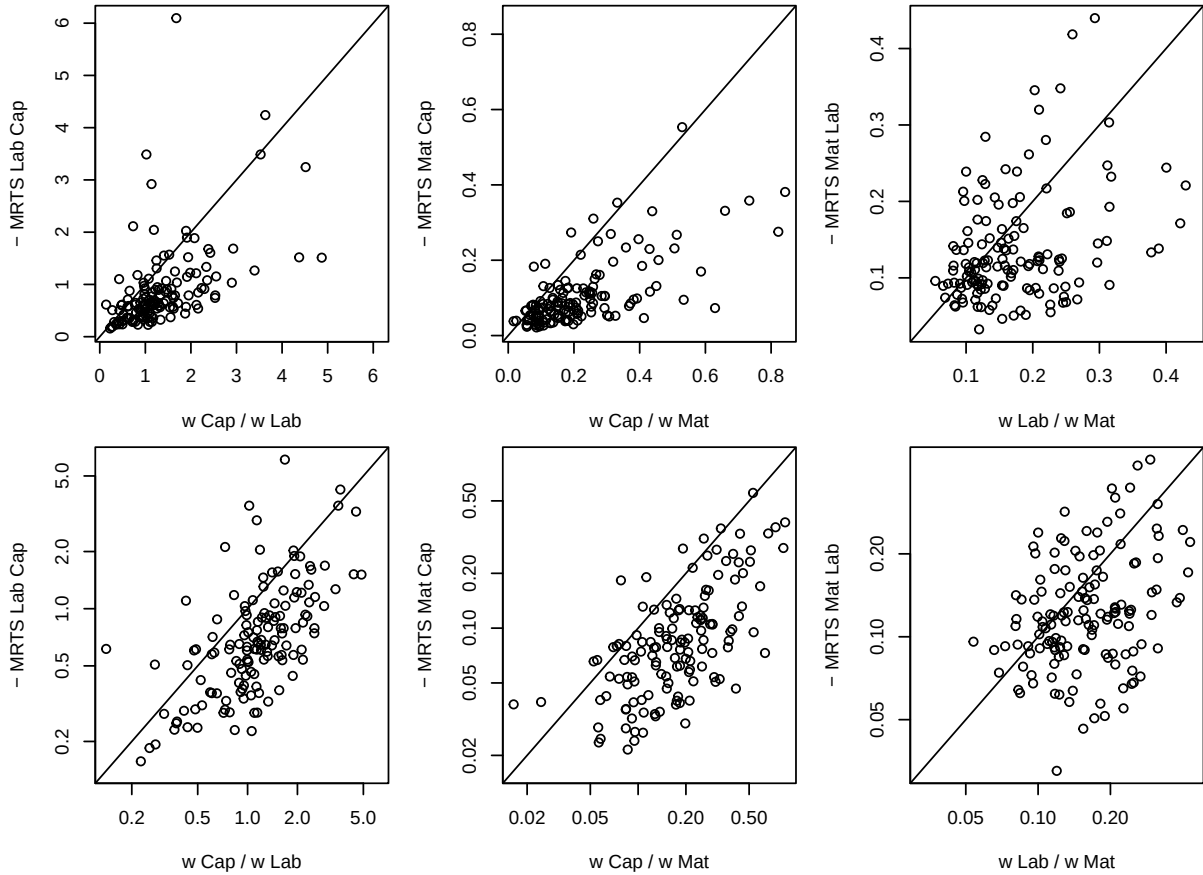


Figure 2.19: First-order conditions for cost minimization

The resulting graphs are shown in figure 2.19.

Furthermore, we use histograms to visualize the (absolute and relative) differences between the input price ratios and the corresponding negative inverse marginal rates of technical substitution:

```
> hist( - dat$mrtsLabCapCD - dat$pCap / dat$pLab )
> hist( - dat$mrtsMatCapCD - dat$pCap / dat$pMat )
```

```
> hist( - dat$mrtsMatLabCD - dat$pLab / dat$pMat )
> hist( log( - dat$mrtsLabCapCD / ( dat$pCap / dat$pLab ) ) )
> hist( log( - dat$mrtsMatCapCD / ( dat$pCap / dat$pMat ) ) )
> hist( log( - dat$mrtsMatLabCD / ( dat$pLab / dat$pMat ) ) )
```

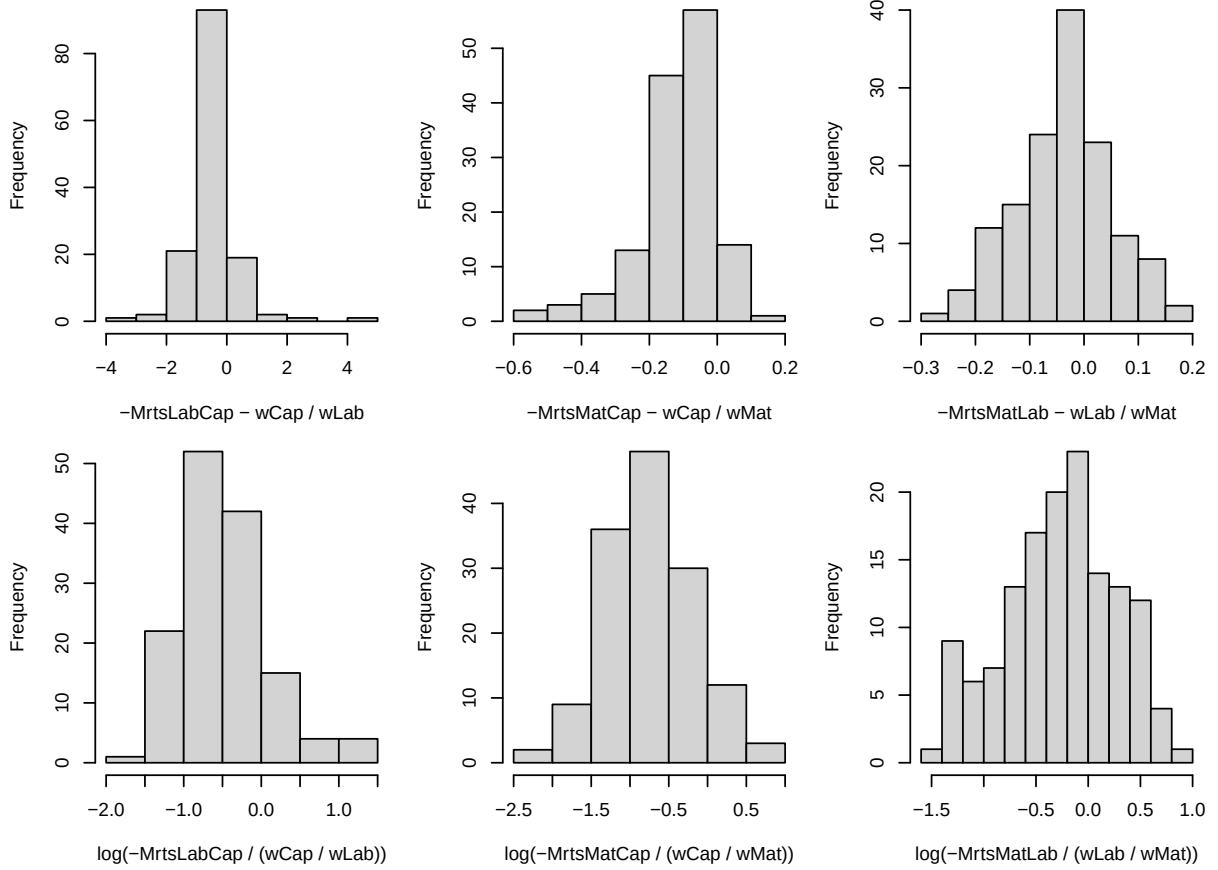


Figure 2.20: First-order conditions for costs minimization

The resulting graphs are shown in figure 2.20. The left graphs in figures 2.19 and 2.20 show that the ratio between the capital price and the labor price is larger than the absolute value of the marginal rate of technical substitution between labor and capital for the most firms in the sample:

$$\frac{w_{cap}}{w_{lab}} > -MRTS_{lab,cap} = \frac{MP_{cap}}{MP_{lab}} \quad (2.97)$$

Hence, most firms can get closer to the minimum of their production costs by substituting labor for capital, because this will decrease the marginal product of labor and increase the marginal product of capital so that the absolute value of the MRTS between labor and capital increases and gets closer to the corresponding input price ratio. Similarly, the graphs in the middle column indicate that most firms should substitute materials for capital and the graphs on the right indicate that the majority of the firms should substitute materials for labor. Hence, the majority

of the firms could reduce production costs particularly by using less capital and more materials⁴ but there might be (legal) regulations that restrict the use of materials (e.g. fertilizers, pesticides).

Given that cost minimization (as well as profit maximisation) implies that the cost shares are equal to the corresponding output elasticities divided by the elasticity of scale (see equation 2.60), we can check the conditions for cost minimization also by:

```
> hist( dat$sCap )
> abline( v = coef( prodCD )[ "log(qCap)" ] / sum( coef( prodCD )[-1] ),
+   lwd = 2 )
> hist( dat$sLab )
> abline( v = coef( prodCD )[ "log(qLab)" ] / sum( coef( prodCD )[-1] ),
+   lwd = 2 )
> hist( dat$sMat )
> abline( v = coef( prodCD )[ "log(qMat)" ] / sum( coef( prodCD )[-1] ),
+   lwd = 2 )
```

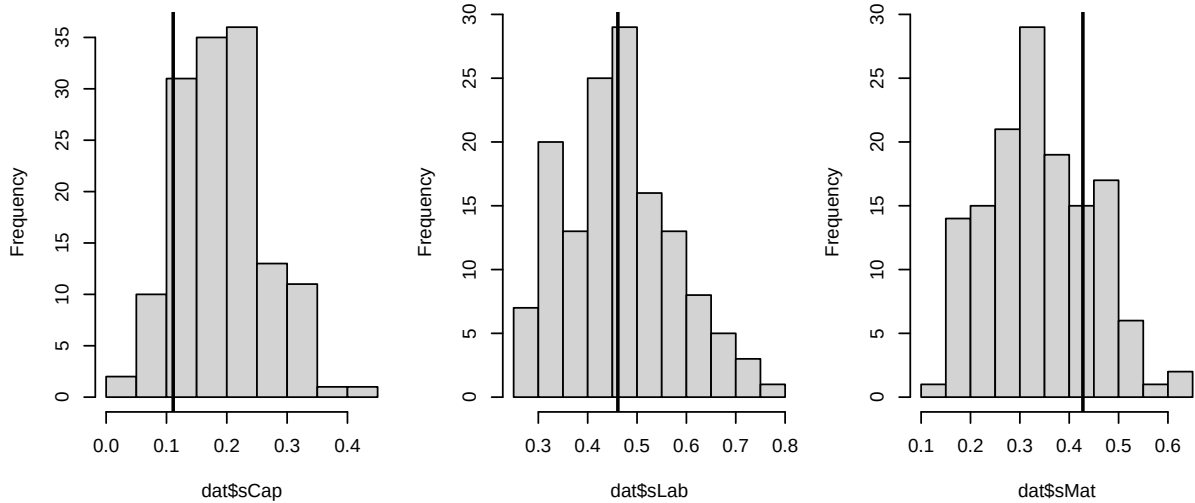


Figure 2.21: Cost shares and output elasticities divided by the elasticity of scale

The resulting graphs are shown in figure 2.21. They indicate that the majority of firms uses more capital and less material than would be cost-minimizing according to our estimated Cobb-Douglas production function.

⁴This generally confirms the results of the linear production function for the relationships between capital and labor and the relationship between capital and materials. However, in contrast to the linear production function, the results obtained by the Cobb-Douglas functional form indicate that most firms should substitute materials for labor (rather than the other way round).

2.4.15 Derived input demand functions and output supply functions

Given a Cobb-Douglas production function (2.79), the input quantities chosen by a profit maximizing producer are

$$x_i(p, w) = \begin{cases} \frac{\alpha_i}{w_i} \left(P A \prod_j \left(\frac{\alpha_j}{w_j} \right)^{\alpha_j} \right)^{\frac{1}{1-\alpha}} & \text{if } \alpha < 1 \\ 0 \vee \infty & \text{if } \alpha = 1 \\ \infty & \text{if } \alpha > 1 \end{cases} \quad (2.98)$$

and the output quantity is

$$y(p, w) = \begin{cases} \left(A P^\alpha \prod_j \left(\frac{\alpha_j}{w_j} \right)^{\alpha_j} \right)^{\frac{1}{1-\alpha}} & \text{if } \alpha < 1 \\ 0 \vee \infty & \text{if } \alpha = 1 \\ \infty & \text{if } \alpha > 1 \end{cases} \quad (2.99)$$

with $\alpha = \sum_j \alpha_j$. Hence, if the Cobb-Douglas production function exhibits increasing returns to scale ($\epsilon = \alpha > 1$), the optimal input and output quantities are infinity. As our estimated Cobb-Douglas production function has increasing returns to scale, the optimal input quantities are infinity. Therefore, we cannot evaluate the effect of prices on the optimal input quantities.

A cost minimizing producer would choose the following input quantities:

$$x_i(w, y) = \left(\frac{y}{A} \prod_{j \neq i} \left(\frac{\alpha_i w_j}{\alpha_j w_i} \right)^{\alpha_j} \right)^{\frac{1}{\alpha}} \quad (2.100)$$

For our three-input Cobb-Douglas production function, we get following conditional input demand functions

$$x_{cap}(w, y) = \left(\frac{y}{A} \left(\frac{\alpha_{cap}}{w_{cap}} \right)^{\alpha_{lab} + \alpha_{mat}} \left(\frac{w_{lab}}{\alpha_{lab}} \right)^{\alpha_{lab}} \left(\frac{w_{mat}}{\alpha_{mat}} \right)^{\alpha_{mat}} \right)^{\frac{1}{\alpha_{cap} + \alpha_{lab} + \alpha_{mat}}} \quad (2.101)$$

$$x_{lab}(w, y) = \left(\frac{y}{A} \left(\frac{w_{cap}}{\alpha_{cap}} \right)^{\alpha_{cap}} \left(\frac{\alpha_{lab}}{w_{lab}} \right)^{\alpha_{cap} + \alpha_{mat}} \left(\frac{w_{mat}}{\alpha_{mat}} \right)^{\alpha_{mat}} \right)^{\frac{1}{\alpha_{cap} + \alpha_{lab} + \alpha_{mat}}} \quad (2.102)$$

$$x_{mat}(w, y) = \left(\frac{y}{A} \left(\frac{w_{cap}}{\alpha_{cap}} \right)^{\alpha_{cap}} \left(\frac{w_{lab}}{\alpha_{lab}} \right)^{\alpha_{lab}} \left(\frac{\alpha_{mat}}{w_{mat}} \right)^{\alpha_{cap} + \alpha_{lab}} \right)^{\frac{1}{\alpha_{cap} + \alpha_{lab} + \alpha_{mat}}} \quad (2.103)$$

We can use these formulas to calculate the cost-minimizing input quantities based on the observed input prices (w) and the predicted output quantities ($f(x)$). Alternatively, we could calculate the cost-minimizing input quantities based on the observed input prices (w) and the *observed*

output quantities (y). However, in the latter case, the predicted output quantities based on the cost-minimizing input quantities would differ from the predicted output quantities based on the observed input quantities (i.e. $y = f(x(w, y)) \neq f(x)$) so that a comparison of the cost-minimizing input quantities ($x(w, y)$) with the observed input quantities (x) would be less useful. As the coefficients of the Cobb-Douglas function repeatedly occur in the formulas for calculating the cost-minimizing input quantities, it is convenient to define short-cuts for them:

```
> A <- exp( coef( prodCD )[ "(Intercept)" ] )
> aCap <- coef( prodCD )[ "log(qCap)" ]
> aLab <- coef( prodCD )[ "log(qLab)" ]
> aMat <- coef( prodCD )[ "log(qMat)" ]
```

Now, we can calculate the cost-minimizing input quantities:

```
> dat$qCapCD <- with( dat,
+   ( ( qOutCD / A ) * ( aCap / pCap )^( aLab + aMat )
+     * ( pLab / aLab )^aLab * ( pMat / aMat )^aMat
+   )^(1/( aCap + aLab + aMat ) ) )
> dat$qLabCD <- with( dat,
+   ( ( qOutCD / A ) * ( pCap / aCap )^aCap
+     * ( aLab / pLab )^( aCap + aMat ) * ( pMat / aMat )^aMat
+   )^(1/( aCap + aLab + aMat ) ) )
> dat$qMatCD <- with( dat,
+   ( ( qOutCD / A ) * ( pCap / aCap )^aCap
+     * ( pLab / aLab )^aLab * ( aMat / pMat )^( aCap + aLab )
+   )^(1/( aCap + aLab + aMat ) ) )
```

Before we continue, we will check whether it is indeed possible to produce the predicted output with the calculated cost-minimizing input quantities:

```
> dat$qOutTest <- with( dat,
+   A * qCapCD^aCap * qLabCD^aLab * qMatCD^aMat )
> all.equal( dat$qOutCD, dat$qOutTest )
```

```
[1] TRUE
```

Given that the output quantities predicted from the cost-minimizing input quantities are all equal to the output quantities predicted from the observed input quantities, we can be pretty sure that our calculations are correct. Now, we can use scatter plots to compare the cost-minimizing input quantities with the observed input quantities:

```
> compPlot( dat$qCapCD, dat$qCap )
> compPlot( dat$qLabCD, dat$qLab )
```

```
> compPlot( dat$qMatCD, dat$qMat )
> compPlot( dat$qCapCD, dat$qCap, log = "xy" )
> compPlot( dat$qLabCD, dat$qLab, log = "xy" )
> compPlot( dat$qMatCD, dat$qMat, log = "xy" )
```

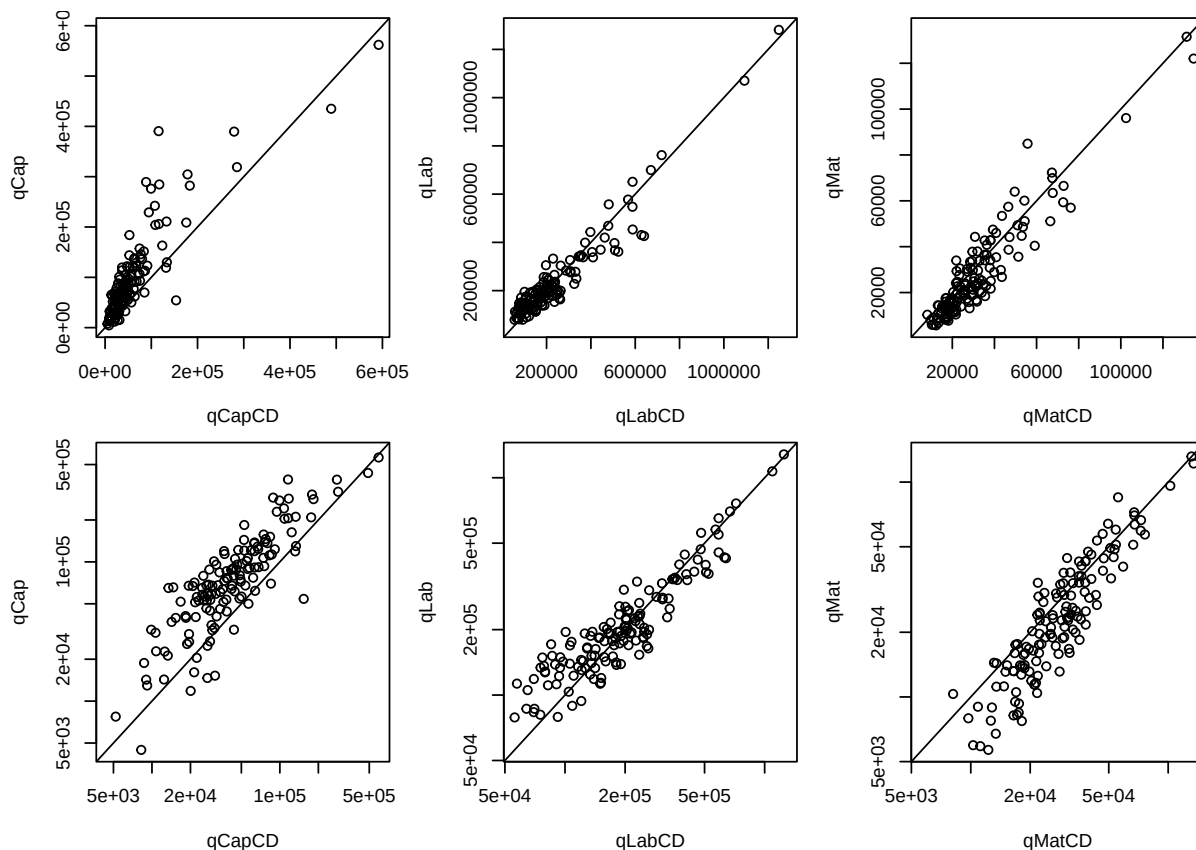


Figure 2.22: Optimal and observed input quantities

The resulting graphs are shown in figure 2.22. As we already found out in section 2.4.14, many firms could reduce their costs by substituting materials for capital.

We can also evaluate the potential for cost reductions by comparing the observed costs with the costs when using the cost-minimizing input quantities:

```
> dat$costProdCD <- with( dat,
+   pCap * qCapCD + pLab * qLabCD + pMat * qMatCD )
> mean( dat$costProdCD / dat$cost )
```

```
[1] 0.9308039
```

Our model predicts that the firms could reduce their costs on average by 7% by using cost-minimizing input quantities. The variation of the firms' cost reduction potentials are shown by a histogram:

```
> hist( dat$costProdCD / dat$cost )
```

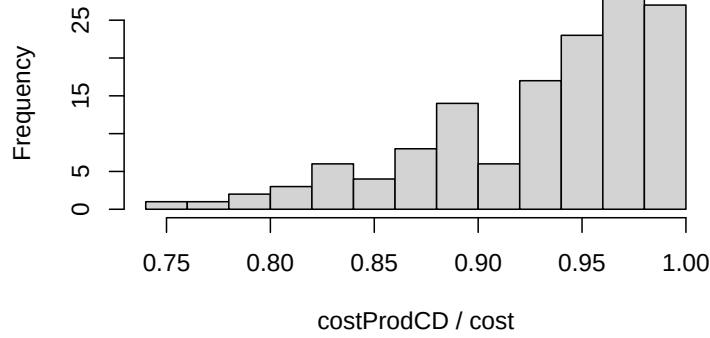


Figure 2.23: Minimum total costs as share of actual total costs

The resulting graph is shown in figure 2.23. While many firms have a rather small potential for reducing costs by reallocating input quantities, there are some firms that could save up to 25% of their total costs by using the optimal combination of input quantities.

We can also compare the observed input quantities with the cost-minimizing input quantities and the observed costs with the minimum costs for each single observation (e.g. when consulting individual firms in the sample):

```
> round( subset( dat, , c("qCap", "qCapCD", "qLab", "qLabCD", "qMat", "qMatCD",
+ "cost", "costProdCD") ) ) [1:5,]
```

	qCap	qCapCD	qLab	qLabCD	qMat	qMatCD	cost	costProdCD
1	84050	33720	360066	405349	34087	38038	846329	790968
2	39663	18431	249769	334442	40819	36365	580545	545777
3	37051	14257	140286	135701	24219	32176	306040	281401
4	21222	13300	83427	69713	18893	25890	199634	191709
5	44675	28400	89223	108761	14424	13107	226578	221302

2.4.16 Derived input demand elasticities

We can measure the effect of the input prices and the output quantity on the cost-minimizing input quantities by calculating the conditional price elasticities based on the partial derivatives of the conditional input demand functions (2.100) with respect to the input prices and the output quantity. In case of two inputs, we can calculate the demand elasticities of the first input by:

$$x_1(w, y) = \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}} \quad (2.104)$$

$$\epsilon_{11}(w, y) = \frac{\partial x_1(w, y)}{\partial w_1} \frac{w_1}{x_1(w, y)} \quad (2.105)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \alpha_2 \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2-1} \left(-\frac{\alpha_1 w_2}{\alpha_2 w_1^2} \right) \frac{w_1}{x_1} \quad (2.106)$$

$$= -\frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2-1} \frac{\alpha_1 w_2 \alpha_2}{\alpha_2 w_1 x_1} \quad (2.107)$$

$$= -\frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \frac{\alpha_2}{x_1} \quad (2.108)$$

$$= -\frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}} \frac{\alpha_2}{x_1} \quad (2.109)$$

$$= -\frac{1}{\alpha} x_1 \frac{\alpha_2}{x_1} = -\frac{\alpha_2}{\alpha} = \frac{\alpha_1 - \alpha}{\alpha} = \frac{\alpha_1}{\alpha} - 1 \quad (2.110)$$

$$\epsilon_{12}(w, y) = \frac{\partial x_1(w, y)}{\partial w_2} \frac{w_2}{x_1(w, y)} \quad (2.111)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \alpha_2 \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2-1} \frac{\alpha_1}{\alpha_2 w_1} \frac{1}{x_1} \frac{w_2}{x_1} \quad (2.112)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2-1} \frac{\alpha_1 w_2 \alpha_2}{\alpha_2 w_1 x_1} \quad (2.113)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \frac{\alpha_2}{x_1} \quad (2.114)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}} \frac{\alpha_2}{x_1} \quad (2.115)$$

$$= \frac{1}{\alpha} x_1 \frac{\alpha_2}{x_1} = \frac{\alpha_2}{\alpha} \quad (2.116)$$

$$\epsilon_{1y}(w, y) = \frac{\partial x_1(w, y)}{\partial y} \frac{y}{x_1(w, y)} \quad (2.117)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{1}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \frac{y}{x_1} \quad (2.118)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}-1} \frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \frac{1}{x_1} \quad (2.119)$$

$$= \frac{1}{\alpha} \left(\frac{y}{A} \left(\frac{\alpha_1 w_2}{\alpha_2 w_1} \right)^{\alpha_2} \right)^{\frac{1}{\alpha}} \frac{1}{x_1} \quad (2.120)$$

$$= \frac{1}{\alpha} x_1 \frac{1}{x_1} = \frac{1}{\alpha} \quad (2.121)$$

and analogously the demand elasticities of the second input:

$$x_2(w, y) = \left(\frac{y}{A} \left(\frac{\alpha_2 w_1}{\alpha_1 w_2} \right)^{\alpha_1} \right)^{\frac{1}{\alpha}} \quad (2.122)$$

$$\epsilon_{22}(w, y) = \frac{\partial x_2(w, y)}{\partial w_2} \frac{w_2}{x_2(w, y)} = -\frac{\alpha_1}{\alpha} = \frac{\alpha_2 - \alpha}{\alpha} = \frac{\alpha_2}{\alpha} - 1 \quad (2.123)$$

$$\epsilon_{21}(w, y) = \frac{\partial x_2(w, y)}{\partial w_1} \frac{w_1}{x_2(w, y)} = \frac{\alpha_1}{\alpha} \quad (2.124)$$

$$\epsilon_{2y}(w, y) = \frac{\partial x_2(w, y)}{\partial y} \frac{y}{x_2(w, y)} = \frac{1}{\alpha}. \quad (2.125)$$

One can similarly derive the input demand elasticities for the general case of N inputs:

$$\epsilon_{ij}(w, y) = \frac{\partial x_i(w, y)}{\partial w_j} \frac{w_j}{x_i(w, y)} = \frac{\alpha_j}{\alpha} - \delta_{ij} \quad (2.126)$$

$$\epsilon_{iy}(w, y) = \frac{\partial x_i(w, y)}{\partial y} \frac{y}{x_i(w, y)} = \frac{1}{\alpha}, \quad (2.127)$$

where δ_{ij} is (again) Kronecker's delta (2.95). We have calculated all these elasticities based on the estimated coefficients of the Cobb-Douglas production function; these elasticities are presented in table 2.1. If the price of capital increases by one percent, the cost-minimizing firm will decrease the use of capital by 0.89% and increase the use of labor and materials by 0.11% each. If the price of labor increases by one percent, the cost-minimizing firm will decrease the use of labor by 0.54% and increase the use of capital and materials by 0.46% each. If the price of materials increases by one percent, the cost-minimizing firm will decrease the use of materials by 0.57% and increase the use of capital and labor by 0.43% each. If the cost-minimizing firm increases the output quantity by one percent, (s)he will increase all input quantities by 0.68%.

Table 2.1: Conditional demand elasticities derived from Cobb-Douglas production function

	w_{cap}	w_{lab}	w_{mat}	y
x_{cap}	-0.89	0.46	0.43	0.68
x_{lab}	0.11	-0.54	0.43	0.68
x_{mat}	0.11	0.46	-0.57	0.68

2.5 Quadratic production function

2.5.1 Specification

A quadratic production function is defined as

$$y = \beta_0 + \sum_i \beta_i x_i + \frac{1}{2} \sum_i \sum_j \beta_{ij} x_i x_j, \quad (2.128)$$

where the restriction $\beta_{ij} = \beta_{ji}$ is required to identify all coefficients, because $x_i x_j$ and $x_j x_i$ are the same regressors. Based on this general form, we can derive the specification of a quadratic production function with three inputs:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \frac{1}{2} \beta_{11} x_1^2 + \frac{1}{2} \beta_{22} x_2^2 + \frac{1}{2} \beta_{33} x_3^2 + \beta_{12} x_1 x_2 + \beta_{13} x_1 x_3 + \beta_{23} x_2 x_3 \quad (2.129)$$

2.5.2 Estimation

We can estimate this quadratic production function with the command

```
> prodQuad <- lm( qOut ~ qCap + qLab + qMat
+      + I( 0.5 * qCap^2 ) + I( 0.5 * qLab^2 ) + I( 0.5 * qMat^2 )
```

2 Primal Approach: Production Function

```
+      + I( qCap * qLab ) + I( qCap * qMat ) + I( qLab * qMat ),
+      data = dat )
> summary( prodQuad )
```

Call:

```
lm(formula = qOut ~ qCap + qLab + qMat + I(0.5 * qCap^2) + I(0.5 *
      qLab^2) + I(0.5 * qMat^2) + I(qCap * qLab) + I(qCap * qMat) +
      I(qLab * qMat), data = dat)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-3928802	-695518	-186123	545509	4474143

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-2.911e+05	3.615e+05	-0.805	0.422072
qCap	5.270e+00	4.403e+00	1.197	0.233532
qLab	6.077e+00	3.185e+00	1.908	0.058581 .
qMat	1.430e+01	2.406e+01	0.595	0.553168
I(0.5 * qCap^2)	5.032e-05	3.699e-05	1.360	0.176039
I(0.5 * qLab^2)	-3.084e-05	2.081e-05	-1.482	0.140671
I(0.5 * qMat^2)	-1.896e-03	8.951e-04	-2.118	0.036106 *
I(qCap * qLab)	-3.097e-05	1.498e-05	-2.067	0.040763 *
I(qCap * qMat)	-4.160e-05	1.474e-04	-0.282	0.778206
I(qLab * qMat)	4.011e-04	1.112e-04	3.608	0.000439 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1344000 on 130 degrees of freedom

Multiple R-squared: 0.8449, Adjusted R-squared: 0.8342

F-statistic: 78.68 on 9 and 130 DF, p-value: < 2.2e-16

Although many of the estimated coefficients are statistically not significantly different from zero, the statistical significance of some quadratic and interaction terms indicates that the linear production function, which neither has quadratic terms nor interaction terms, is not suitable to model the true production technology. As the linear production function is “nested” in the quadratic production function, we can apply a “Wald test” or a likelihood ratio test to check whether the linear production function is rejected in favor of the quadratic production function. These tests can be done by the functions `waldtest` and `lrtest` (package *lmtest*):

```
> library( "lmtest" )
```

```
> waldtest( prodLin, prodQuad )
```

Wald test

```
Model 1: qOut ~ qCap + qLab + qMat
```

```
Model 2: qOut ~ qCap + qLab + qMat + I(0.5 * qCap^2) + I(0.5 * qLab^2) +
      I(0.5 * qMat^2) + I(qCap * qLab) + I(qCap * qMat) + I(qLab *
      qMat)
```

	Res.Df	Df	F	Pr(>F)
1	136			
2	130	6	8.1133	1.869e-07 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( prodLin, prodQuad )
```

Likelihood ratio test

```
Model 1: qOut ~ qCap + qLab + qMat
```

```
Model 2: qOut ~ qCap + qLab + qMat + I(0.5 * qCap^2) + I(0.5 * qLab^2) +
      I(0.5 * qMat^2) + I(qCap * qLab) + I(qCap * qMat) + I(qLab *
      qMat)
```

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	5	-2191.3			
2	11	-2169.1	6	44.529	5.806e-08 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

These tests show that the linear production function is clearly inferior to the quadratic production function and hence, should not be used for analyzing the production technology of the firms in this data set.

2.5.3 Properties

We cannot see from the estimated coefficients whether the monotonicity condition is fulfilled. Unless all coefficients are non-negative (but not necessarily the intercept), quadratic production functions cannot be globally monotone, because there will always be a set of input quantities that result in negative marginal products. We will check the monotonicity condition at each observation in section 2.5.5.

Our estimated quadratic production function does not fulfill the weak essentiality assumption, because the intercept is different from zero (but its deviation from zero is not statistically signif-

icant). The production technology described by a quadratic production function with more than one (relevant) input never shows strict essentiality.

The input requirement sets derived from quadratic production functions are always closed and non-empty.

The quadratic production function always returns finite, real, and single values but the non-negativity assumption is only fulfilled, if all coefficients (including the intercept), are non-negative.

All quadratic production functions are continuous and twice-continuously differentiable.

2.5.4 Predicted output quantities

We can obtain the predicted output quantities with the `fitted` method:

```
> dat$qOutQuad <- fitted( prodQuad )
```

We can evaluate the “fit” of the model by comparing the observed with the fitted output quantities:

```
> compPlot( dat$qOut, dat$qOutQuad )
> compPlot( dat$qOut, dat$qOutQuad, log = "xy" )
```

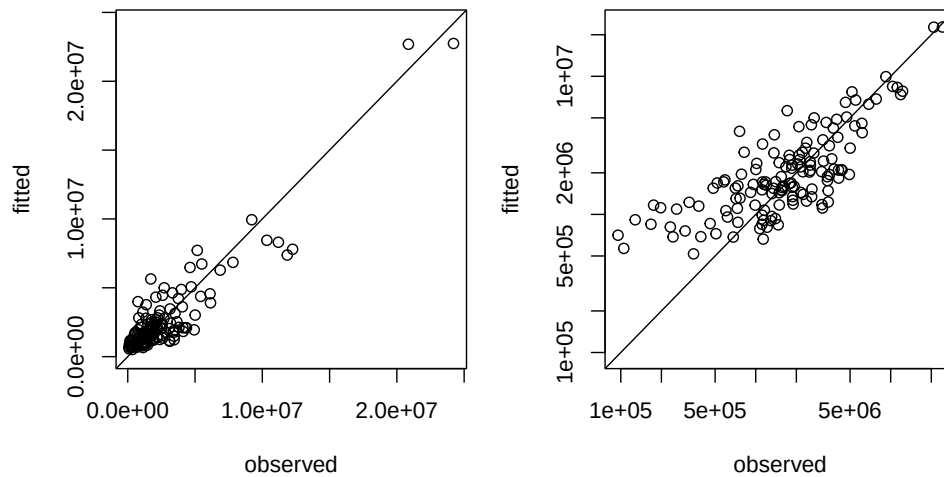


Figure 2.24: Quadratic production function: fit of the model

The resulting graphs are shown in figure 2.24. While the graph in the left panel uses a linear scale for the axes, the graph in the right panel uses a logarithmic scale for both axes. Hence, the deviations from the 45°-line illustrate the *absolute* deviations in the left panel and the *relative* deviations in the right panel. The fit of the model looks okay in the scatter plot on the left-hand side, but if we use a logarithmic scale on both axes (as in the graph on the right-hand side), we can see that the output quantity is over-estimated if the the observed output quantity is small.

As negative output quantities would render the corresponding output elasticities useless, we have carefully check the sign of the predicted output quantities:


```
> table( dat$qOutQuad >= 0 )
```

```
TRUE
```

```
140
```

Fortunately, not a single predicted output quantity is negative.

2.5.5 Marginal products

In case of a quadratic production function, the marginal products are

$$MP_i = \beta_i + \sum_j \beta_{ij} x_j \quad (2.130)$$

We can simplify the code for computing the marginal products and some other figures by using short names for the coefficients:

```
> b1 <- coef( prodQuad )[ "qCap" ]
> b2 <- coef( prodQuad )[ "qLab" ]
> b3 <- coef( prodQuad )[ "qMat" ]
> b11 <- coef( prodQuad )[ "I(0.5 * qCap^2)" ]
> b22 <- coef( prodQuad )[ "I(0.5 * qLab^2)" ]
> b33 <- coef( prodQuad )[ "I(0.5 * qMat^2)" ]
> b12 <- b21 <- coef( prodQuad )[ "I(qCap * qLab)" ]
> b13 <- b31 <- coef( prodQuad )[ "I(qCap * qMat)" ]
> b23 <- b32 <- coef( prodQuad )[ "I(qLab * qMat)" ]
```

Now, we can use the following commands to calculate the marginal products in R”:

```
> dat$mpCapQuad <- with( dat,
+   b1 + b11 * qCap + b12 * qLab + b13 * qMat )
> dat$mpLabQuad <- with( dat,
+   b2 + b21 * qCap + b22 * qLab + b23 * qMat )
> dat$mpMatQuad <- with( dat,
+   b3 + b31 * qCap + b32 * qLab + b33 * qMat )
```

We can visualize (the variation of) these marginal products with histograms:

```
> hist( dat$mpCapQuad, 15 )
> hist( dat$mpLabQuad, 15 )
> hist( dat$mpMatQuad, 15 )
```

The resulting graphs are shown in figure 2.25. If the firms increase capital input by one unit, the output of most firms will increase by around 2 units. If the firms increase labor input by one unit, the output of most firms will increase by around 5 units. If the firms increase material

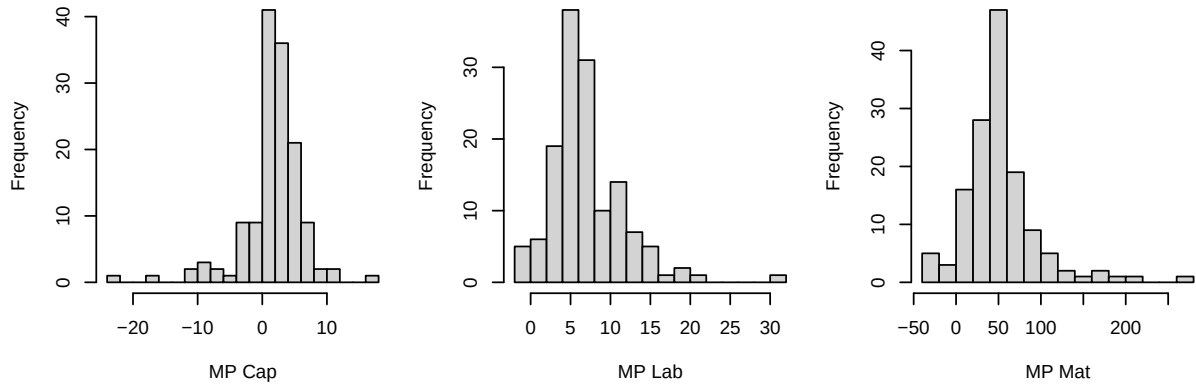


Figure 2.25: Quadratic production function: marginal products

input by one unit, the output of most firms will increase by around 50 units. These graphs also show that the monotonicity condition is not fulfilled for all observations:

```
> table( dat$mpCapQuad >= 0 )
```

```
FALSE  TRUE
    28   112
```

```
> table( dat$mpLabQuad >= 0 )
```

```
FALSE  TRUE
     5   135
```

```
> table( dat$mpMatQuad >= 0 )
```

```
FALSE  TRUE
     8   132
```

```
> dat$monoQuad <- with( dat, mpCapQuad >= 0 & mpLabQuad >= 0 & mpMatQuad >= 0 )
```

```
> table( dat$monoQuad )
```

```
FALSE  TRUE
    39   101
```

28 firms have a negative marginal product of capital, 5 firms have a negative marginal product of labor, and 8 firms have a negative marginal product of materials. In total the monotonicity condition is not fulfilled at 39 out of 140 observations. Although the monotonicity conditions are still fulfilled for the largest part of firms in our data set, these frequent violations could indicate a possible model misspecification.

2.5.6 Output elasticities

We can obtain output elasticities based on the quadratic production function by the standard formula for output elasticities:

$$\epsilon_i = MP_i \frac{x_i}{y} \quad (2.131)$$

As explained in section 2.4.11.1, we will use the predicted output quantities rather than the observed output quantities. We can calculate the output elasticities with:

```
> dat$eCapQuad <- with( dat, mpCapQuad * qCap / qOutQuad )
> dat$eLabQuad <- with( dat, mpLabQuad * qLab / qOutQuad )
> dat$eMatQuad <- with( dat, mpMatQuad * qMat / qOutQuad )
```

We can visualize (the variation of) these output elasticities with histograms:

```
> hist( dat$eCapQuad, 15 )
> hist( dat$eLabQuad, 15 )
> hist( dat$eMatQuad, 15 )
```

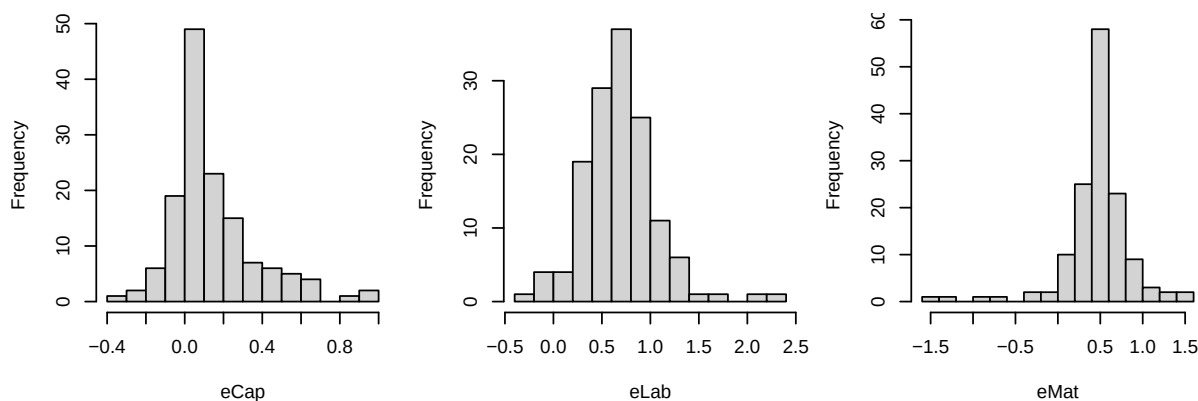


Figure 2.26: Quadratic production function: output elasticities

The resulting graphs are shown in figure 2.26. If the firms increase capital input by one percent, the output of most firms will increase by around 0.05 percent. If the firms increase labor input by one percent, the output of most firms will increase by around 0.7 percent. If the firms increase material input by one percent, the output of most firms will increase by around 0.5 percent.

2.5.7 Elasticity of scale

The elasticity of scale can—as always—be calculated as the sum of all output elasticities.

```
> dat$eScaleQuad <- dat$eCapQuad + dat$eLabQuad +
+   dat$eMatQuad
```

The (variation of the) elasticities of scale can be visualized with a histogram.

```
> hist( dat$eScaleQuad, 30 )
> hist( dat$eScaleQuad[ dat$monoQuad ], 30 )
```

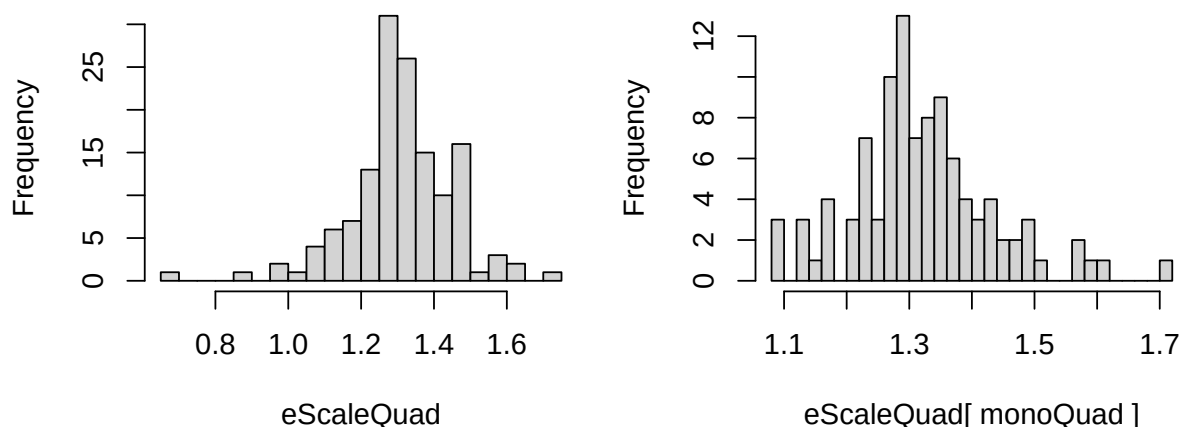


Figure 2.27: Quadratic production function: elasticities of scale

The resulting graphs are shown in figure 2.27. Only a very few firms (4 out of 140) experience decreasing returns to scale. If we only consider the observations where all monotonicity conditions are fulfilled, our results suggest that all firms have increasing returns to scale. Most firms have an elasticity of scale around 1.3. Hence, if these firms increase all input quantities by one percent, the output of most firms will increase by around 1.3 percent. These elasticities of scale are much more realistic than the elasticities of scale based on the linear production function.

Information on the optimal firm size can be obtained by analyzing the interrelationship between firm size and the elasticity of scale, where we can either use the observed output or the quantity index of the inputs as proxies of the firm size:

```
> plot( dat$qOut, dat$eScaleQuad, log = "x" )
> abline( 1, 0 )
> plot( dat$X, dat$eScaleQuad, log = "x" )
> abline( 1, 0 )
> plot( dat$qOut[ dat$monoQuad ], dat$eScaleQuad[ dat$monoQuad ], log = "x" )
> plot( dat$X[ dat$monoQuad ], dat$eScaleQuad[ dat$monoQuad ], log = "x" )
```

The resulting graphs are shown in figure 2.28. They all indicate that there are increasing returns to scale for all firm sizes in the sample. Hence, all firms in the sample would gain from increasing their size and the optimal firm size seems to be larger than the largest firm in the sample.

2.5.8 Marginal rates of technical substitution

We can calculate the marginal rates of technical substitution (MRTS) based on our estimated quadratic production function by following commands:

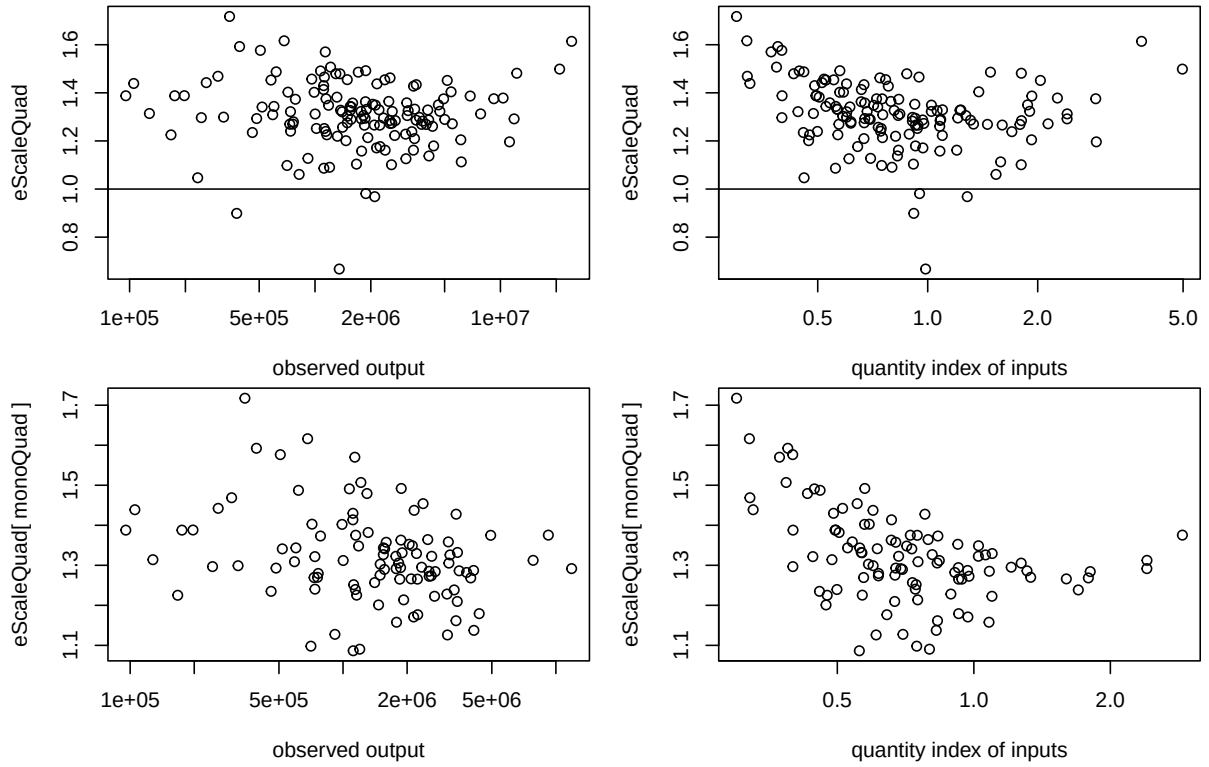


Figure 2.28: Quadratic production function: elasticities of scale at different firm sizes

```
> dat$mrtsCapLabQuad <- with( dat, - mpLabQuad / mpCapQuad )
> dat$mrtsLabCapQuad <- with( dat, - mpCapQuad / mpLabQuad )
> dat$mrtsCapMatQuad <- with( dat, - mpMatQuad / mpCapQuad )
> dat$mrtsMatCapQuad <- with( dat, - mpCapQuad / mpMatQuad )
> dat$mrtsLabMatQuad <- with( dat, - mpMatQuad / mpLabQuad )
> dat$mrtsMatLabQuad <- with( dat, - mpLabQuad / mpMatQuad )
```

As the marginal rates of technical substitution (MRTS) are meaningless if the monotonicity condition is not fulfilled, we visualize (the variation of) these MRTSs only for the observations, where the monotonicity condition is fulfilled:

```
> hist( dat$mrtsCapLabQuad[ dat$monoQuad ], 30 )
> hist( dat$mrtsLabCapQuad[ dat$monoQuad ], 30 )
> hist( dat$mrtsCapMatQuad[ dat$monoQuad ], 30 )
> hist( dat$mrtsMatCapQuad[ dat$monoQuad ], 30 )
> hist( dat$mrtsLabMatQuad[ dat$monoQuad ], 30 )
> hist( dat$mrtsMatLabQuad[ dat$monoQuad ], 30 )
```

The resulting graphs are shown in figure 2.29. As some outliers hide the variation of the majority of the RMRTS, we use function `colMedians` (package *miscTools*) to show the median values of the MRTS:

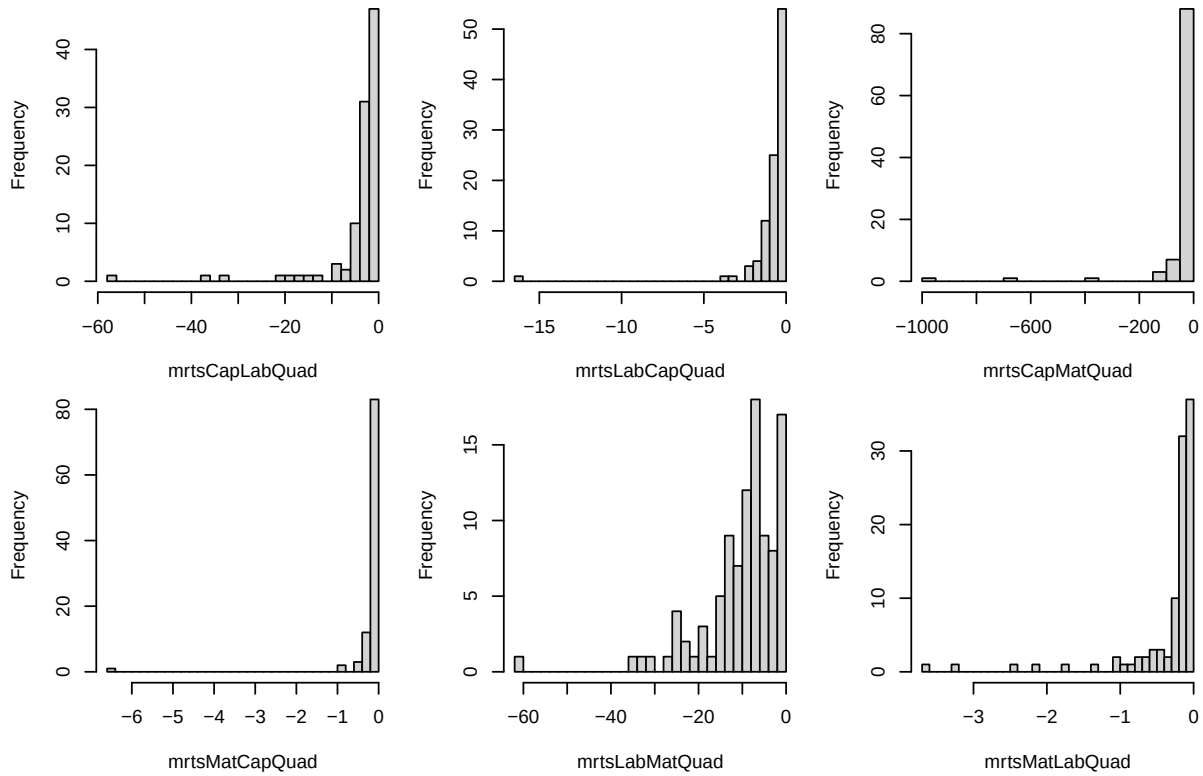


Figure 2.29: Quadratic production function: marginal rates of technical substitution (RMRTS)

```
> colMedians( subset( dat, monoQuad,
+   c( "mrtsCapLabQuad", "mrtsLabCapQuad", "mrtsCapMatQuad",
+     "mrtsMatCapQuad", "mrtsLabMatQuad", "mrtsMatLabQuad" ) ) )

mrtsCapLabQuad mrtsLabCapQuad mrtsCapMatQuad mrtsMatCapQuad mrtsLabMatQuad
-2.23505371    -0.44741654   -14.19802214    -0.07043235    -7.86423950
mrtsMatLabQuad
-0.12715788
```

Given that the median marginal rate of technical substitution between capital and labor is -2.24, a typical firm that reduces the use of labor by one unit, has to use around 2.24 additional units of capital in order to produce the same amount of output as before. Alternatively, the typical firm can replace one unit of labor by using 0.13 additional units of materials.

2.5.9 Relative marginal rates of technical substitution

As we do not have a practical interpretation of the units of measurement of the input quantities, the relative marginal rates of technical substitution (RMRTS) are practically more meaningful than the MRTS. The following commands calculate the RMRTS:

```
> dat$rmrtsCapLabQuad <- with( dat, - eLabQuad / eCapQuad )
> dat$rmrtsLabCapQuad <- with( dat, - eCapQuad / eLabQuad )
> dat$rmrtsCapMatQuad <- with( dat, - eMatQuad / eCapQuad )
> dat$rmrtsMatCapQuad <- with( dat, - eCapQuad / eMatQuad )
> dat$rmrtsLabMatQuad <- with( dat, - eMatQuad / eLabQuad )
> dat$rmrtsMatLabQuad <- with( dat, - eLabQuad / eMatQuad )
```

As the (relative) marginal rates of technical substitution are meaningless if the monotonicity condition is not fulfilled, we visualize (the variation of) these RMRTSs only for the observations, where the monotonicity condition is fulfilled:

```
> hist( dat$rmrtsCapLabQuad[ dat$monoQuad ], 30 )
> hist( dat$rmrtsLabCapQuad[ dat$monoQuad ], 30 )
> hist( dat$rmrtsCapMatQuad[ dat$monoQuad ], 30 )
> hist( dat$rmrtsMatCapQuad[ dat$monoQuad ], 30 )
> hist( dat$rmrtsLabMatQuad[ dat$monoQuad ], 30 )
> hist( dat$rmrtsMatLabQuad[ dat$monoQuad ], 30 )
```

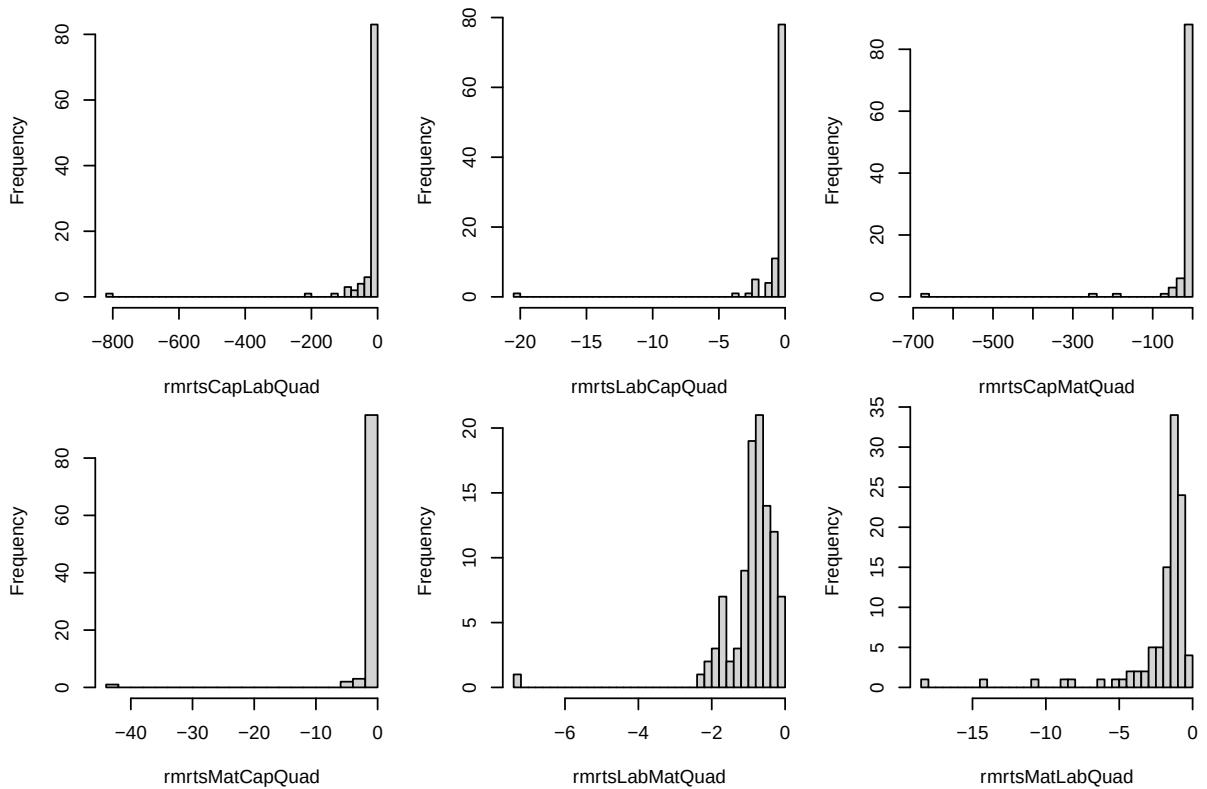


Figure 2.30: Quadratic production function: relative marginal rates of technical substitution (RMRTS)

The resulting graphs are shown in figure 2.30. As some outliers hide the variation of the majority of the RMRTS, we use function `colMedians` (package *miscTools*) to show the median values of

the RMRTS:

```
> colMedians( subset( dat, monoQuad,
+   c( "rmrtsCapLabQuad", "rmrtsLabCapQuad", "rmrtsCapMatQuad",
+     "rmrtsMatCapQuad", "rmrtsLabMatQuad", "rmrtsMatLabQuad" ) ) )

rmrtsCapLabQuad rmrtsLabCapQuad rmrtsCapMatQuad rmrtsMatCapQuad rmrtsLabMatQuad
-5.5741780      -0.1793986      -4.2567577      -0.2349206      -0.7745132
rmrtsMatLabQuad
-1.2911336
```

Given that the median relative marginal rate of technical substitution between capital and labor is -5.57, a typical firm that reduces the use of labor by one percent, has to use around 5.57 percent more capital in order to produce the same amount of output as before. Alternatively, the typical firm can replace one percent of labor by using 1.29 percent more materials.

2.5.10 Quasiconcavity

In order to check whether the estimated quadratic production function is quasiconcave, we need to construct the bordered Hessian matrix. As the first derivatives of the quadratic production function differ between observations, also the bordered Hessian matrix differs between observations. As a starting point, we construct the bordered Hessian matrix just for the first observation:

```
> bhm <- matrix( 0, nrow = 4, ncol = 4 )
> bhm[ 1, 2 ] <- bhm[ 2, 1 ] <- dat$mpCapQuad[ 1 ]
> bhm[ 1, 3 ] <- bhm[ 3, 1 ] <- dat$mpLabQuad[ 1 ]
> bhm[ 1, 4 ] <- bhm[ 4, 1 ] <- dat$mpMatQuad[ 1 ]
> bhm[ 2, 2 ] <- b11
> bhm[ 3, 3 ] <- b22
> bhm[ 4, 4 ] <- b33
> bhm[ 2, 3 ] <- bhm[ 3, 2 ] <- b12
> bhm[ 2, 4 ] <- bhm[ 4, 2 ] <- b13
> bhm[ 3, 4 ] <- bhm[ 4, 3 ] <- b23
> print(bhm)
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.000000	-3.068305e+00	6.042883e+00	9.063140e+01
[2,]	-3.068305	5.032000e-05	-3.096510e-05	-4.160252e-05
[3,]	6.042883	-3.096510e-05	-3.084032e-05	4.011378e-04
[4,]	90.631400	-4.160252e-05	4.011378e-04	-1.895501e-03

Based on this bordered Hessian matrix, we can check whether the estimated quadratic production function is quasiconcave at the first observation:


```
> det( bhm[ 1:2, 1:2 ] )
```

```
[1] -9.414493
```

```
> det( bhm[ 1:3, 1:3 ] )
```

```
[1] -0.0003988879
```

```
> det( bhm )
```

```
[1] 3.541548e-05
```

Quasiconcavity requires that the first determinant calculated above (i.e. the determinant of the upper-left 2×2 submatrix) is negative and the following determinants alternate in sign. As this condition is not fulfilled for the second and third determinant, we can conclude that the estimated quadratic production function is not quasiconcave at the first observation.

Now, we check whether our estimated quadratic production function is quasiconcave at each observation by stacking the bordered Hessian matrices of all observation on top of each other, which results in a three-dimensional array:

```
> bhmQuad <- array( 0, c( 4, 4, nrow( dat ) ) )
> bhmQuad[ 1, 2, ] <- bhmQuad[ 2, 1, ] <- dat$mpCapQuad
> bhmQuad[ 1, 3, ] <- bhmQuad[ 3, 1, ] <- dat$mpLabQuad
> bhmQuad[ 1, 4, ] <- bhmQuad[ 4, 1, ] <- dat$mpMatQuad
> bhmQuad[ 2, 2, ] <- b11
> bhmQuad[ 3, 3, ] <- b22
> bhmQuad[ 4, 4, ] <- b33
> bhmQuad[ 2, 3, ] <- bhmQuad[ 3, 2, ] <- b12
> bhmQuad[ 2, 4, ] <- bhmQuad[ 4, 2, ] <- b13
> bhmQuad[ 3, 4, ] <- bhmQuad[ 4, 3, ] <- b23
> dat$quasiConcQuad <- apply( bhmQuad[ 1:2, 1:2, ], 3, det ) <= 0 &
+   apply( bhmQuad[ 1:3, 1:3, ], 3, det ) >= 0 &
+   apply( bhmQuad, 3, det ) <= 0
> table( dat$quasiConcQuad )
```

```
FALSE
```

```
140
```

Our estimated quadratic production function is quasiconcave at none of the 140 observations, which implies that the isoquants are convex not at a single observation.

We can check whether the (partial) isoquants are convex if we hold one input fixed and only vary the other two inputs:

```
> bhmQuadCapLab <- bhmQuad[ -4, -4, ]
> dat$quasiConcCapLabQuad <-
+   apply( bhmQuadCapLab[ 1:2, 1:2, ], 3, det ) <= 0 &
+   apply( bhmQuadCapLab, 3, det ) >= 0
> table( dat$quasiConcCapLabQuad )
```

```
FALSE  TRUE
  121    19
```

```
> bhmQuadCapMat <- bhmQuad[ -3, -3, ]
> dat$quasiConcCapMatQuad <-
+   apply( bhmQuadCapMat[ 1:2, 1:2, ], 3, det ) <= 0 &
+   apply( bhmQuadCapMat, 3, det ) >= 0
> table( dat$quasiConcCapMatQuad )
```

```
FALSE  TRUE
  114    26
```

```
> bhmQuadLabMat <- bhmQuad[ -2, -2, ]
> dat$quasiConcLabMatQuad <-
+   apply( bhmQuadLabMat[ 1:2, 1:2, ], 3, det ) <= 0 &
+   apply( bhmQuadLabMat, 3, det ) >= 0
> table( dat$quasiConcLabMatQuad )
```

```
TRUE
  140
```

While the isoquants between labor and materials holding capital constant are convex at all 140 observations, the isoquants between capital and labor holding materials constant and between capital and materials holding labor constant are non-convex at most of the observations.

The non-convex isoquants between capital and the other two inputs at many observations are partly caused by a positive second derivative of the estimated quadratic production function with respect to capital (coefficient $\beta_{cap, cap} > 0$), which indicates increasing returns to capital.

2.5.11 Elasticities of substitution

2.5.11.1 Direct elasticities of substitution

We can calculate the direct elasticities of substitution using equation (2.20):

```
> dat$esdCapLabQuad <- with( dat, - ( mpCapQuad * mpLabQuad *
+   ( qCap * mpCapQuad + qLab * mpLabQuad ) ) /
+   ( qCap * qLab * ( b11 * mpLabQuad^2 -
```

```

+      2 * b12 * mpCapQuad * mpLabQuad + b22 * mpCapQuad^2 ) ) )
> dat$esdCapMatQuad <- with( dat, - ( mpCapQuad * mpMatQuad *
+   ( qCap * mpCapQuad + qMat * mpMatQuad ) ) /
+   ( qCap * qMat * ( b11 * mpMatQuad^2 -
+     2 * b13 * mpCapQuad * mpMatQuad + b33 * mpCapQuad^2 ) ) )
> dat$esdLabMatQuad <- with( dat, - ( mpLabQuad * mpMatQuad *
+   ( qLab * mpLabQuad + qMat * mpMatQuad ) ) /
+   ( qLab * qMat * ( b22 * mpMatQuad^2
+     - 2 * b23 * mpLabQuad * mpMatQuad + b33 * mpLabQuad^2 ) ) )

```

As the elasticities of substitution measure changes in the marginal rates of technical substitution (MRTS) and the MRTS are meaningless if the monotonicity conditions are not fulfilled, also the elasticities of substitution are meaningless if the monotonicity conditions are not fulfilled. Hence, we visualize (the variation of) the direct elasticities of substitution only for the observations, where the monotonicity condition is fulfilled:

```

> hist( dat$esdCapLabQuad[ dat$monoQuad ], 30 )
> hist( dat$esdCapMatQuad[ dat$monoQuad ], 30 )
> hist( dat$esdLabMatQuad[ dat$monoQuad ], 30 )

```

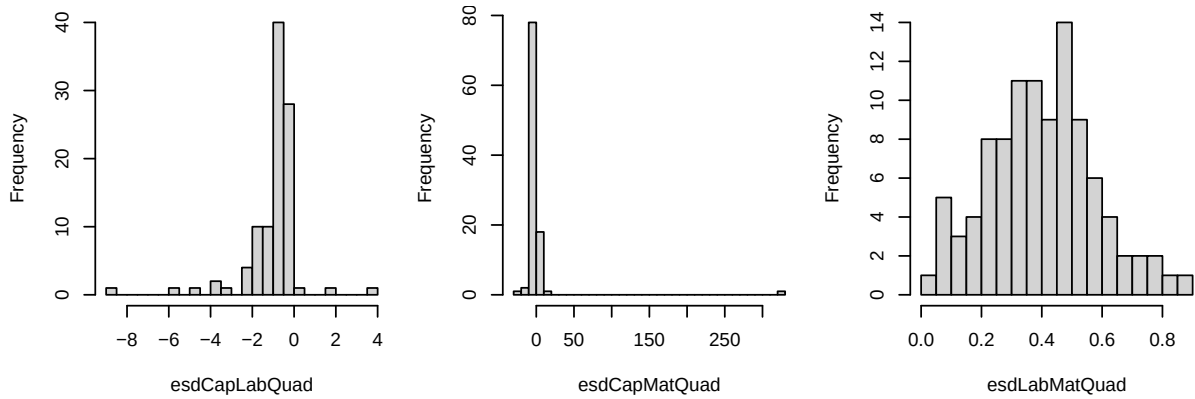


Figure 2.31: Quadratic production function: direct elasticities of substitution

The resulting graphs are shown in figure 2.31. The estimated direct elasticities of substitution between capital and labor are almost always negative, the estimated direct elasticities of substitution between capital and materials are partly negative and partly positive, and the estimated direct elasticities of substitution between labor and materials are always positive. As the direct elasticities of substitution assume that only the quantities of the two considered inputs are varied, it is expected that all direct elasticities of substitution are non-negative. However, the estimated quadratic production function is not quasiconcave, which implies that the isoquants are not convex. In case of non-convex isoquants, the elasticities of substitution do not indicate substitutability between inputs.

As only the (partial) isoquants between labor and materials holding capital constant are convex at all observations (see section 2.5.10) and the direct elasticities of substitution between labor and materials also assume that capital is held constant, we focus on these elasticities of substitution. For all firms in our sample, the estimated values of the direct elasticities of substitution between labor and materials lie between the value implied by the Leontief production function ($\sigma = 0$) and the value implied by the Cobb-Douglas production function ($\sigma = 1$). Hence, the substitutability between labor and materials seems to be between very low and moderate. In fact, the direct elasticity of substitution between labor and materials is for most of the firms around 0.4. Hence, if the firms keep the capital quantity and the output quantity unchanged and substitute labor for materials (or vice versa) so that the ratio between the labor quantity and the quantity of materials increases (decreases) by around 0.4 percent, the MRTS between labor and materials increases (decreases) by one percent. If the ratio between the materials price and the labor price increases by one percent, firms that keep the capital input and output quantity unchanged and minimize their costs will substitute labor for materials so that ratio between the labor quantity and the quantity of materials increases by around 0.4 percent. Hence, the relative change of the ratio of the input quantities is considerably smaller than the relative change of the ratio of the input prices, which indicates a low substitutability between labor and materials.

2.5.11.2 Allen elasticities of substitution

In the following, we calculate the Allen elasticities of substitution.⁵ In order to check whether our calculations are correct, we can use equation (2.24) to derive the following conditions:

$$\sum_i x_i MP_i \sigma_{ij} = 0 \quad \forall j \quad (2.132)$$

In order to check this condition, we need to calculate not only (normal) elasticities of substitution ($\sigma_{ij}; i \neq j$) but also economically not meaningful “elasticities of self-substitution” (σ_{ii}):

```
> dat$FCapLabQuad <- - apply( bhmQuad[ -2, -3, ], 3, det )
> dat$FCapMatQuad <- apply( bhmQuad[ -2, -4, ], 3, det )
> dat$FLabMatQuad <- - apply( bhmQuad[ -3, -4, ], 3, det )
> dat$FCapCapQuad <- apply( bhmQuad[ -2, -2, ], 3, det )
> dat$FLabLabQuad <- apply( bhmQuad[ -3, -3, ], 3, det )
> dat$FMatMatQuad <- apply( bhmQuad[ -4, -4, ], 3, det )
> dat$bhmDetQuad <- apply( bhmQuad, 3, det )
> dat$enumeratorQuad <- with( dat,
+   qCap * mpCapQuad + qLab * mpLabQuad + qMat * mpMatQuad )
> dat$esaCapLabQuad <- with( dat,
```

⁵We do not calculate the Morishima elasticities of substitution for the quadratic production function. The calculation of the Morishima elasticities of substitution requires only minimal changes of the code for calculating the Allen elasticities of substitution (compare sections 2.4.11.2 and 2.4.11.3).

```
+ numeratorQuad / ( qCap * qLab ) * FCapLabQuad / bhmDetQuad )
> dat$esaCapMatQuad <- with( dat,
+ numeratorQuad / ( qCap * qMat ) * FCapMatQuad / bhmDetQuad )
> dat$esaLabMatQuad <- with( dat,
+ numeratorQuad / ( qLab * qMat ) * FLabMatQuad / bhmDetQuad )
> dat$esaCapCapQuad <- with( dat,
+ numeratorQuad / ( qCap * qCap ) * FCapCapQuad / bhmDetQuad )
> dat$esaLabLabQuad <- with( dat,
+ numeratorQuad / ( qLab * qLab ) * FLabLabQuad / bhmDetQuad )
> dat$esaMatMatQuad <- with( dat,
+ numeratorQuad / ( qMat * qMat ) * FMatMatQuad / bhmDetQuad )
```

Before we take a look at and interpret the elasticities of substitution, we check whether the conditions (2.132) are fulfilled:

```
> range( with( dat, qCap * mpCapQuad * esaCapCapQuad +
+ qLab * mpLabQuad * esaCapLabQuad + qMat * mpMatQuad * esaCapMatQuad ) )
[1] -1.788139e-07 2.235174e-08

> range( with( dat, qCap * mpCapQuad * esaCapLabQuad +
+ qLab * mpLabQuad * esaLabLabQuad + qMat * mpMatQuad * esaLabMatQuad ) )
[1] -1.629815e-09 2.561137e-09

> range( with( dat, qCap * mpCapQuad * esaCapMatQuad +
+ qLab * mpLabQuad * esaLabMatQuad + qMat * mpMatQuad * esaMatMatQuad ) )
[1] -1.862645e-09 1.303852e-08
```

The extremely small deviations from zero are most likely caused by rounding errors that are unavoidable on digital computers. This test does not prove that all our calculations are done correctly but if we had made a mistake, we would have discovered it with a very high probability. Hence, we can be rather sure that our calculations are correct.

As for the direct elasticities of substitution, we visualize (the variation of) the Allen elasticities of substitution only for the observations, where the monotonicity condition is fulfilled:

```
> hist( dat$esaCapLabQuad[ dat$monoQuad ], 30 )
> hist( dat$esaCapMatQuad[ dat$monoQuad ], 30 )
> hist( dat$esaLabMatQuad[ dat$monoQuad ], 30 )
```

The resulting graphs are shown in figure 2.32. As the Allen elasticities of substitution allow the other inputs to adjust, a meaningful interpretation requires that not only the partial isoquants of the two considered inputs are convex but that the production is quasiconcave. As the estimated quadratic production is not quasiconcave at any observation, we do not interpret the obtained Allen elasticities of substitution.

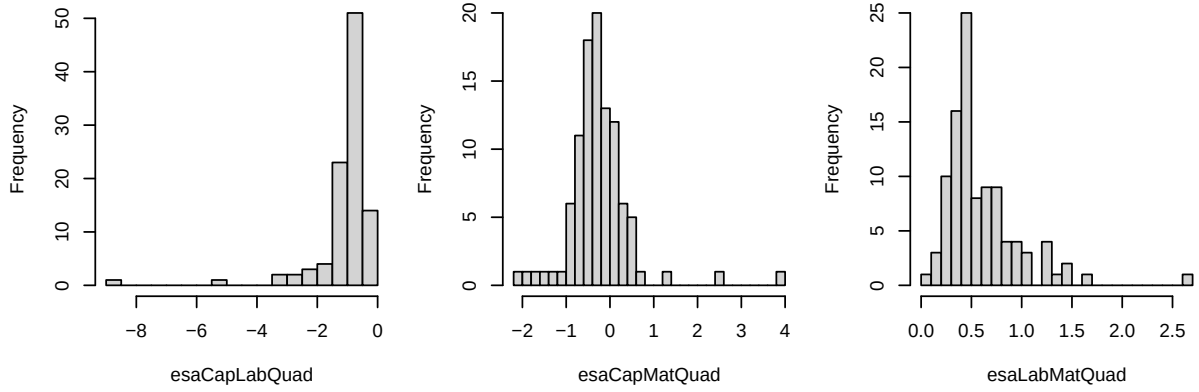


Figure 2.32: Quadratic production function: Allen elasticities of substitution

2.5.11.3 Comparison of direct and Allen elasticities of substitution

In the following, we use scatter plots to compare the estimated direct elasticities of substitution with the estimated Allen elasticities of substitution:

```
> compPlot( dat$esdCapLabQuad[ dat$monoQuad ],
+   dat$esaCapLabQuad[ dat$monoQuad ], lim = c( -4, 1 ) )
> compPlot( dat$esdCapMatQuad[ dat$monoQuad ],
+   dat$esaCapMatQuad[ dat$monoQuad ], lim = c( -10, 5 ) )
> compPlot( dat$esdLabMatQuad[ dat$monoQuad ],
+   dat$esaLabMatQuad[ dat$monoQuad ] )
```

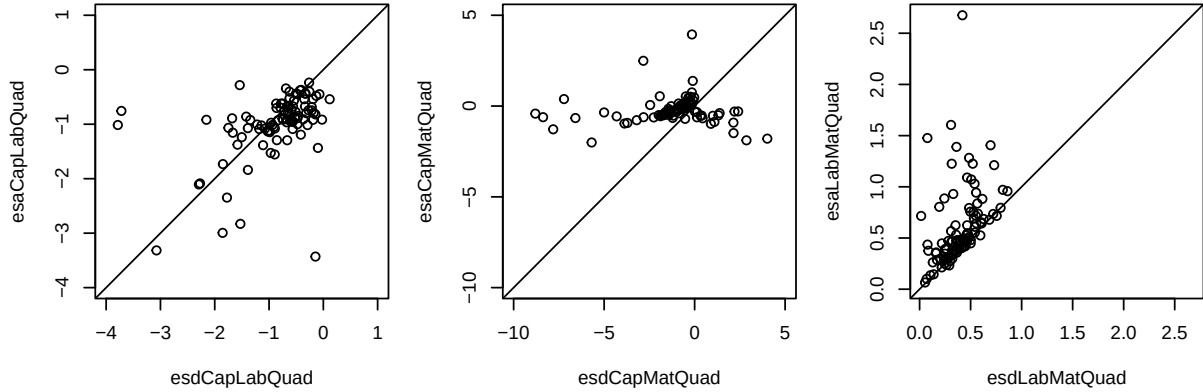


Figure 2.33: Quadratic production function: Comparison of direct and Allen elasticities of substitution

The resulting graphs are shown in figure 2.33. We focus on the elasticities of substitution between labor and materials, because the elasticities of substitution between capital and labor and the elasticities of substitution between capital and material cannot be meaningful interpreted (see section 2.5.11.1). In contrast to the direct elasticities of substitution, the Allen elasticities of substitution allow the other inputs to adjust. Therefore, it is not surprising that the Allen

elasticities of substitution between labor and materials are about the same or a little larger than the direct elasticities of substitution between labor and materials. However, even the Allen elasticities of substitution between labor and materials indicate a low substitutability between labor and materials (see also section 2.5.11.1).

2.5.12 First-order conditions for profit maximization

In this section, we will check to what extent the first-order conditions for profit maximization (2.40) are fulfilled, i.e. to what extent the firms use the optimal input quantities. We do this by comparing the marginal value products of the inputs with the corresponding input prices. We can calculate the marginal value products by multiplying the marginal products by the output price:

```
> dat$mvpCapQuad <- dat$pOut * dat$mpCapQuad
> dat$mvpLabQuad <- dat$pOut * dat$mpLabQuad
> dat$mvpMatQuad <- dat$pOut * dat$mpMatQuad
```

The command `compPlot` (package *miscTools*) can be used to compare the marginal value products with the corresponding input prices. As the logarithm of a non-positive number is not defined, we have to limit the comparisons on the logarithmic scale to observations with positive marginal products:

```
> compPlot( dat$pCap, dat$mvpCapQuad )
> compPlot( dat$pLab, dat$mvpLabQuad )
> compPlot( dat$pMat, dat$mvpMatQuad )
> compPlot( dat$pCap[ dat$monoQuad ], dat$mvpCapQuad[ dat$monoQuad ], log = "xy" )
> compPlot( dat$pLab[ dat$monoQuad ], dat$mvpLabQuad[ dat$monoQuad ], log = "xy" )
> compPlot( dat$pMat[ dat$monoQuad ], dat$mvpMatQuad[ dat$monoQuad ], log = "xy" )
```

The resulting graphs are shown in figure 2.34. They indicate that the marginal value products of most firms are higher than the corresponding input prices. This indicates that most firms could increase their profit by using more of all inputs. Given that the estimated quadratic function shows that (almost) all firms operate under increasing returns to scale, it is not surprising that most firms would gain from increasing all input quantities. Therefore, the question arises why the firms in the sample did not do this. This questions has already been addressed in section 2.3.10.

2.5.13 First-order conditions for cost minimization

As the marginal rates of technical substitution differ between observations for the three other functional forms, we use scatter plots for visualizing the comparison of the input price ratios with the negative inverse marginal rates of technical substitution. As the marginal rates of technical substitution are meaningless if the monotonicity condition is not fulfilled, we limit the comparisons to the observations, where all monotonicity conditions are fulfilled:

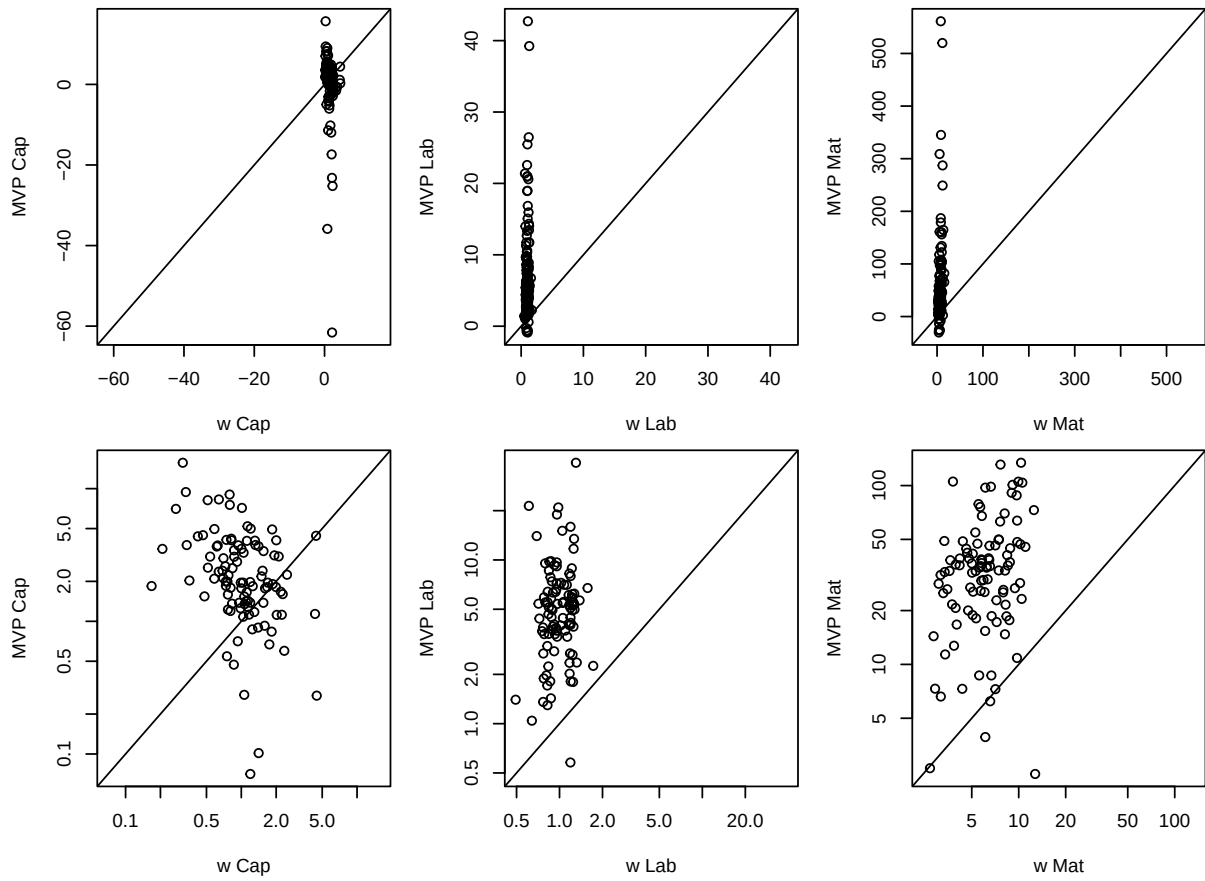


Figure 2.34: Marginal value products and corresponding input prices


```
> compPlot( ( dat$pCap / dat$pLab )[ dat$monoQuad ],
+   - dat$mrtsLabCapQuad[ dat$monoQuad ] )
> compPlot( ( dat$pCap / dat$pMat )[ dat$monoQuad ],
+   - dat$mrtsMatCapQuad[ dat$monoQuad ] )
> compPlot( ( dat$pLab / dat$pMat )[ dat$monoQuad ],
+   - dat$mrtsMatLabQuad[ dat$monoQuad ] )
> compPlot( ( dat$pCap / dat$pLab )[ dat$monoQuad ],
+   - dat$mrtsLabCapQuad[ dat$monoQuad ], log = "xy" )
> compPlot( ( dat$pCap / dat$pMat )[ dat$monoQuad ],
+   - dat$mrtsMatCapQuad[ dat$monoQuad ], log = "xy" )
> compPlot( ( dat$pLab / dat$pMat )[ dat$monoQuad ],
+   - dat$mrtsMatLabQuad[ dat$monoQuad ], log = "xy" )
```

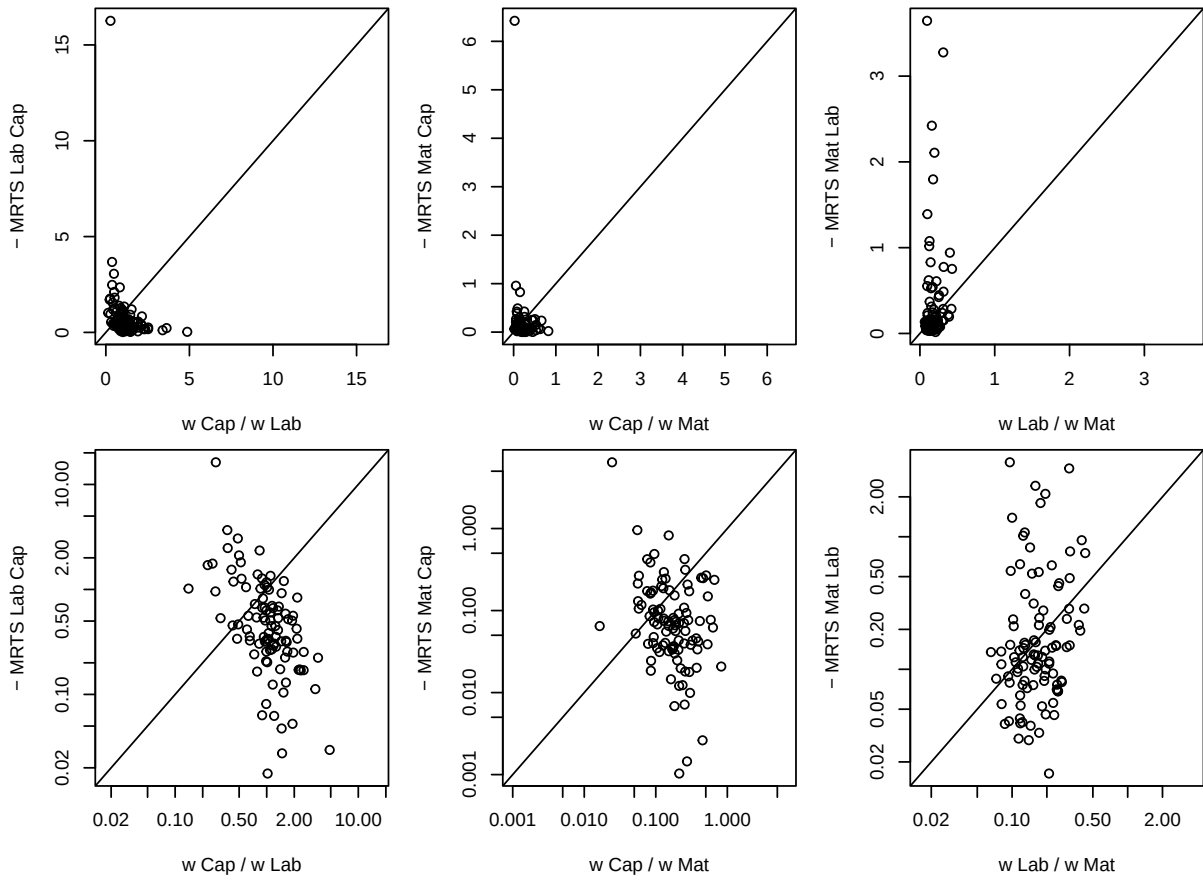


Figure 2.35: First-order conditions for costs minimization

The resulting graphs are shown in figure 2.35.

Furthermore, we use histograms to visualize the (absolute and relative) differences between the input price ratios and the corresponding negative inverse marginal rates of technical substitution:

```
> hist( ( - dat$mrtsLabCapQuad - dat$pCap / dat$pLab )[ dat$monoQuad ] )
```

```
> hist( ( - dat$mrtsMatCapQuad - dat$pCap / dat$pMat ) [ dat$monoQuad ] )
> hist( ( - dat$mrtsMatLabQuad - dat$pLab / dat$pMat ) [ dat$monoQuad ] )
> hist( log( - dat$mrtsLabCapQuad / ( dat$pCap / dat$pLab ) ) [ dat$monoQuad ] )
> hist( log( - dat$mrtsMatCapQuad / ( dat$pCap / dat$pMat ) ) [ dat$monoQuad ] )
> hist( log( - dat$mrtsMatLabQuad / ( dat$pLab / dat$pMat ) ) [ dat$monoQuad ] )
```

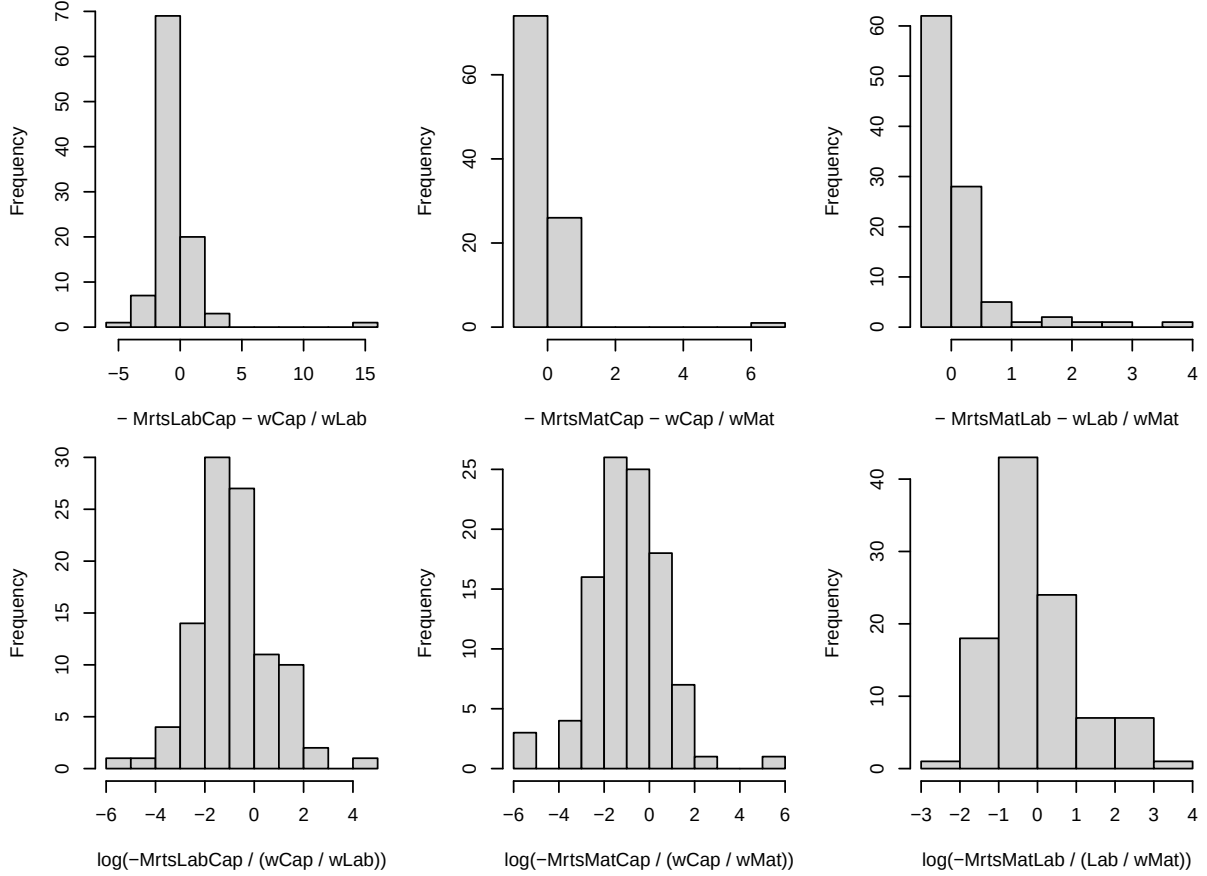


Figure 2.36: First-order conditions for costs minimization

The resulting graphs are shown in figure 2.36. The left graphs in figures 2.35 and 2.36 show that the ratio between the capital price and the labor price is larger than the absolute value of the marginal rate of technical substitution between labor and capital for a majority of the firms in the sample:

$$\frac{w_{\text{cap}}}{w_{\text{lab}}} > -MRTS_{\text{lab}, \text{cap}} = \frac{MP_{\text{cap}}}{MP_{\text{lab}}} \quad (2.133)$$

Hence, these firms can get closer to the minimum of their production costs by substituting labor for capital, because this will decrease the marginal product of labor and increase the marginal product of capital so that the absolute value of the MRTS between labor and capital increases and gets closer to the corresponding input price ratio. Similarly, the graphs in the middle column indicate that a majority of the firms should substitute materials for capital and the graphs on

the right indicate that a little more than half of the firms should substitute materials for labor. Hence, the majority of the firms could reduce production costs particularly by using less capital and using more labor or more materials. ⁶

2.6 Translog production function

2.6.1 Specification

The Translog function is a more flexible extension of the Cobb-Douglas function as the quadratic function is a more flexible extension of the linear function. Hence, the Translog function can be seen as a combination of the Cobb-Douglas function and the quadratic function. The Translog production function has following specification:

$$\ln y = \alpha_0 + \sum_i \alpha_i \ln x_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij} \ln x_i \ln x_j \quad (2.134)$$

with $\alpha_{ij} = \alpha_{ji}$.

2.6.2 Estimation

We can estimate this Translog production function with the command

```
> prodTL <- lm( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat )
+      + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qLab )^2 )
+      + I( 0.5 * log( qMat )^2 ) + I( log( qCap ) * log( qLab ) )
+      + I( log( qCap ) * log( qMat ) ) + I( log( qLab ) * log( qMat ) ),
+      data = dat )
> summary( prodTL )
```

Call:

```
lm(formula = log(qOut) ~ log(qCap) + log(qLab) + log(qMat) +
    I(0.5 * log(qCap)^2) + I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) +
    I(log(qCap) * log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) *
    log(qMat)), data = dat)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.68015	-0.36688	0.05389	0.44125	1.26560

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
--	----------	------------	---------	----------

⁶This generally confirms the results of the Cobb-Douglas production function.

2 Primal Approach: Production Function

```
(Intercept)          -4.14581    21.35945   -0.194    0.8464
log(qCap)             -2.30683     2.28829   -1.008    0.3153
log(qLab)              1.99328     4.56624    0.437    0.6632
log(qMat)              2.23170     3.76334    0.593    0.5542
I(0.5 * log(qCap)^2)  -0.02573     0.20834   -0.124    0.9019
I(0.5 * log(qLab)^2)  -1.16364     0.67943   -1.713    0.0892 .
I(0.5 * log(qMat)^2)  -0.50368     0.43498   -1.158    0.2490
I(log(qCap) * log(qLab)) 0.56194     0.29120    1.930    0.0558 .
I(log(qCap) * log(qMat)) -0.40996     0.23534   -1.742    0.0839 .
I(log(qLab) * log(qMat)) 0.65793     0.42750    1.539    0.1262
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6412 on 130 degrees of freedom

Multiple R-squared: 0.6296, Adjusted R-squared: 0.6039

F-statistic: 24.55 on 9 and 130 DF, p-value: < 2.2e-16

None of the estimated coefficients is statistically significantly different from zero at the 5% significance level and only three coefficients are statistically significant at the 10% level. As the Cobb-Douglas production function is “nested” in the Translog production function, we can apply a “Wald test” or “likelihood ratio test” to check whether the Cobb-Douglas production function is rejected in favor of the Translog production function. This can be done by the functions `waldtest` and `lrtest` (package *lmtest*):

```
> waldtest( prodCD, prodTL )
```

Wald test

Model 1: $\log(q_{\text{Out}}) \sim \log(q_{\text{Cap}}) + \log(q_{\text{Lab}}) + \log(q_{\text{Mat}})$

Model 2: $\log(q_{\text{Out}}) \sim \log(q_{\text{Cap}}) + \log(q_{\text{Lab}}) + \log(q_{\text{Mat}}) + I(0.5 * \log(q_{\text{Cap}})^2) + I(0.5 * \log(q_{\text{Lab}})^2) + I(0.5 * \log(q_{\text{Mat}})^2) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Lab}})) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Mat}})) + I(\log(q_{\text{Lab}}) * \log(q_{\text{Mat}}))$

```
Res.Df Df      F Pr(>F)
```

```
1      136
```

```
2      130  6 2.062 0.06202 .
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( prodCD, prodTL )
```

Likelihood ratio test

```

Model 1: log(qOut) ~ log(qCap) + log(qLab) + log(qMat)
Model 2: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
  I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
  log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))
#Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -137.61
2   11 -131.25  6 12.727    0.04757 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

At the 5% significance level, the Cobb-Douglas production function is accepted by the Wald test but rejected in favor of the Translog production function by the likelihood ratio test. In order to reduce the chance of using a too restrictive functional form, we proceed with the Translog production function.

2.6.3 Statistical significance of individual inputs

The following code applies Wald tests (using R functions `linearHypothesis` and `waldtest`) and Likelihood Ratio tests (using the R function `lrtest`) to test whether each of the three input quantities significantly affects the output quantity in the estimated Translog production function.

```

> library( "car" )
> # capital
> linearHypothesis( prodTL, c( "log(qCap) = 0", "I(0.5 * log(qCap)^2) = 0",
+   "I(log(qCap) * log(qLab)) = 0", "I(log(qCap) * log(qMat)) = 0" ) )

```

Linear hypothesis test:

```

log(qCap) = 0
I(0.5 * log(qCap)^2) = 0
I(log(qCap) * log(qLab)) = 0
I(log(qCap) * log(qMat)) = 0

```

Model 1: restricted model

```

Model 2: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
  I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
  log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))

```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	134	57.039				
2	130	53.447	4	3.5916	2.184	0.0743 .

2 Primal Approach: Production Function

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> waldtest( prodTL, log( qOut ) ~ log( qLab ) + log( qMat )
+   + I( 0.5 * log( qLab )^2 ) + I( 0.5 * log( qMat )^2 )
+   + I( log( qLab ) * log( qMat ) ) )
```

Wald test

```
Model 1: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
  I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
  log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))
Model 2: log(qOut) ~ log(qLab) + log(qMat) + I(0.5 * log(qLab)^2) + I(0.5 *
  log(qMat)^2) + I(log(qLab) * log(qMat))
```

	Res.Df	Df	F	Pr(>F)
1	130			
2	134	-4	2.184	0.0743 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( prodTL, log( qOut ) ~ log( qLab ) + log( qMat )
+   + I( 0.5 * log( qLab )^2 ) + I( 0.5 * log( qMat )^2 )
+
+   + I( log( qLab ) * log( qMat ) ) )
```

Likelihood ratio test

```
Model 1: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
  I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
  log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))
Model 2: log(qOut) ~ log(qLab) + log(qMat) + I(0.5 * log(qLab)^2) + I(0.5 *
  log(qMat)^2) + I(log(qLab) * log(qMat))
```

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	11	-131.25			
2	7	-135.80	-4	9.1052	0.05852 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> # labor
> linearHypothesis( prodTL, c( "log(qLab) = 0", "I(0.5 * log(qLab)^2) = 0",
+   "I(log(qCap) * log(qLab)) = 0", "I(log(qLab) * log(qMat)) = 0" ) )
```

Linear hypothesis test:

$\log(q_{\text{Lab}}) = 0$

$I(0.5 * \log(q_{\text{Lab}})^2) = 0$

$I(\log(q_{\text{Cap}}) * \log(q_{\text{Lab}})) = 0$

$I(\log(q_{\text{Lab}}) * \log(q_{\text{Mat}})) = 0$

Model 1: restricted model

Model 2: $\log(q_{\text{Out}}) \sim \log(q_{\text{Cap}}) + \log(q_{\text{Lab}}) + \log(q_{\text{Mat}}) + I(0.5 * \log(q_{\text{Cap}})^2) + I(0.5 * \log(q_{\text{Lab}})^2) + I(0.5 * \log(q_{\text{Mat}})^2) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Lab}})) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Mat}})) + I(\log(q_{\text{Lab}}) * \log(q_{\text{Mat}}))$

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	134	66.649				
2	130	53.447	4	13.202	8.028	8.14e-06 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> waldtest( prodTL, log( qOut ) ~ log( qCap ) + log( qMat )
+ + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qMat )^2 )
+ + I( log( qCap ) * log( qMat ) ) )
```

Wald test

Model 1: $\log(q_{\text{Out}}) \sim \log(q_{\text{Cap}}) + \log(q_{\text{Lab}}) + \log(q_{\text{Mat}}) + I(0.5 * \log(q_{\text{Cap}})^2) + I(0.5 * \log(q_{\text{Lab}})^2) + I(0.5 * \log(q_{\text{Mat}})^2) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Lab}})) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Mat}})) + I(\log(q_{\text{Lab}}) * \log(q_{\text{Mat}}))$

Model 2: $\log(q_{\text{Out}}) \sim \log(q_{\text{Cap}}) + \log(q_{\text{Mat}}) + I(0.5 * \log(q_{\text{Cap}})^2) + I(0.5 * \log(q_{\text{Mat}})^2) + I(\log(q_{\text{Cap}}) * \log(q_{\text{Mat}}))$

	Res.Df	Df	F	Pr(>F)
1	130			
2	134	-4	8.028	8.14e-06 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( prodTL, log( qOut ) ~ log( qCap ) + log( qMat )
+ + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qMat )^2 )
+ + I( log( qCap ) * log( qMat ) ) )
```

Likelihood ratio test

Model 1: $\log(q_{\text{Out}}) \sim \log(q_{\text{Cap}}) + \log(q_{\text{Lab}}) + \log(q_{\text{Mat}}) + I(0.5 * \log(q_{\text{Cap}})^2) +$

2 Primal Approach: Production Function

```

I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))
Model 2: log(qOut) ~ log(qCap) + log(qMat) + I(0.5 * log(qCap)^2) + I(0.5 *
log(qMat)^2) + I(log(qCap) * log(qMat))
#Df LogLik Df Chisq Pr(>Chisq)
1 11 -131.25
2 7 -146.70 -4 30.905 3.201e-06 ***

```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

> # materials
> linearHypothesis( prodTL, c( "log(qMat) = 0", "I(0.5 * log(qMat)^2) = 0",
+ "I(log(qCap) * log(qMat)) = 0", "I(log(qLab) * log(qMat)) = 0" ) )

```

Linear hypothesis test:

```

log(qMat) = 0
I(0.5 * log(qMat)^2) = 0
I(log(qCap) * log(qMat)) = 0
I(log(qLab) * log(qMat)) = 0

```

Model 1: restricted model

```

Model 2: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))

```

```

Res.Df RSS Df Sum of Sq F Pr(>F)
1 134 68.178
2 130 53.447 4 14.731 8.9578 2.02e-06 ***

```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

> waldtest( prodTL, log( qOut ) ~ log( qCap ) + log( qLab )
+ + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qLab )^2 )
+ + I( log( qCap ) * log( qLab ) ) )

```

Wald test

```

Model 1: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))
Model 2: log(qOut) ~ log(qCap) + log(qLab) + I(0.5 * log(qCap)^2) + I(0.5 *

```



```

log(qLab)^2) + I(log(qCap) * log(qLab))
Res.Df Df      F    Pr(>F)
1      130
2      134 -4 8.9578 2.02e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> lrtest( prodTL, log( qOut ) ~ log( qCap ) + log( qLab )
+   + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qLab )^2 )
+   + I( log( qCap ) * log( qLab ) ) )

Likelihood ratio test

Model 1: log(qOut) ~ log(qCap) + log(qLab) + log(qMat) + I(0.5 * log(qCap)^2) +
  I(0.5 * log(qLab)^2) + I(0.5 * log(qMat)^2) + I(log(qCap) *
  log(qLab)) + I(log(qCap) * log(qMat)) + I(log(qLab) * log(qMat))
Model 2: log(qOut) ~ log(qCap) + log(qLab) + I(0.5 * log(qCap)^2) + I(0.5 *
  log(qLab)^2) + I(log(qCap) * log(qLab))
#Df  LogLik Df  Chisq Pr(>Chisq)
1   11 -131.25
2    7 -148.28 -4 34.081 7.173e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

All inputs have a statistically significant effect at the 10% significance level, while labour and materials also have statistically significant effects at the 5% and 1% (and at even higher) significance levels.

2.6.4 Properties

We cannot see from the estimated coefficients whether the monotonicity condition is fulfilled. The Translog production function cannot be globally monotone, because there will be always a set of input quantities that result in negative marginal products.⁷ The Translog function would only be globally monotone, if all first-order coefficients are positive and all second-order coefficients are zero, which is equivalent to a Cobb-Douglas function. We will check the monotonicity condition at each observation in section 2.6.6.

All Translog production functions fulfill the weak and the strong essentiality assumption, because as soon as a single input quantity approaches zero, the right-hand side of equation (2.134) approaches minus infinity (if monotonicity is fulfilled), and thus, the output quantity $y = \exp(\ln y)$ approaches zero. Hence, if a data set includes observations with a positive output quantity but

⁷Please note that $\ln x_j$ is a large negative number if x_j is a very small positive number.

at least one input quantity that is zero, strict essentiality cannot be fulfilled in the underlying true production technology so that the Translog production function is not a suitable functional form for analyzing this data set.

The input requirement sets derived from Translog production functions are always closed and non-empty. The Translog production function always returns finite, real, non-negative, and single values as long as all input quantities are strictly positive. All Translog production functions are continuous and twice-continuously differentiable.

2.6.5 Predicted output quantities

As before, we can easily obtain the predicted output quantities with the `fitted` method. As we used the logarithmic output quantity as dependent variable in our estimated model, we must use the exponential function to obtain the output quantities measured in levels:

```
> dat$qOutTL <- exp( fitted( prodTL ) )
```

Now, we can evaluate the “fit” of the model by comparing the observed with the fitted output quantities:

```
> compPlot( dat$qOut, dat$qOutTL )
> compPlot( dat$qOut, dat$qOutTL, log = "xy" )
```

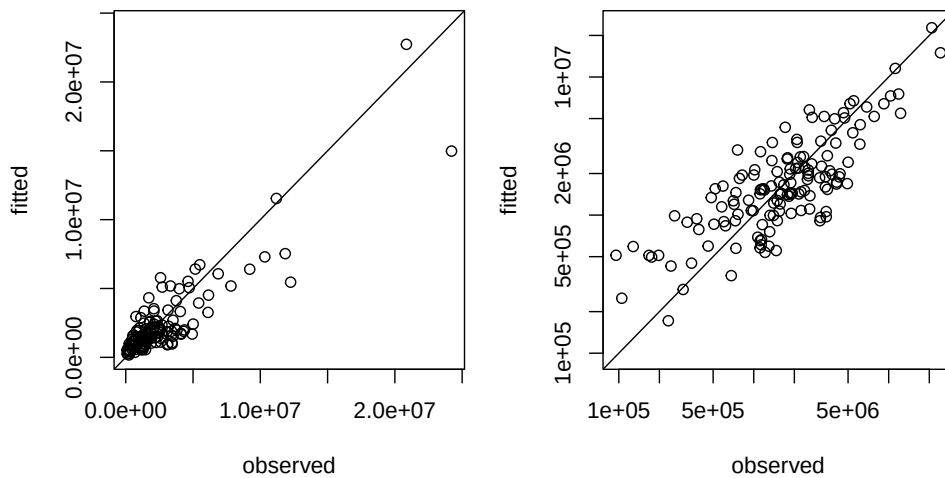


Figure 2.37: Translog production function: fit of the model

The resulting graphs are shown in figure 2.37. While the graph in the left panel uses a linear scale for the axes, the graph in the right panel uses a logarithmic scale for both axes. Hence, the deviations from the 45°-line illustrate the *absolute* deviations in the left panel and the *relative* deviations in the right panel. The fit of the model looks rather okay, but there are some observations, at which the predicted output quantity is not very close to the observed output quantity.

2.6.6 Output elasticities

The output elasticities calculated from a Translog production function are:

$$\epsilon_i = \frac{\partial \ln y}{\partial \ln x_i} = \alpha_i + \sum_j \alpha_{ij} \ln x_j \quad (2.135)$$

We can simplify the code for computing these output elasticities by using short names for the coefficients:

```
> a1 <- coef( prodTL )[ "log(qCap)" ]
> a2 <- coef( prodTL )[ "log(qLab)" ]
> a3 <- coef( prodTL )[ "log(qMat)" ]
> a11 <- coef( prodTL )[ "I(0.5 * log(qCap)^2)" ]
> a22 <- coef( prodTL )[ "I(0.5 * log(qLab)^2)" ]
> a33 <- coef( prodTL )[ "I(0.5 * log(qMat)^2)" ]
> a12 <- a21 <- coef( prodTL )[ "I(log(qCap) * log(qLab))" ]
> a13 <- a31 <- coef( prodTL )[ "I(log(qCap) * log(qMat))" ]
> a23 <- a32 <- coef( prodTL )[ "I(log(qLab) * log(qMat))" ]
```

Now, we can use the following commands to calculate the output elasticities in R:

```
> dat$eCapTL <- with( dat,
+   a1 + a11 * log(qCap) + a12 * log(qLab) + a13 * log(qMat) )
> dat$eLabTL <- with( dat,
+   a2 + a21 * log(qCap) + a22 * log(qLab) + a23 * log(qMat) )
> dat$eMatTL <- with( dat,
+   a3 + a31 * log(qCap) + a32 * log(qLab) + a33 * log(qMat) )
```

We can visualize (the variation of) these output elasticities with histograms:

```
> hist( dat$eCapTL, 15 )
> hist( dat$eLabTL, 15 )
> hist( dat$eMatTL, 15 )
```

The resulting graphs are shown in figure 2.38. If the firms increase capital input by one percent, the output of most firms will increase by around 0.2 percent. If the firms increase labor input by one percent, the output of most firms will increase by around 0.5 percent. If the firms increase material input by one percent, the output of most firms will increase by around 0.7 percent. These graphs also show that the monotonicity condition is not fulfilled for all observations:

```
> table( dat$eCapTL >= 0 )
```

```
FALSE  TRUE
   32   108
```

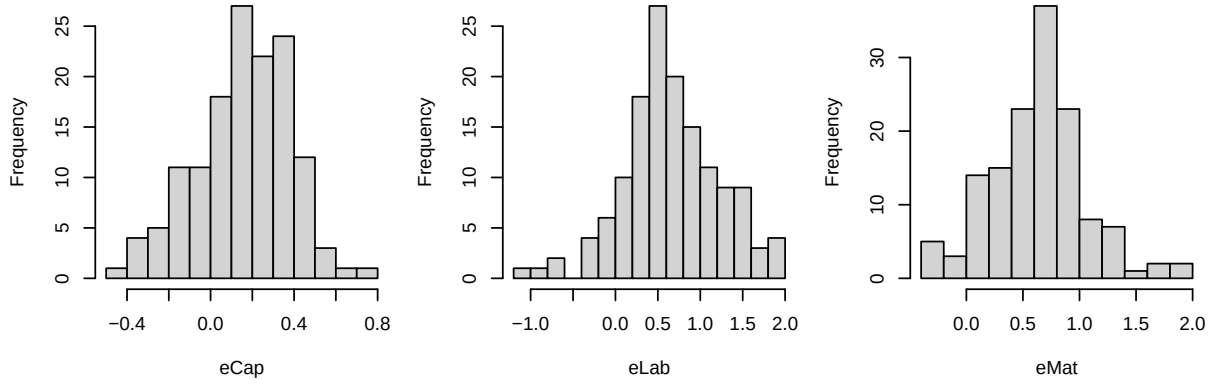


Figure 2.38: Translog production function: output elasticities

```
> table( dat$eLabTL >= 0 )
```

```
FALSE  TRUE
    14   126
```

```
> table( dat$eMatTL >= 0 )
```

```
FALSE  TRUE
     8   132
```

```
> dat$monoTL <- with( dat, eCapTL >= 0 & eLabTL >= 0 & eMatTL >= 0 )
```

```
> table( dat$monoTL )
```

```
FALSE  TRUE
    48    92
```

32 firms have a negative output elasticity of capital, 14 firms have a negative output elasticity of labor, and 8 firms have a negative output elasticity of materials. In total the monotonicity condition is not fulfilled at 48 out of 140 observations. Although the monotonicity conditions are fulfilled for a large part of firms in our data set, these frequent violations indicate a possible model misspecification.

2.6.7 Marginal products

The first derivatives (marginal products) of the Translog production function with respect to the input quantities are:

$$MP_i = \frac{\partial y}{\partial x_i} = \frac{y}{x_i} \frac{\partial \ln y}{\partial \ln x_i} = \frac{y}{x_i} \left(\alpha_i + \sum_j \alpha_{ij} \ln x_j \right) \quad (2.136)$$

We can calculate the marginal products based on the output elasticities that we have calculated above. As argued in section 2.4.11.1, we use the predicted output quantities in this calculation:

```
> dat$mpCapTL <- with( dat, eCapTL * qOutTL / qCap )
> dat$mpLabTL <- with( dat, eLabTL * qOutTL / qLab )
> dat$mpMatTL <- with( dat, eMatTL * qOutTL / qMat )
```

We can visualize (the variation of) these marginal products with histograms:

```
> hist( dat$mpCapTL, 15 )
> hist( dat$mpLabTL, 15 )
> hist( dat$mpMatTL, 15 )
```

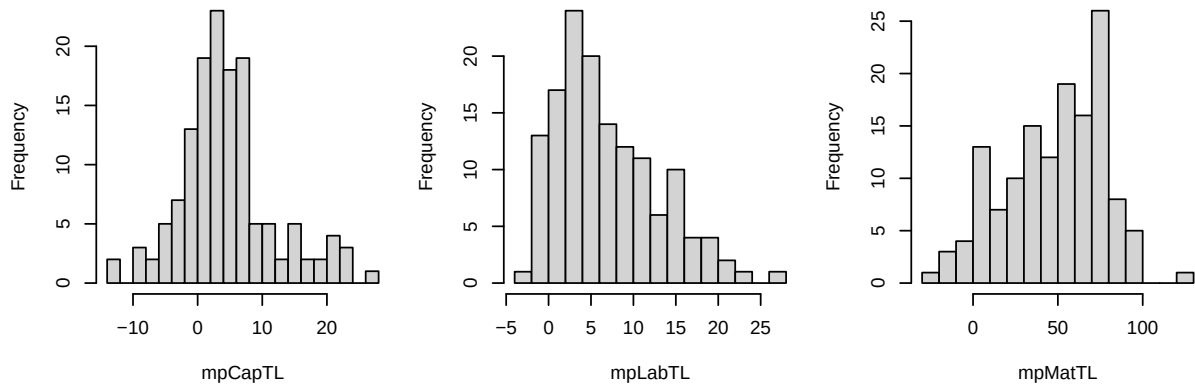


Figure 2.39: Translog production function: marginal products

The resulting graphs are shown in figure 2.39. If the firms increase capital input by one unit, the output of most firms will increase by around 4 units. If the firms increase labor input by one unit, the output of most firms will increase by around 4 units. If the firms increase material input by one unit, the output of most firms will increase by around 70 units.

2.6.8 Elasticity of scale

The elasticity of scale can—as always—be calculated as the sum of all output elasticities.

```
> dat$eScaleTL <- dat$eCapTL + dat$eLabTL +
+   dat$eMatTL
```

The (variation of the) elasticities of scale can be visualized with a histogram.

```
> hist( dat$eScaleTL, 30 )
> hist( dat$eScaleTL[ dat$monoTL ], 30 )
```

The resulting graphs are shown in figure 2.40. All firms experience increasing returns to scale and most of them have an elasticity of scale around 1.45. Hence, if these firms increase all input quantities by one percent, the output of most firms will increase by around 1.45 percent. These elasticities of scale are realistic and on average close to the elasticity of scale obtained from the Cobb-Douglas production function (1.47).

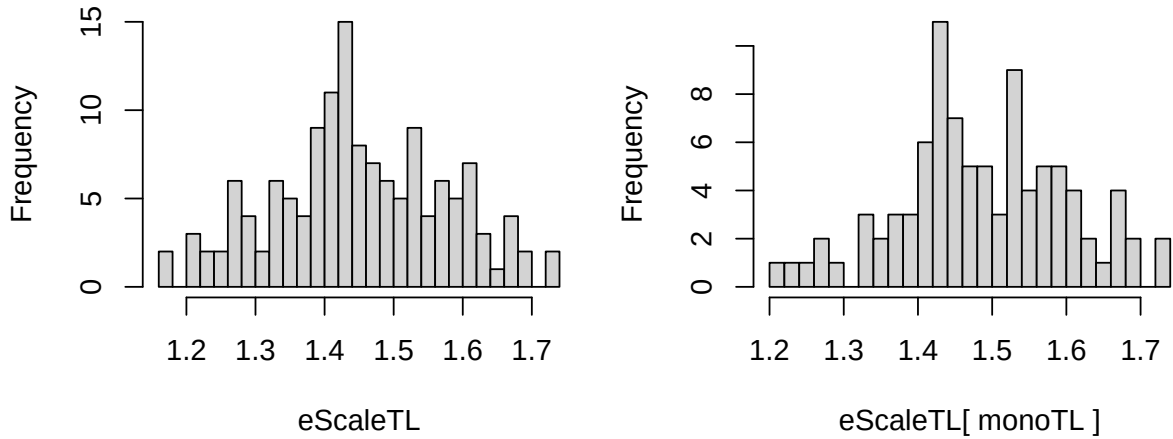


Figure 2.40: Translog production function: elasticities of scale

Information on the optimal firm size can be obtained by analyzing the relationship between firm size and the elasticity of scale. We can either use the observed or the predicted output:

```
> plot( dat$qOut, dat$eScaleTL, log = "x" )
> plot( dat$X, dat$eScaleTL, log = "x" )
> plot( dat$qOut[ dat$monoTL ], dat$eScaleTL[ dat$monoTL ], log = "x" )
> plot( dat$X[ dat$monoTL ], dat$eScaleTL[ dat$monoTL ], log = "x" )
```

The resulting graphs are shown in figure 2.41. Both of them indicate that the elasticity of scale slightly decreases with firm size but there are considerable increasing returns to scale even for the largest firms in the sample. Hence, all firms in the sample would gain from increasing their size and the optimal firm size seems to be larger than the largest firm in the sample.

2.6.9 Marginal rates of technical substitution

We can calculate the marginal rates of technical substitution (MRTS) based on our estimated Translog production function by following commands:

```
> dat$mrtsCapLabTL <- with( dat, - mpLabTL / mpCapTL )
> dat$mrtsLabCapTL <- with( dat, - mpCapTL / mpLabTL )
> dat$mrtsCapMatTL <- with( dat, - mpMatTL / mpCapTL )
> dat$mrtsMatCapTL <- with( dat, - mpCapTL / mpMatTL )
> dat$mrtsLabMatTL <- with( dat, - mpMatTL / mpLabTL )
> dat$mrtsMatLabTL <- with( dat, - mpLabTL / mpMatTL )
```

As the marginal rates of technical substitution are meaningless if the monotonicity condition is not fulfilled, we visualize (the variation of) these MRTS only for the observations, where the monotonicity condition is fulfilled:

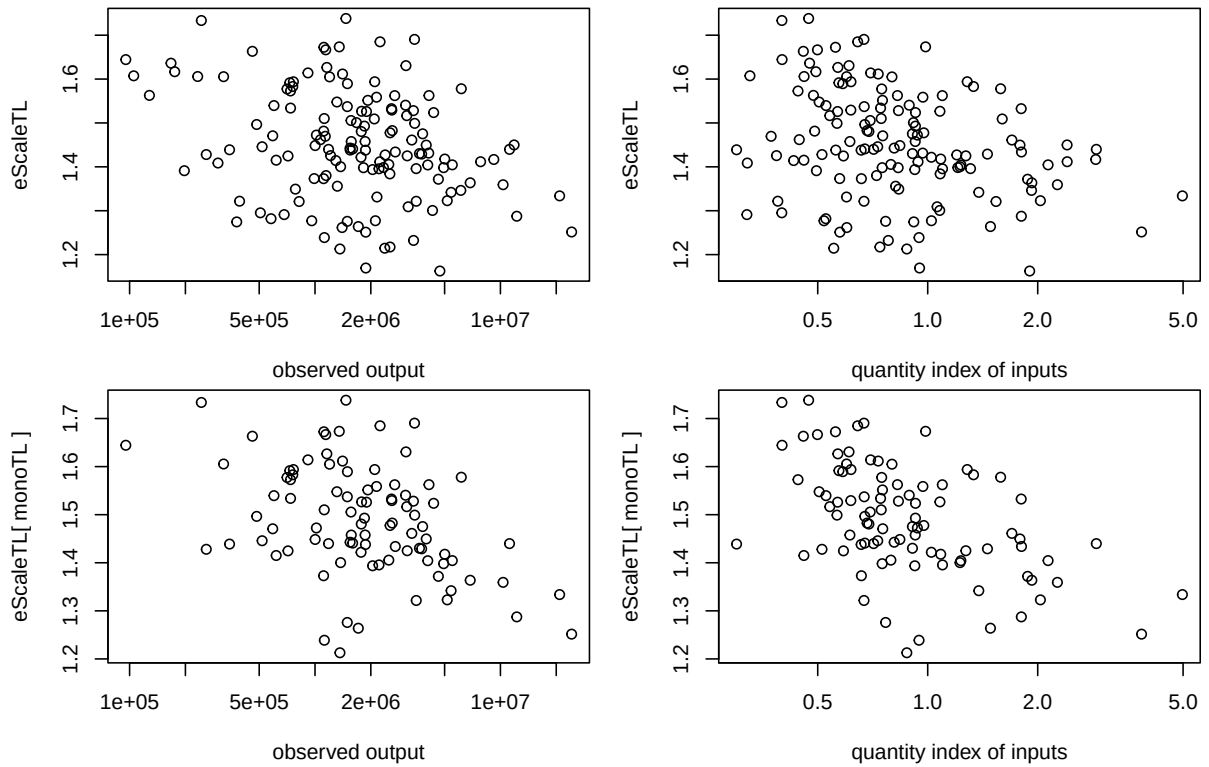


Figure 2.41: Translog production function: elasticities of scale at different firm sizes

```
> hist( dat$mrtsCapLabTL[ dat$monoTL ], 30 )
> hist( dat$mrtsLabCapTL[ dat$monoTL ], 30 )
> hist( dat$mrtsCapMatTL[ dat$monoTL ], 30 )
> hist( dat$mrtsMatCapTL[ dat$monoTL ], 30 )
> hist( dat$mrtsLabMatTL[ dat$monoTL ], 30 )
> hist( dat$mrtsMatLabTL[ dat$monoTL ], 30 )
```

The resulting graphs are shown in figure 2.43. As some outliers hide the variation of the majority of the MRTS, we use function `colMedians` (package *miscTools*) to show the median values of the MRTS:

```
> colMedians( subset( dat, monoTL,
+   c( "mrtsCapLabTL", "mrtsLabCapTL", "mrtsCapMatTL",
+     "mrtsMatCapTL", "mrtsLabMatTL", "mrtsMatLabTL" ) ) )
```

```
mrtsCapLabTL mrtsLabCapTL mrtsCapMatTL mrtsMatCapTL mrtsLabMatTL mrtsMatLabTL
-0.83929283 -1.19196521 -12.72554396 -0.07858435 -12.79850828 -0.07813810
```

Given that the median marginal rate of technical substitution between capital and labor is -0.84, a typical firm that reduces the use of labor by one unit, has to use around 0.84 additional units of capital in order to produce the same amount of output as before. Alternatively, the typical firm can replace one unit of labor by using 0.08 additional units of materials.

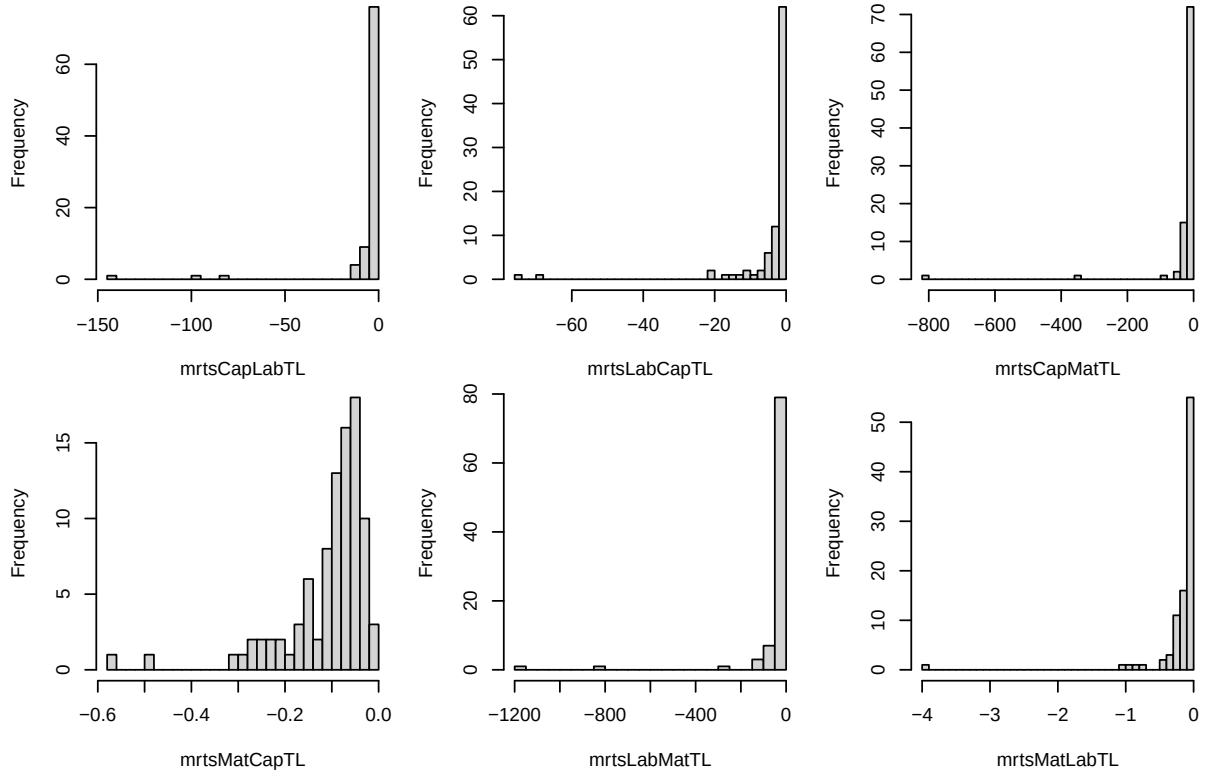


Figure 2.42: Translog production function: marginal rates of technical substitution (MRTS)

2.6.10 Relative marginal rates of technical substitution

As we do not have a practical interpretation of the units of measurement of the input quantities, the relative marginal rates of technical substitution (RMRTS) are practically more meaningful than the MRTS. The following commands calculate the RMRTS:

```
> dat$rmrtsCapLabTL <- with( dat, - eLabTL / eCapTL )
> dat$rmrtsLabCapTL <- with( dat, - eCapTL / eLabTL )
> dat$rmrtsCapMatTL <- with( dat, - eMatTL / eCapTL )
> dat$rmrtsMatCapTL <- with( dat, - eCapTL / eMatTL )
> dat$rmrtsLabMatTL <- with( dat, - eMatTL / eLabTL )
> dat$rmrtsMatLabTL <- with( dat, - eLabTL / eMatTL )
```

As the (relative) marginal rates of technical substitution are meaningless if the monotonicity condition is not fulfilled, we visualize (the variation of) these RMRTS only for the observations, where the monotonicity condition is fulfilled:

```
> hist( dat$rmrtsCapLabTL[ dat$monoTL ], 30 )
> hist( dat$rmrtsLabCapTL[ dat$monoTL ], 30 )
> hist( dat$rmrtsCapMatTL[ dat$monoTL ], 30 )
> hist( dat$rmrtsMatCapTL[ dat$monoTL ], 30 )
```



```
> hist( dat$rmrtsLabMatTL[ dat$monoTL ], 30 )
> hist( dat$rmrtsMatLabTL[ dat$monoTL ], 30 )
```

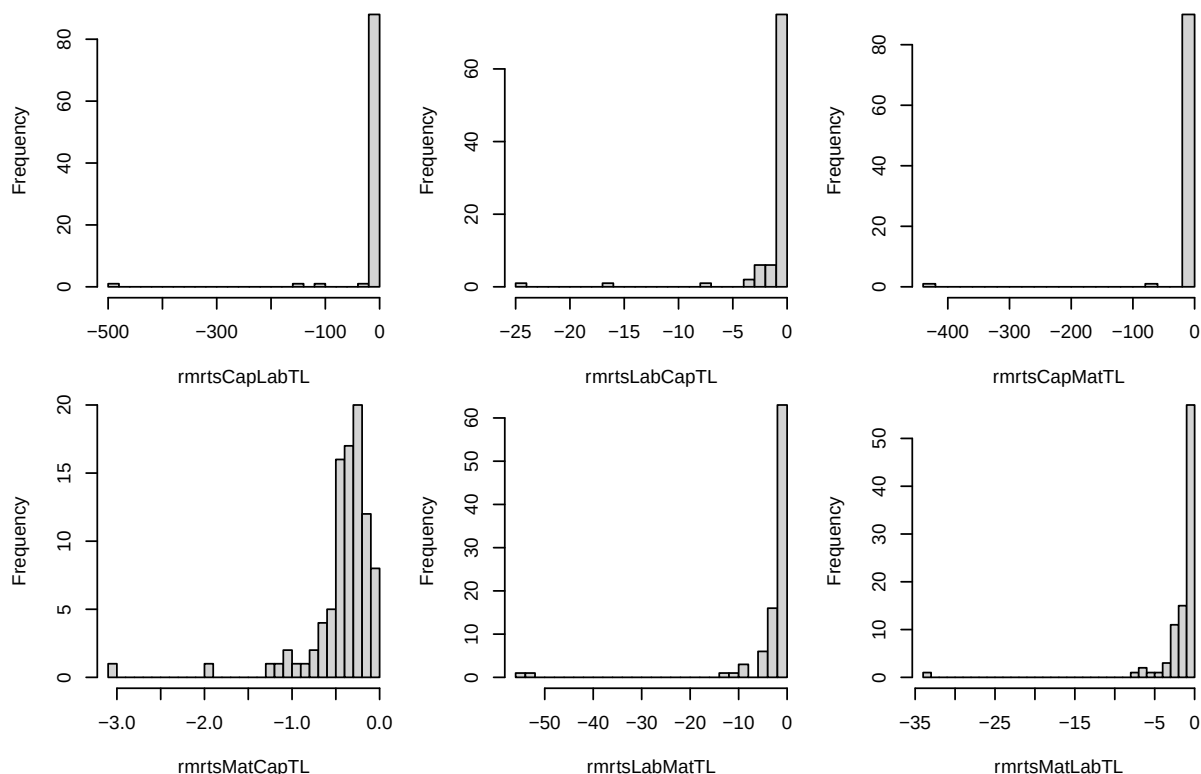


Figure 2.43: Translog production function: relative marginal rates of technical substitution (RMRTS)

The resulting graphs are shown in figure 2.43. As some outliers hide the variation of the majority of the RMRTS, we use function `colMedians` (package *miscTools*) to show the median values of the RMRTS:

```
> colMedians( subset( dat, monoTL,
+   c( "rmrtsCapLabTL", "rmrtsLabCapTL", "rmrtsCapMatTL",
+     "rmrtsMatCapTL", "rmrtsLabMatTL", "rmrtsMatLabTL" ) ) )

rmrtsCapLabTL rmrtsLabCapTL rmrtsCapMatTL rmrtsMatCapTL rmrtsLabMatTL
-2.8357239    -0.3539150    -3.0064237    -0.3331325    -1.3444115
rmrtsMatLabTL
-0.7439008
```

Given that the median relative marginal rate of technical substitution between capital and labor is -2.84, a typical firm that reduces the use of labor by one percent, has to use around 2.84 percent more capital in order to produce the same amount of output as before. Alternatively, the typical firm can replace one percent of labor by using 0.74 percent more materials.

2.6.11 Second partial derivatives

In order to compute the elasticities of substitution, we need obtain the second derivatives of the Translog function. We can calculate them as derivatives of the first derivatives of the Translog function:

$$\frac{\partial^2 y}{\partial x_i \partial x_j} = \frac{\partial \left(\frac{\partial y}{\partial x_i} \right)}{\partial x_j} = \frac{\partial \left((\alpha_i + \sum_k \alpha_{ik} \ln x_k) \frac{y}{x_i} \right)}{\partial x_j} \quad (2.137)$$

$$= \frac{\alpha_{ij}}{x_j} \frac{y}{x_i} + \frac{\alpha_i + \sum_k \alpha_{ik} \ln x_k}{x_i} \frac{\partial y}{\partial x_j} - \delta_{ij} \left(\alpha_i + \sum_k \alpha_{ik} \ln x_k \right) \frac{y}{x_i^2} \quad (2.138)$$

$$= \frac{\alpha_{ij}}{x_i} \frac{y}{x_j} + \frac{\alpha_i + \sum_k \alpha_{ik} \ln x_k}{x_i} \left(\alpha_j + \sum_k \alpha_{jk} \ln x_k \right) \frac{y}{x_j} - \delta_{ij} \left(\alpha_i + \sum_k \alpha_{ik} \ln x_k \right) \frac{y}{x_i^2} \quad (2.139)$$

$$= \frac{\alpha_{ij}}{x_i} \frac{y}{x_j} + \frac{\epsilon_i}{x_i} \frac{\epsilon_j}{x_j} \frac{y}{x_j} - \delta_{ij} \frac{\epsilon_i}{x_i^2} \frac{y}{x_j} \quad (2.140)$$

$$= \frac{y}{x_i x_j} (\alpha_{ij} + \epsilon_i \epsilon_j - \delta_{ij} \epsilon_i), \quad (2.141)$$

where δ_{ij} is (again) Kronecker's delta (2.95). Alternatively, the second derivatives of the Translog function can be expressed based on the marginal products (instead of the output elasticities):

$$\frac{\partial^2 y}{\partial x_i \partial x_j} = \frac{\alpha_{ij}}{x_i} \frac{y}{x_j} + \frac{MP_i}{y} \frac{MP_j}{x_j} - \delta_{ij} \frac{MP_i}{x_i} \quad (2.142)$$

Now, we can calculate the second derivatives for each observation in our data set:

```
> dat$fCapCapTL <- with( dat,
+   qOutTL / qCap^2 * ( a11 + eCapTL^2 - eCapTL ) )
> dat$fLabLabTL <- with( dat,
+   qOutTL / qLab^2 * ( a22 + eLabTL^2 - eLabTL ) )
> dat$fMatMatTL <- with( dat,
+   qOutTL / qMat^2 * ( a33 + eMatTL^2 - eMatTL ) )
> dat$fCapLabTL <- with( dat,
+   qOutTL / ( qCap * qLab ) * ( a12 + eCapTL * eLabTL ) )
> dat$fCapMatTL <- with( dat,
+   qOutTL / ( qCap * qMat ) * ( a13 + eCapTL * eMatTL ) )
> dat$fLabMatTL <- with( dat,
+   qOutTL / ( qLab * qMat ) * ( a23 + eLabTL * eMatTL ) )
```

2.6.12 Quasiconcavity

In order to check whether the estimated Translog production function is quasiconcave, we need to construct the bordered Hessian matrix. As the first-order and second-order derivatives of

the Translog production function differ between observations, also the bordered Hessian matrix differs between observations. As a starting point, we construct the bordered Hessian matrix just for the first observation:

```
> bhm <- matrix( 0, nrow = 4, ncol = 4 )
> bhm[ 1, 2 ] <- bhm[ 2, 1 ] <- dat$mpCapTL[ 1 ]
> bhm[ 1, 3 ] <- bhm[ 3, 1 ] <- dat$mpLabTL[ 1 ]
> bhm[ 1, 4 ] <- bhm[ 4, 1 ] <- dat$mpMatTL[ 1 ]
> bhm[ 2, 2 ] <- dat$fCapCapTL[ 1 ]
> bhm[ 3, 3 ] <- dat$fLabLabTL[ 1 ]
> bhm[ 4, 4 ] <- dat$fMatMatTL[ 1 ]
> bhm[ 2, 3 ] <- bhm[ 3, 2 ] <- dat$fCapLabTL[ 1 ]
> bhm[ 2, 4 ] <- bhm[ 4, 2 ] <- dat$fCapMatTL[ 1 ]
> bhm[ 3, 4 ] <- bhm[ 4, 3 ] <- dat$fLabMatTL[ 1 ]
> print(bhm)
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.000000	1.246349e+01	3.206967e+00	73.2131083318
[2,]	12.463493	-1.142053e-04	7.420925e-05	-0.0002080891
[3,]	3.206967	7.420925e-05	-3.595311e-05	0.0002498286
[4,]	73.213108	-2.080891e-04	2.498286e-04	-0.0020045254

Based on this bordered Hessian matrix, we can check whether the estimated Translog production function is quasiconcave at the first observation:

```
> det( bhm[ 1:2, 1:2 ] )

[1] -155.3387

> det( bhm[ 1:3, 1:3 ] )

[1] 0.01269176

> det( bhm )

[1] -2.99626e-05
```

Quasiconcavity requires that the first determinant calculated above (i.e. the determinant of the upper-left 2×2 submatrix) is negative and the following determinants alternate in sign. As this condition is fulfilled, we can conclude that the estimated Translog production function is quasiconcave at the first observation.

Now, we check whether our estimated Translog production function is quasiconcave at each individual observation:

```
> bhmTL <- array( 0, c( 4, 4, nrow( dat ) ) )
> bhmTL[ 1, 2, ] <- bhmTL[ 2, 1, ] <- dat$mpCapTL
> bhmTL[ 1, 3, ] <- bhmTL[ 3, 1, ] <- dat$mpLabTL
> bhmTL[ 1, 4, ] <- bhmTL[ 4, 1, ] <- dat$mpMatTL
> bhmTL[ 2, 2, ] <- dat$fCapCapTL
> bhmTL[ 3, 3, ] <- dat$fLabLabTL
> bhmTL[ 4, 4, ] <- dat$fMatMatTL
> bhmTL[ 2, 3, ] <- bhmTL[ 3, 2, ] <- dat$fCapLabTL
> bhmTL[ 2, 4, ] <- bhmTL[ 4, 2, ] <- dat$fCapMatTL
> bhmTL[ 3, 4, ] <- bhmTL[ 4, 3, ] <- dat$fLabMatTL
> dat$quasiConcTL <- apply( bhmTL[ 1:2, 1:2, ], 3, det ) < 0 &
+   apply( bhmTL[ 1:3, 1:3, ], 3, det ) > 0 &
+   apply( bhmTL, 3, det ) < 0
> table( dat$quasiConcTL )
```

```
FALSE  TRUE
    77    63
```

Our estimated Translog production function is quasiconcave at 63 of the 140 observations.

In total, there are 57 observations, at which both the monotonicity condition and the quasiconcavity condition are fulfilled:

```
> table( dat[ , c( "monoTL", "quasiConcTL" ) ] )
```

	quasiConcTL	
monoTL	FALSE	TRUE
FALSE	42	6
TRUE	35	57

There is a clear relationship between monotonicity and quasiconcavity: out of the 48 observations, at which monotonicity is violated, 42 observations (88%) also violate the quasiconcavity condition; and out of the 92 observations, at which monotonicity is fulfilled, 57 observations (62%) also fulfil the quasiconcavity condition.

2.6.13 Elasticities of substitution

2.6.13.1 Direct elasticities of substitution

We can calculate the direct elasticities of substitution using equation (2.20):

```
> dat$esdCapLabTL <- with( dat, - ( mpCapTL * mpLabTL *
+   ( qCap * mpCapTL + qLab * mpLabTL ) ) /
+   ( qCap * qLab * ( fCapCapTL * mpLabTL^2 -
```

```

+      2 * fCapLabTL * mpCapTL * mpLabTL + fLabLabTL * mpCapTL^2 ) ) )
> dat$esdCapMatTL <- with( dat, - ( mpCapTL * mpMatTL *
+      ( qCap * mpCapTL + qMat * mpMatTL ) ) /
+      ( qCap * qMat * ( fCapCapTL * mpMatTL^2 -
+      2 * fCapMatTL * mpCapTL * mpMatTL + fMatMatTL * mpCapTL^2 ) ) )
> dat$esdLabMatTL <- with( dat, - ( mpLabTL * mpMatTL *
+      ( qLab * mpLabTL + qMat * mpMatTL ) ) /
+      ( qLab * qMat * ( fLabLabTL * mpMatTL^2
+      - 2 * fLabMatTL * mpLabTL * mpMatTL + fMatMatTL * mpLabTL^2 ) ) )

```

As the elasticities of substitution measure changes in the marginal rates of technical substitution (MRTS) and the MRTS are meaningless if the monotonicity condition is not fulfilled, also the elasticities of substitution are meaningless if the monotonicity condition is not fulfilled. Hence, we visualize (the variation of) the direct elasticities of substitution only for the observations, where the monotonicity condition is fulfilled:

```

> hist( dat$esdCapLabTL[ dat$monoTL ], 30 )
> hist( dat$esdCapMatTL[ dat$monoTL ], 50 )
> hist( dat$esdLabMatTL[ dat$monoTL ], 30 )

```

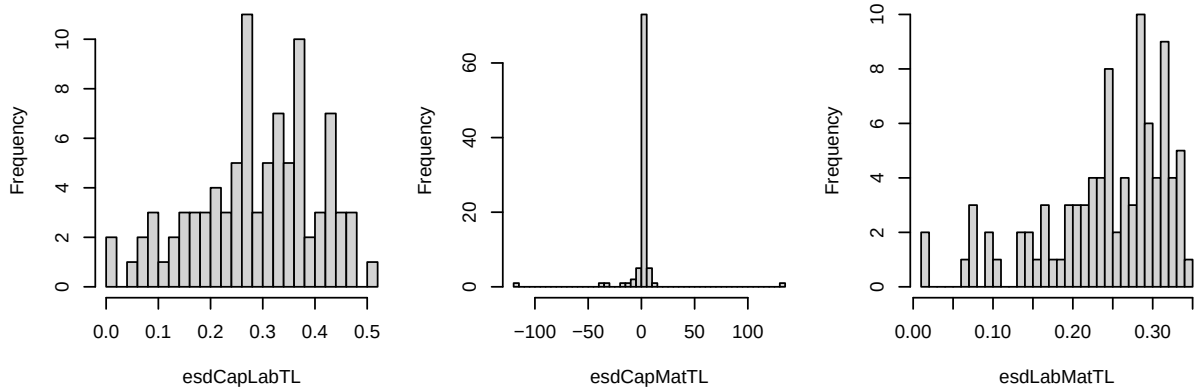


Figure 2.44: Translog production function: direct elasticities of substitution (at observations, where the monotonicity condition is fulfilled)

The resulting graphs are shown in figure 2.44. The estimated direct elasticities of substitution between capital and labor as well as between labor and materials are all positive, while the estimated direct elasticities of substitution between capital and materials are mostly positive but a small proportion of them is negative. As the direct elasticities of substitution assume that only the quantities of the two considered inputs are varied, while all other input quantities and the output quantity remain unchanged, it is expected that all direct elasticities of substitution are always positive. However, as we obtain a few negative direct elasticities of substitution between capital and materials, it seems that the isoquants between these two inputs are not always convex (see section 2.6.12). In case of non-convex isoquants, even the direct elasticities of substitution can

become negative. As non-convex isoquants are more similar to linear isoquants than to isoquants of Leontief production functions, negative direct elasticities of substitution rather indicate good substitutability than complementarity between the inputs.

In order to simplify the interpretation of the estimated direct elasticities of substitution, we visualize (the variation of) these elasticities of substitution only for the observations, where both the monotonicity condition and the quasiconcavity condition are fulfilled:

```
> hist( dat$esdCapLabTL[ dat$monoTL & dat$quasiConcTL ], 30 )
> hist( dat$esdCapMatTL[ dat$monoTL & dat$quasiConcTL ], 30 )
> hist( dat$esdLabMatTL[ dat$monoTL & dat$quasiConcTL ], 30 )
```

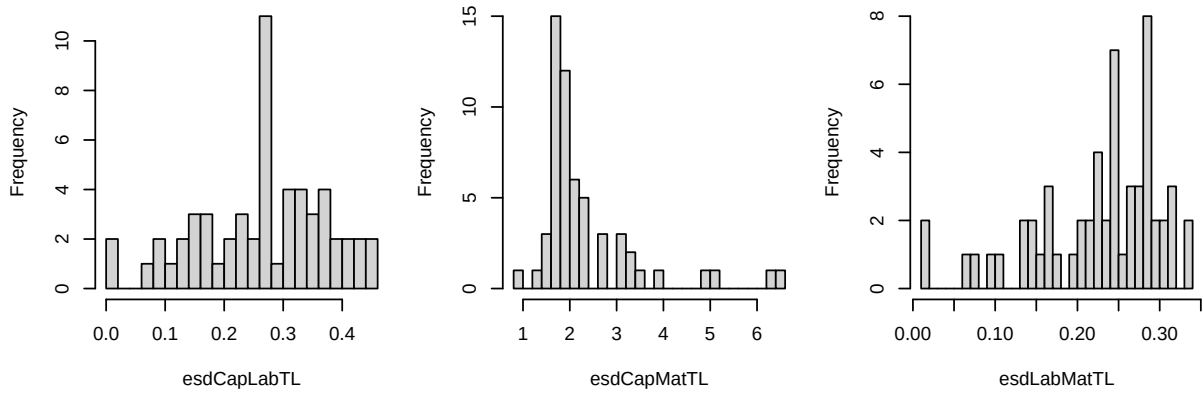


Figure 2.45: Translog production function: direct elasticities of substitution (at observations, where both the monotonicity condition and the quasiconcavity condition are fulfilled)

The resulting graphs are shown in figure 2.45. The corresponding median values of the direct elasticities of substitution can be obtained by:

```
> colMedians( subset( dat, dat$monoTL & dat$quasiConcTL,
+   c( "esdCapLabTL", "esdCapMatTL", "esdLabMatTL" ) ) )
```

```
esdCapLabTL esdCapMatTL esdLabMatTL
0.2685804   1.9410943   0.2441392
```

For all firms in our sample, the estimated direct elasticities of substitution between capital and labor and between labor and materials are considerably smaller than the elasticity of substitution implied by the Cobb-Douglas production function ($\sigma = 1$) and are somewhat close to the elasticity of substitution implied by the Leontief production function ($\sigma = 0$). Hence, the substitutability between capital and labor as well as the substitutability between labor and materials seems to be rather low.

The direct elasticities of substitution between capital and labor are for most firms around 0.27. Hence, if these firms keep the materials quantity and the output quantity unchanged and substitute capital for labor (or vice versa) so that the ratio between the capital quantity and the

labor quantity increases (decreases) by around 0.27 percent, the MRTS between capital and labor increases (decreases) by one percent. If the ratio between the labor price and the capital price increases by one percent, firms that keep the materials input and the output quantity unchanged and minimize costs will substitute capital for labor so that the ratio between the capital quantity and the labor quantity increases by around 0.27 percent. Hence, the relative changes of the ratios of the input quantities are much smaller than the relative changes of the ratios of the corresponding input prices, which indicates a very low substitutability between capital and labor.

Similarly, the direct elasticities of substitution between labor and materials are for most firms around 0.24. If these firms keep the capital quantity and the output quantity unchanged and substitute labor for materials (or vice versa) so that the ratio between the labor quantity and the quantity of materials increases (decreases) by around 0.24 percent, the MRTS between labor and materials increases (decreases) by one percent. If the ratio between the materials price and the labor price increases by one percent, firms that keep the capital input and the output quantity unchanged and minimize costs will substitute labor for materials so that the ratio between the labor quantity and the materials quantity increases by around 0.24 percent. Again, the relative changes of the ratios of the input quantities are much smaller than the relative changes of the ratios of the corresponding input prices, which indicates a very low substitutability between labor and materials.

In contrast, the direct elasticities of substitution between capital and materials are for most firms around 1.94 and thus, considerably larger than the elasticity of substitution implied by the Cobb-Douglas production function ($\sigma = 1$). Hence, if these firms keep the labor quantity and the output quantity unchanged and substitute capital for materials (or vice versa) so that the ratio between the capital quantity and the quantity of materials increases (decreases) by around 1.94 percent, the MRTS between capital and materials increases (decreases) by one percent. If the ratio between the materials price and the capital price increases by one percent, firms that keep the labor input and the output quantity unchanged and minimize costs will substitute capital for materials so that the ratio between the capital quantity and the quantity of materials increases by around 1.94 percent. Hence, the relative change of the ratio of the input quantities is around twice as large as the relative change of the ratio of the input prices, which indicates a quite substantial substitutability between capital and materials.

2.6.13.2 Allen elasticities of substitution

As for the quadratic production function, we only calculate the direct and Allen elasticities of substitution. The calculation of the Morishima elasticities of substitution requires only minimal changes of the code for calculating the Allen elasticities of substitution. In order to check whether our calculations are correct, we will—as before—check if the conditions (2.132) are fulfilled. In order to check these conditions, we need to calculate not only (normal) elasticities of substitution ($\sigma_{ij}; i \neq j$) but also economically not meaningful “elasticities of self-substitution” (σ_{ii}):

```

> dat$FCapLabTL <- - apply( bhmTL[ -2, -3, ], 3, det )
> dat$FCapMatTL <- apply( bhmTL[ -2, -4, ], 3, det )
> dat$FLabMatTL <- - apply( bhmTL[ -3, -4, ], 3, det )
> dat$FCapCapTL <- apply( bhmTL[ -2, -2, ], 3, det )
> dat$FLabLabTL <- apply( bhmTL[ -3, -3, ], 3, det )
> dat$FMatMatTL <- apply( bhmTL[ -4, -4, ], 3, det )
> dat$bhmDetTL <- apply( bhmTL, 3, det )
> dat$enumeratorTL <- with( dat,
+   qCap * mpCapTL + qLab * mpLabTL + qMat * mpMatTL )
> dat$esaCapLabTL <- with( dat,
+   numeratorTL / ( qCap * qLab ) * FCapLabTL / bhmDetTL )
> dat$esaCapMatTL <- with( dat,
+   numeratorTL / ( qCap * qMat ) * FCapMatTL / bhmDetTL )
> dat$esaLabMatTL <- with( dat,
+   numeratorTL / ( qLab * qMat ) * FLabMatTL / bhmDetTL )
> dat$esaCapCapTL <- with( dat,
+   numeratorTL / ( qCap * qCap ) * FCapCapTL / bhmDetTL )
> dat$esaLabLabTL <- with( dat,
+   numeratorTL / ( qLab * qLab ) * FLabLabTL / bhmDetTL )
> dat$esaMatMatTL <- with( dat,
+   numeratorTL / ( qMat * qMat ) * FMatMatTL / bhmDetTL )

```

Before we take a look at and interpret the elasticities of substitution, we check whether the conditions (2.132) are fulfilled:

```

> range( with( dat, qCap * mpCapTL * esaCapCapTL +
+   qLab * mpLabTL * esaCapLabTL + qMat * mpMatTL * esaCapMatTL ) )

[1] -5.215406e-08  9.536743e-07

> range( with( dat, qCap * mpCapTL * esaCapLabTL +
+   qLab * mpLabTL * esaLabLabTL + qMat * mpMatTL * esaLabMatTL ) )

[1] -2.980232e-08  5.960464e-08

> range( with( dat, qCap * mpCapTL * esaCapMatTL +
+   qLab * mpLabTL * esaLabMatTL + qMat * mpMatTL * esaMatMatTL ) )

[1] -2.235174e-08  5.960464e-07

```

The extremely small deviations from zero are most likely caused by rounding errors that are unavoidable on digital computers. This test does not prove that all of our calculations are done

correctly but if we had made a mistake, we probably would have discovered it. Hence, we can be rather sure that our calculations are correct.

As explained above in section 2.6.13.1, the elasticities of substitution are meaningless if the monotonicity condition is violated and they are difficult to interpret if the quasiconcavity condition is violated. Hence, we visualize (the variation of) the Allen elasticities of substitution only for the observations, where both the monotonicity condition and the quasiconcavity condition are fulfilled:

```
> hist( dat$esaCapLabTL[ dat$monoTL & dat$quasiConcTL ], 30 )
> hist( dat$esaCapMatTL[ dat$monoTL & dat$quasiConcTL ], 30 )
> hist( dat$esaLabMatTL[ dat$monoTL & dat$quasiConcTL ], 30 )
> hist( dat$esaCapLabTL[ dat$monoTL & dat$quasiConcTL &
+       abs( dat$esaCapLabTL ) < 10 ], 30 )
> hist( dat$esaCapMatTL[ dat$monoTL & dat$quasiConcTL &
+       abs( dat$esaCapMatTL ) < 10 ], 30 )
> hist( dat$esaLabMatTL[ dat$monoTL & dat$quasiConcTL &
+       abs( dat$esaLabMatTL ) < 10 ], 30 )
```

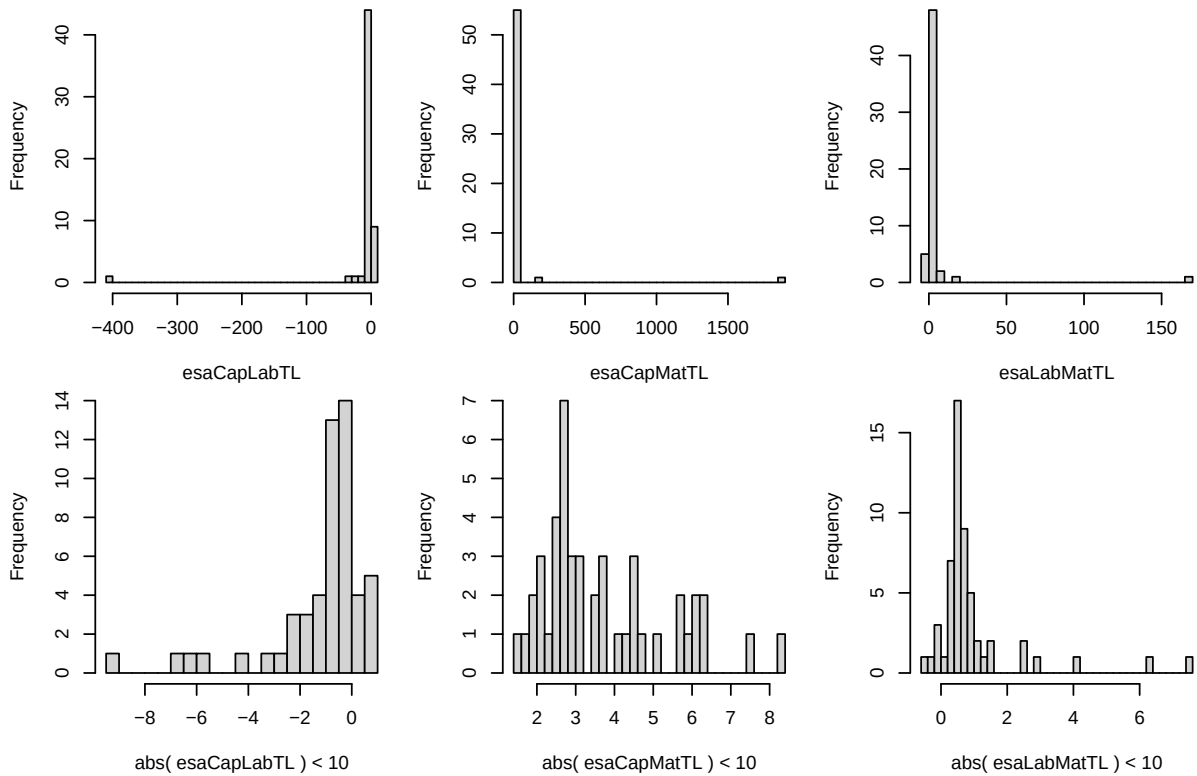


Figure 2.46: Translog production function: Allen elasticities of substitution

The resulting graphs are shown in figure 2.46. The estimated elasticities of substitution between capital and labor suggest that capital and labor are complements for the majority of firms. In

contrast, capital and materials are always substitutes and labor and materials are substitutes for the majority of firms. In order to avoid the effects of outliers, we use function `colMedians` (package *miscTools*) to obtain the median values of the Allen elasticities of substitution:

```
> colMedians( subset( dat, monoTL & quasiConcTL,
+   c( "esaCapLabTL", "esaCapMatTL", "esaLabMatTL" ) ) )

esaCapLabTL esaCapMatTL esaLabMatTL
-0.8018194    3.6322399    0.5848447
```

The median elasticity of substitution between labor and materials (0.58) lies between the elasticity of substitution implied by the Leontief production function ($\sigma = 0$) and the elasticity of substitution implied by the Cobb-Douglas production function ($\sigma = 1$). Hence, the substitutability between labor and materials seems to be rather low. If a typical firm keeps the output quantity constant, substitutes materials for labor (or vice versa) so that the ratio between the quantity of materials and the labor quantity increases (decreases) by 0.58 percent, and adjusts the capital quantity accordingly, the MRTS between materials and labor will increase (decrease) by one percent. If the ratio between the labor price and the materials price increases by one percent, a firm that keeps the output quantity constant, adjusts all input quantities and minimizes costs, will substitute materials for labor so that the ratio between the quantity of materials and the labor quantity increases by 0.58 percent. Hence, the relative change of the ratio of the input quantities is considerably smaller than the relative change of the ratio of the input prices, which indicates a low substitutability between labor and materials. In contrast, the median elasticity of substitution between capital and materials is larger than one (3.63), which indicates that it is much easier to substitute between capital and materials.

2.6.13.3 Comparison of direct and Allen elasticities of substitution

In the following, we use scatter plots to compare the estimated direct elasticities of substitution with the estimated Allen elasticities of substitution:

```
> compPlot( dat$esdCapLabTL[ dat$monoTL & dat$quasiConcTL ],
+   dat$esaCapLabTL[ dat$monoTL & dat$quasiConcTL ],
+   lim = c( -2, 2 ) )
> compPlot( dat$esdCapMatTL[ dat$monoTL & dat$quasiConcTL ],
+   dat$esaCapMatTL[ dat$monoTL & dat$quasiConcTL ],
+   lim = c( -10, 10 ) )
> compPlot( dat$esdLabMatTL[ dat$monoTL & dat$quasiConcTL ],
+   dat$esaLabMatTL[ dat$monoTL & dat$quasiConcTL ],
+   lim = c( -2, 2 ) )
```

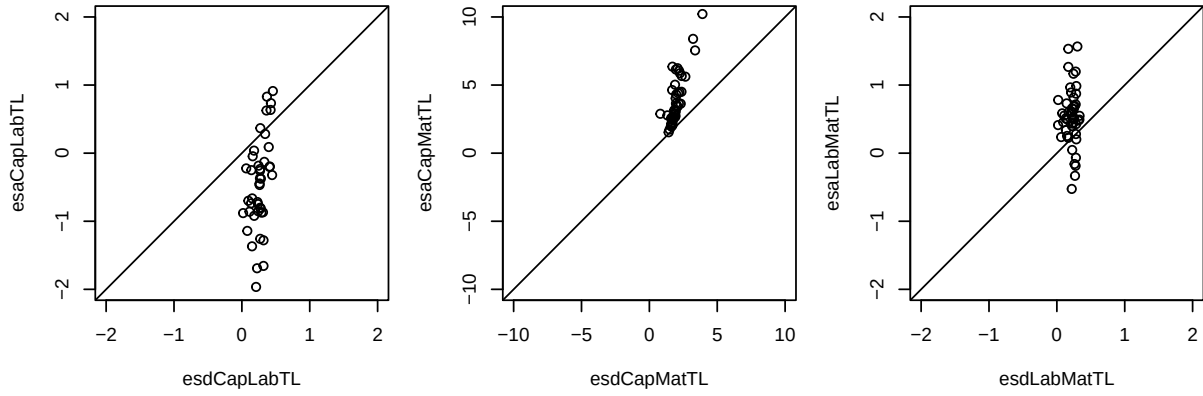


Figure 2.47: Translog production function: Comparison of direct and Allen elasticities of substitution

The resulting graphs are shown in figure 2.47. As can also be seen from comparing the histograms in Figures 2.45 and 2.46, the variation of the Allen elasticities of substitution is much larger than the variation of the corresponding direct elasticities of substitution.

2.6.14 Mean-scaled quantities

The Translog functional form is often estimated with mean-scaled variables. In the case of a Translog *production* function, it is estimated with mean-scaled input quantities and sometimes also with mean-scaled output quantities:

$$y^* = y/\bar{y} \quad (2.143)$$

$$x_i^* = x_i/\bar{x}_i \quad \forall i \quad (2.144)$$

The following commands create variables with mean-scaled output and input quantities:

```
> dat$qmOut <- with( dat, qOut / mean( qOut ) )
> dat$qmCap <- with( dat, qCap / mean( qCap ) )
> dat$qmLab <- with( dat, qLab / mean( qLab ) )
> dat$qmMat <- with( dat, qMat / mean( qMat ) )
```

This implies that the logarithms of the mean values of these variables are zero (except for negligible very small rounding errors):

```
> log( colMeans( dat[ , c( "qmOut", "qmCap", "qmLab", "qmMat" ) ] ) )
```

qmOut	qmCap	qmLab	qmMat
-1.110223e-16	-1.110223e-16	0.000000e+00	0.000000e+00

Please note that mean-scaling does *not* imply that the mean values of the logarithmic variables are zero:

```
> colMeans( log( dat[ , c( "qmOut", "qmCap", "qmLab", "qmMat" ) ] ) )

      qmOut      qmCap      qmLab      qmMat
-0.4860021 -0.3212057 -0.1565112 -0.2128551
```

The following derivation explores the relationship between a Translog production function with mean-scaled quantities and a Translog production function with the original quantities:

$$\ln y^* = \alpha_0^* + \sum_i \alpha_i^* \ln x_i^* + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln x_i^* \ln x_j^* \quad (2.145)$$

$$\ln (y/\bar{y}) = \alpha_0^* + \sum_i \alpha_i^* \ln (x_i/\bar{x}_i) + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln (x_i/\bar{x}_i) \ln (x_j/\bar{x}_j) \quad (2.146)$$

$$\ln y - \ln \bar{y} = \alpha_0^* + \sum_i \alpha_i^* (\ln x_i - \ln \bar{x}_i) + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* (\ln x_i - \ln \bar{x}_i) (\ln x_j - \ln \bar{x}_j) \quad (2.147)$$

$$\ln y = \alpha_0^* + \ln \bar{y} + \sum_i \alpha_i^* \ln x_i - \sum_i \alpha_i^* \ln \bar{x}_i \quad (2.148)$$

$$+ \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln x_i \ln x_j - \sum_i \sum_j \alpha_{ij}^* \ln x_i \ln \bar{x}_j + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln \bar{x}_i \ln \bar{x}_j$$

$$\ln y = \left(\alpha_0^* + \ln \bar{y} - \sum_i \alpha_i^* \ln \bar{x}_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln \bar{x}_i \ln \bar{x}_j \right) \quad (2.149)$$

$$+ \sum_i \left(\alpha_i^* - \sum_j \alpha_{ij}^* \ln \bar{x}_j \right) \ln x_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln x_i \ln x_j$$

Thus, the relationship between the coefficients of the Translog function with the original quantities and the coefficients of the Translog function with mean-scaled quantities is:

$$\alpha_0 = \alpha_0^* + \ln \bar{y} - \sum_i \alpha_i^* \ln \bar{x}_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij}^* \ln \bar{x}_i \ln \bar{x}_j \quad (2.150)$$

$$\alpha_i = \alpha_i^* - \sum_j \alpha_{ij}^* \ln \bar{x}_j \quad \forall i \quad (2.151)$$

$$\alpha_{ij} = \alpha_{ij}^* \quad \forall i, j. \quad (2.152)$$

Accordingly, the reciprocal relationship is:

$$\alpha_0^* = \alpha_0 - \ln \bar{y} + \sum_i \alpha_i \ln \bar{x}_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij} \ln \bar{x}_i \ln \bar{x}_j \quad (2.153)$$

$$\alpha_i^* = \alpha_i + \sum_j \alpha_{ij} \ln \bar{x}_j \quad \forall i \quad (2.154)$$

$$\alpha_{ij}^* = \alpha_{ij} \quad \forall i, j. \quad (2.155)$$

The following command estimates the Translog production function with mean-scaled quantities:

2 Primal Approach: Production Function

```
> prodTLm <- lm( log( qmOut ) ~ log( qmCap ) + log( qmLab ) + log( qmMat )
+   + I( 0.5 * log( qmCap )^2 ) + I( 0.5 * log( qmLab )^2 )
+   + I( 0.5 * log( qmMat )^2 ) + I( log( qmCap ) * log( qmLab ) )
+   + I( log( qmCap ) * log( qmMat ) ) + I( log( qmLab ) * log( qmMat ) ),
+   data = dat )
> summary( prodTLm )
```

Call:

```
lm(formula = log(qmOut) ~ log(qmCap) + log(qmLab) + log(qmMat) +
    I(0.5 * log(qmCap)^2) + I(0.5 * log(qmLab)^2) + I(0.5 * log(qmMat)^2) +
    I(log(qmCap) * log(qmLab)) + I(log(qmCap) * log(qmMat)) +
    I(log(qmLab) * log(qmMat)), data = dat)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.68015	-0.36688	0.05389	0.44125	1.26560

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.09392	0.08815	-1.065	0.28864
log(qmCap)	0.15004	0.11134	1.348	0.18013
log(qmLab)	0.79339	0.17477	4.540	1.27e-05 ***
log(qmMat)	0.50201	0.16608	3.023	0.00302 **
I(0.5 * log(qmCap)^2)	-0.02573	0.20834	-0.124	0.90189
I(0.5 * log(qmLab)^2)	-1.16364	0.67943	-1.713	0.08916 .
I(0.5 * log(qmMat)^2)	-0.50368	0.43498	-1.158	0.24902
I(log(qmCap) * log(qmLab))	0.56194	0.29120	1.930	0.05582 .
I(log(qmCap) * log(qmMat))	-0.40996	0.23534	-1.742	0.08387 .
I(log(qmLab) * log(qmMat))	0.65793	0.42750	1.539	0.12623

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6412 on 130 degrees of freedom

Multiple R-squared: 0.6296, Adjusted R-squared: 0.6039

F-statistic: 24.55 on 9 and 130 DF, p-value: < 2.2e-16

As expected, the intercept and the first-order coefficients have adjusted to the new units of measurement, while the second-order coefficients of the Translog function remain unchanged (compare with estimates in section [2.6.2](#)):

```
> all.equal( coef(prodTL)[-c(1:4)], coef(prodTLm)[-c(1:4)],
```

```
+   check.attributes = FALSE )
```

```
[1] TRUE
```

In case of functional forms that are invariant to the units of measurement (e.g. linear, Cobb-Douglas, quadratic, Translog), mean-scaling does not change the *relative* indicators of the technology (e.g. output elasticities, elasticities of scale, relative marginal rates of technical substitution, elasticities of substitution). As the logarithms of the mean values of the mean-scaled input quantities are zero, the first-order coefficients are equal to the output elasticities at the sample mean (see equation 2.135), i.e. the output elasticity of capital is 0.15, the output elasticity of labor is 0.793, the output elasticity of materials is 0.502, and the elasticity of scale is 1.445 at the sample mean.

2.6.15 First-order conditions for profit maximization

In this section, we will check to what extent the first-order conditions for profit maximization (2.40) are fulfilled, i.e. to what extent the firms use the optimal input quantities. We do this by comparing the marginal value products of the inputs with the corresponding input prices. We can calculate the marginal value products by multiplying the marginal products by the output price:

```
> dat$mvpCapTL <- dat$pOut * dat$mpCapTL
> dat$mvpLabTL <- dat$pOut * dat$mpLabTL
> dat$mvpMatTL <- dat$pOut * dat$mpMatTL
```

The command `compPlot` (package *miscTools*) can be used to compare the marginal value products with the corresponding input prices. As the logarithm of a non-positive number is not defined, we have to limit the comparisons on the logarithmic scale to observations with positive marginal products:

```
> compPlot( dat$pCap, dat$mvpCapTL )
> compPlot( dat$pLab, dat$mvpLabTL )
> compPlot( dat$pMat, dat$mvpMatTL )
> compPlot( dat$pCap[ dat$monoTL ], dat$mvpCapTL[ dat$monoTL ], log = "xy" )
> compPlot( dat$pLab[ dat$monoTL ], dat$mvpLabTL[ dat$monoTL ], log = "xy" )
> compPlot( dat$pMat[ dat$monoTL ], dat$mvpMatTL[ dat$monoTL ], log = "xy" )
```

The resulting graphs are shown in figure 2.48. They indicate that the marginal value products of most firms are higher than the corresponding input prices. This indicates that most firms could increase their profit by using more of all inputs. Given that the estimated Translog function shows that all firms operate under increasing returns to scale, it is not surprising that most firms would gain from increasing all input quantities. Therefore, the question arises why the firms in the sample did not do this. This question has already been addressed in section 2.3.10.

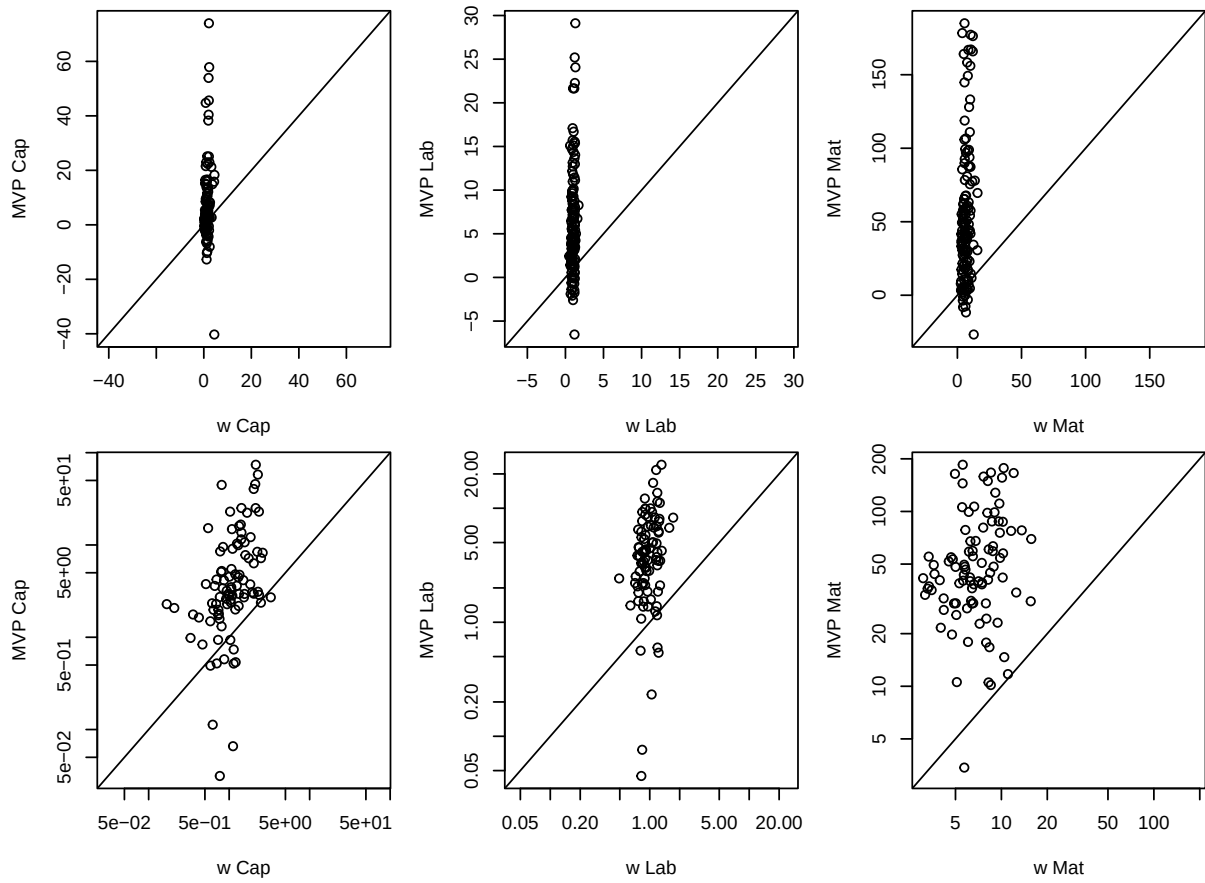


Figure 2.48: Marginal value products and corresponding input prices

2.6.16 First-order conditions for cost minimization

As the marginal rates of technical substitution differ between observations for the three other functional forms, we use scatter plots for visualizing the comparison of the input price ratios with the negative inverse marginal rates of technical substitution: As the marginal rates of technical substitution are meaningless if the monotonicity condition is not fulfilled, we limit the comparisons to the observations, where all monotonicity conditions are fulfilled:

```
> compPlot( ( dat$pCap / dat$pLab )[ dat$monoTL ],
+   - dat$mrtsLabCapTL[ dat$monoTL ] )
> compPlot( ( dat$pCap / dat$pMat )[ dat$monoTL ],
+   - dat$mrtsMatCapTL[ dat$monoTL ] )
> compPlot( ( dat$pLab / dat$pMat )[ dat$monoTL ],
+   - dat$mrtsMatLabTL[ dat$monoTL ] )
> compPlot( ( dat$pCap / dat$pLab )[ dat$monoTL ],
+   - dat$mrtsLabCapTL[ dat$monoTL ], log = "xy" )
> compPlot( ( dat$pCap / dat$pMat )[ dat$monoTL ],
+   - dat$mrtsMatCapTL[ dat$monoTL ], log = "xy" )
> compPlot( ( dat$pLab / dat$pMat )[ dat$monoTL ],
+   - dat$mrtsMatLabTL[ dat$monoTL ], log = "xy" )
```

The resulting graphs are shown in figure 2.49.

Furthermore, we use histograms to visualize the (absolute and relative) differences between the input price ratios and the corresponding negative inverse marginal rates of technical substitution:

```
> hist( ( - dat$mrtsLabCapTL - dat$pCap / dat$pLab )[ dat$monoTL ] )
> hist( ( - dat$mrtsMatCapTL - dat$pCap / dat$pMat )[ dat$monoTL ] )
> hist( ( - dat$mrtsMatLabTL - dat$pLab / dat$pMat )[ dat$monoTL ] )
> hist( log( - dat$mrtsLabCapTL / ( dat$pCap / dat$pLab ) )[ dat$monoTL ] )
> hist( log( - dat$mrtsMatCapTL / ( dat$pCap / dat$pMat ) )[ dat$monoTL ] )
> hist( log( - dat$mrtsMatLabTL / ( dat$pLab / dat$pMat ) )[ dat$monoTL ] )
```

The resulting graphs are shown in figure 2.50. The graphs in the middle column of figures 2.49 and 2.50 show that the ratio between the capital price and the materials price is larger than the absolute value of the marginal rate of technical substitution between materials and capital for a majority of the firms in the sample:

$$\frac{w_{cap}}{w_{mat}} > -MRTS_{mat, cap} = \frac{MP_{cap}}{MP_{mat}} \quad (2.156)$$

Hence, these firms can get closer to the minimum of their production costs by substituting materials for capital, because this will decrease the marginal product of materials and increase the marginal product of capital so that the absolute value of the MRTS between materials and

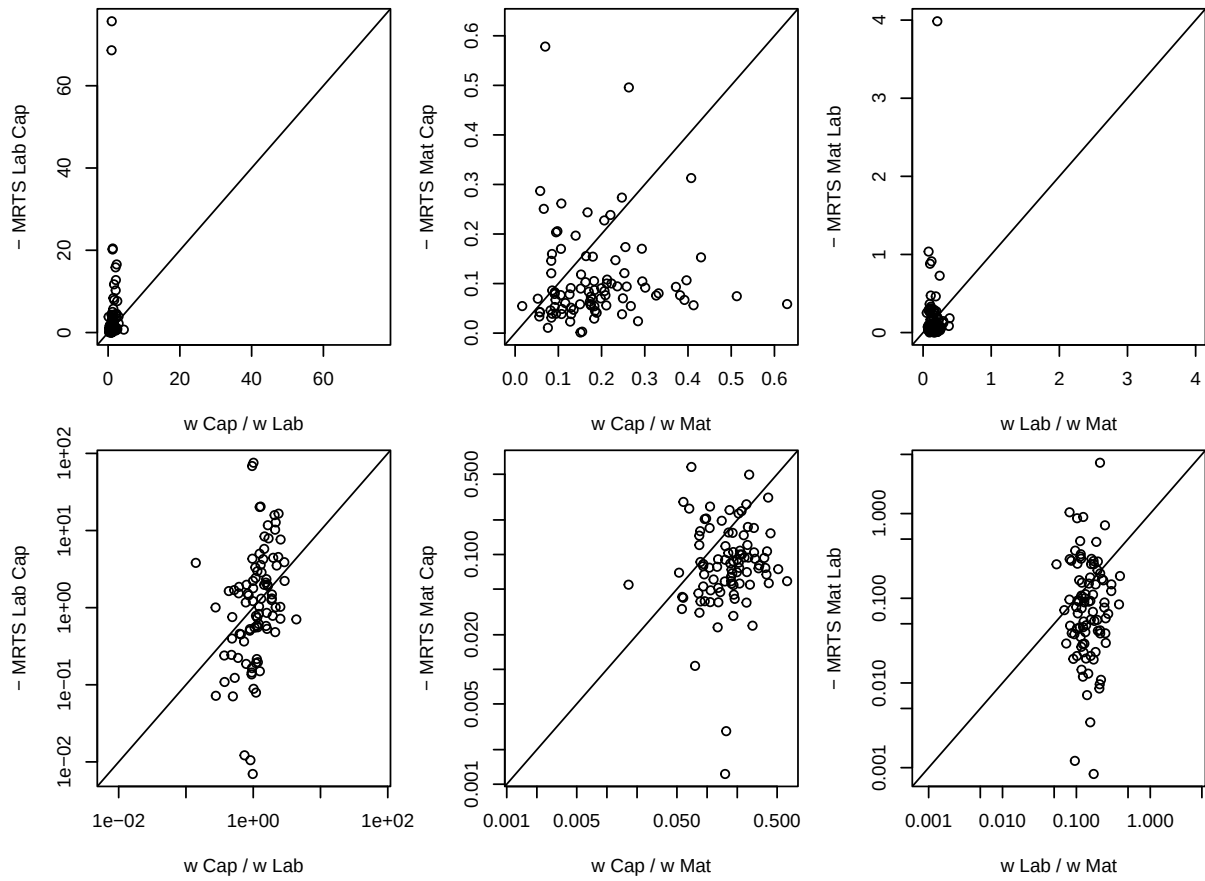


Figure 2.49: First-order conditions for costs minimization

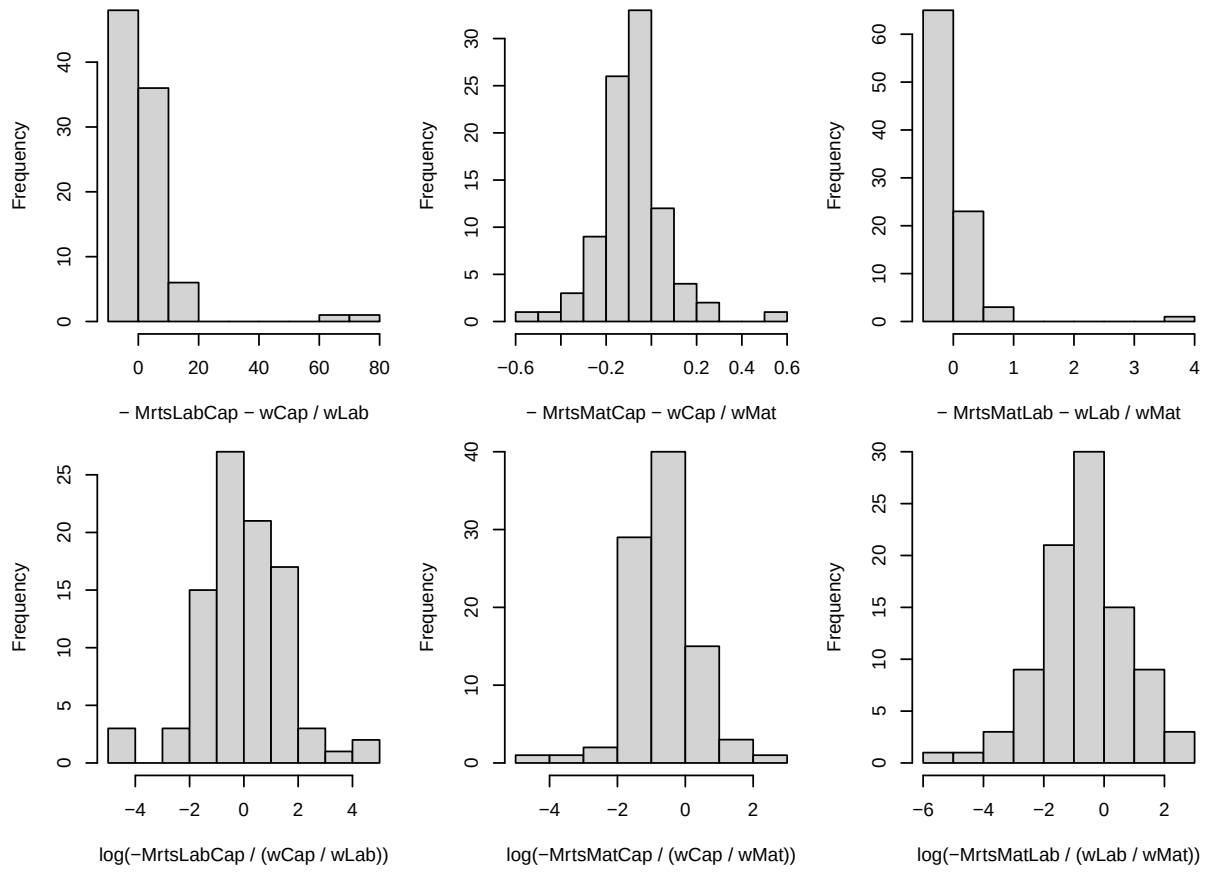


Figure 2.50: First-order conditions for costs minimization

capital increases and gets closer to the corresponding input price ratio. The graphs on the left indicate that approximately half of the firms should substitute labor for capital, while the other half should substitute capital for labor. The graphs on the right indicate that a majority of the firms should substitute materials for labor. Hence, the majority of the firms could reduce production costs particularly by using more materials and using less labor or less capital but there might be (legal) regulations that restrict the use of materials (e.g. fertilizers, pesticides).

2.7 Evaluation of different functional forms

In this section, we will discuss the appropriateness of the four different functional forms for analyzing the production technology in our data set. If one functional form is nested in another functional form, we can use standard statistical tests to compare these functional forms. We have done this already in section 2.5 (linear production function vs. quadratic production function) and in section 2.6 (Cobb-Douglas production function vs. Translog production function). The tests clearly reject the linear production function in favor of the quadratic production function but it is less clear whether the Cobb-Douglas production function is rejected in favor of the Translog production function.

It is much less straight-forward to compare non-nested models such as the quadratic and the Translog production functions.

Potential criteria for selecting functional forms are:

- number of observations and number of inputs: more observations and fewer inputs $\rightarrow$ more flexible functional form
- plausibility of the properties implied by the functional form in the specific empirical application (e.g., constant marginal products, constant output elasticities, constant elasticity of scale, essentiality, specific values of the elasticity of substitution, ...)
- distribution / skewness of variables (e.g., y or $\ln y$?)
- goodness of fit (in terms of y and $\ln y$, taking into account degrees of freedom)
- visual fit
- Wald-tests and/or LR-tests to compare nested functional forms
- statistical significance of inputs
- number of non-negativity violations
- number of monotonicity violations
- quasiconvexity (depending on the production process in the real world and the aim of the analysis)
- tests for the functional form (e.g., RESET)
- homo-/heteroskedasticity (e.g., BP test)
- plausibility of the result

2.7.1 Goodness of fit

As the linear and quadratic models use different dependent variables than the Cobb-Douglas and Translog models (y vs. $\ln y$), we cannot simply compare the R^2 -values. However, we can calculate the hypothetical R^2 -values regarding y for the Cobb-Douglas and Translog production functions and compare them with the R^2 values of the linear and quadratic production functions. We can also calculate the hypothetical R^2 -value regarding $\ln y$ for the linear and quadratic production functions and compare them with the R^2 values of the Cobb-Douglas and Translog production functions. We can calculate the (hypothetical) R^2 values with function `rSquared` (package *misc-Tools*). The first argument of this function must be a vector of the observed dependent variable and the second argument must be a vector of the residuals. We start by extracting the R^2 values from the linear and quadratic models and calculate the hypothetical R^2 -values regarding y for the Cobb-Douglas and Translog production functions:

```
> summary(prodLin)$r.squared

[1] 0.786819

> summary(prodQuad)$r.squared

[1] 0.8448983

> rSquared( dat$qOut, dat$qOut - dat$qOutCD )

      [,1]
[1,] 0.8067198

> rSquared( dat$qOut, dat$qOut - dat$qOutTL )

      [,1]
[1,] 0.7696638
```

In this case, the R^2 value regarding y is considerably higher for the quadratic function than for the other functional forms, particularly the Translog functional form, which has the lowest R^2 value regarding y . Similarly, we can extract the R^2 values from the Cobb-Douglas and Translog models and calculate the hypothetical R^2 -values regarding $\ln y$ for the linear and quadratic production functions. As the logarithm of non-positive numbers is not defined, we need to exclude observations with non-positive predicted output quantities:

```
> summary(prodCD)$r.squared

[1] 0.5943154

> summary(prodTL)$r.squared
```

```
[1] 0.6295696

> rSquared( log( dat$qOut[ dat$qOutLin > 0 ] ),
+   log( dat$qOut[ dat$qOutLin > 0 ] ) - log( dat$qOutLin[ dat$qOutLin > 0 ] ) )

      [,1]
[1,] 0.3796727

> rSquared( log( dat$qOut ), log( dat$qOut ) - log( dat$qOutQuad ) )

      [,1]
[1,] 0.5481309
```

In contrast to the R^2 values regarding y , the R^2 value regarding $\ln y$ is considerably higher for the Translog function than for the other functional forms. Hence, in our case, the R^2 values do not help much to select the most suitable functional form. As the quadratic and the Translog functional forms have the same number of coefficients, we can base our comparison of these functional forms on the unadjusted R^2 values. In order to compare models that have different numbers of coefficients, one should rather use *adjusted* R^2 values.

Furthermore, we can visually compare the fit of the two models by looking at figures 2.7, 2.15, 2.24 and 2.37. The Cobb-Douglas and quadratic production functions clearly over-predict the output of small firms so that small firms have rather large *relative* error terms. On the other hand, the Translog production function has rather large *absolute* error terms for large firms. In total, it seems that the fit of the linear and Translog production functions is slightly better than the fit of the Cobb-Douglas and quadratic production functions.

2.7.2 Test for functional form misspecification

We conduct Ramsey's (1969) Regression Equation Specification Error Test (RESET) on all four functional forms:

```
> resettest( prodLin )

      RESET test

data:  prodLin
RESET = 17.639, df1 = 2, df2 = 134, p-value = 1.584e-07

> resettest( prodCD )

      RESET test

data:  prodCD
RESET = 2.9224, df1 = 2, df2 = 134, p-value = 0.05724
```

```
> resettest( prodQuad )
```

```
RESET test
```

```
data: prodQuad
```

```
RESET = 7.3663, df1 = 2, df2 = 128, p-value = 0.0009374
```

```
> resettest( prodTL )
```

```
RESET test
```

```
data: prodTL
```

```
RESET = 1.2811, df1 = 2, df2 = 128, p-value = 0.2813
```

While the linear and quadratic functional forms are clearly rejected, the Cobb-Douglas functional form is only rejected at the 10%, and the Translog is not rejected at all.

2.7.3 Theoretical consistency

Furthermore, we can compare the theoretical consistency of the models. The total number of monotonicity violations of the four functional forms can be obtained by

```
> with( dat, sum( eCap < 0 ) + sum( eLab < 0 ) + sum( eMat < 0 ) )
```

```
[1] 0
```

```
> with( dat, sum( mpCapCD < 0 ) + sum( mpLabCD < 0 ) + sum( mpMatCD < 0 ) )
```

```
[1] 0
```

```
> with( dat, sum( eCapQuad < 0 ) + sum( eLabQuad < 0 ) + sum( eMatQuad < 0 ) )
```

```
[1] 41
```

```
> with( dat, sum( eCapTL < 0 ) + sum( eLabTL < 0 ) + sum( eMatTL < 0 ) )
```

```
[1] 54
```

Alternatively, we could look at the number of observations, at which the monotonicity condition is violated:

```
> table( dat$monoQuad )
```

```
FALSE TRUE
```

```
39 101
```

```
> table( dat$monoTL )
```

```
FALSE  TRUE
    48    92
```

While the monotonicity conditions are globally fulfilled for the linear and Cobb-Douglas functional forms, the more flexible functional forms have several monotonicity violations. When comparing only the two more flexible functional forms, we see that the monotonicity condition is more often violated in the Translog function than in the quadratic function (according to both of the measures of monotonicity violation).

While the Cobb-Douglas and Translog production functions always return positive output quantities (as long as all input quantities are strictly positive), this is not necessarily the case for linear and quadratic production functions. However, we have checked this in sections 2.3.4 and 2.5.6 and found only one negative predicted output quantity for the linear production function, while all output quantities predicted by the quadratic production function are positive. Hence, the non-negativity condition is almost fulfilled for the linear production function, while it is completely fulfilled for the remaining three functional forms.

Quasiconcavity is fulfilled at all observations for the linear and Cobb-Douglas production functions and at 63 out of 140 observations for the Translog production function, while it is fulfilled at no observation for the quadratic production function. However, quasiconcavity is mainly assumed to simplify the (further) economic analysis (e.g. to obtain continuous input demand and output supply functions) and there can be found good reasons for why the true production technology is not quasiconcave (e.g. indivisibility of inputs).

2.7.4 Plausible estimates

While the elasticities of scale of some observations were implausibly large when estimated with the linear production function, no elasticities of scale estimated by the Cobb-Douglas, quadratic and Translog production functions are in the implausible range:

```
> table( dat$eScale >= 0.5 & dat$eScale <= 2 )
```

```
FALSE  TRUE
    67    73
```

```
> table( rep( sum( coef( prodCD )[-1] ) >= 0.5 & sum( coef( prodCD )[-1] ) <= 2,
+   nrow( dat ) ) )
```

```
TRUE
 140
```

```
> table( dat$eScaleQuad >= 0.5 & dat$eScaleQuad <= 2 )
```

TRUE

140

```
> table( dat$eScaleTL >= 0.5 & dat$eScaleTL <= 2 )
```

TRUE

140

In the following we check for implausibly large output elasticities:

```
> with( dat, sum( eCap > 1 ) + sum( eLab > 1 ) + sum( eMat > 1 ) )
```

[1] 124

```
> ( coef( prodCD )[2] > 1 ) * nrow( dat ) +
+   ( coef( prodCD )[3] > 1 ) * nrow( dat ) +
+   ( coef( prodCD )[4] > 1 ) * nrow( dat )
```

log(qCap)

0

```
> with( dat, sum( eCapQuad > 1 ) + sum( eLabQuad > 1 ) + sum( eMatQuad > 1 ) )
```

[1] 28

```
> with( dat, sum( eCapTL > 1 ) + sum( eLabTL > 1 ) + sum( eMatTL > 1 ) )
```

[1] 56

The linear production function indicates the most implausibly large output elasticities, followed by the Translog, quadratic, and Cobb-Douglas production functions, whereas the Cobb-Douglas production function has no implausibly large output elasticities. When comparing just the two more flexible functional forms, the Translog production function results in twice as many implausible output elasticities than the quadratic production function.

2.7.5 Summary

The various criteria for assessing and comparing the appropriateness of the four functional forms for analyzing the production technology in our data set are summarized in table 2.2. While the linear production function seems to be the least appropriate for our specific empirical analysis, the Cobb-Douglas functions performs rather well according to most indicators. When comparing the two more flexible functional forms, we can see that the quadratic production function results in less monotonicity violations and less implausible output elasticities than the Translog production function, while the Translog production function seems to give a better fit to the data (visually and according to Ramsey's RESET).

Table 2.2: Criteria for assessing functional forms

	linear	Cobb-Douglas	quadratic	Translog
R^2 of y	0.79	0.81	0.84	0.77
R^2 of $\ln y$	0.38	0.59	0.55	0.63
visual fit	+	—	—	(+)
RESET (P-value)	0	0.05724	0.00094	0.28127
total monotonicity violations	0	0	41	54
observations with monotonicity violated	0	0	39	48
negative output quantities	1	0	0	0
observations with quasiconcavity violated	0	0	140	77
implausible elasticities of scale	67	0	0	0
implausible output elasticities	124	0	28	56

2.8 Non-parametric production function

In order to avoid the specification of a functional form of the production function, the production technology can be analyzed by nonparametric regression. We will use a local-linear kernel regressor with an Epanechnikov kernel for the (continuous) regressors (see, e.g. [Li and Racine, 2007](#); [Racine, 2008](#)). “One can think of this estimator as a set of weighted linear regressions, where a weighted linear regression is performed at each observation and the weights of the other observations decrease with the distance from the respective observation. The weights are determined by a kernel function and a set of bandwidths, where a bandwidth for each explanatory variable must be specified. The smaller the bandwidth, the faster the weight decreases with the distance from the respective observation. In our study, we make the frequently used assumption that the bandwidths can differ between regressors but are constant over the domain of each regressor. While the bandwidths were initially determined by using a rule of thumb, nowadays increased computing power allows us to select the optimal bandwidths for a given model and data set according to the expected Kullback-Leibler cross-validation criterion ([Hurvich, Simonoff, and Tsai, 1998](#)). Hence, in nonparametric kernel regression, the overall shape of the relationship between the inputs and the output is determined by the data and the (marginal) effects of the explanatory variables can differ between observations without being restricted by an arbitrarily chosen functional form.” ([Czekaj and Henningsen, 2012](#)). Given that the distributions of the output quantity and the input quantities are strongly right-skewed in our data set (many firms with small quantities, only a few firms with large quantities), we use the logarithms of the output and input quantities in order to achieve more uniform distributions, which are preferable in case of fixed bandwidths. Furthermore, this allows us to interpret the gradients of the dependent variable (logarithmic output quantity) with respect to the explanatory variables (logarithmic input quantities) as output elasticities. The following commands load the R package *np* ([Hayfield and Racine, 2008](#)), select the optimal bandwidths and estimate the model, and show summary results:

```
> library( "np" )
```

```
> prodNP <- npreg( log(qOut) ~ log(qCap) + log(qLab) + log(qMat), regtype = "ll",
+   bwmethod = "cv.aic", ckertype = "epanechnikov", data = dat,
+   gradients = TRUE )
> summary( prodNP )
```

Regression Data: 140 training points, in 3 variable(s)

log(qCap) log(qLab) log(qMat)

Bandwidth(s): 1.039647 332644 0.8418465

Kernel Regression Estimator: Local-Linear

Bandwidth Type: Fixed

Residual standard error: 0.6227669

R-squared: 0.6237078

Continuous Kernel Type: Second-Order Epanechnikov

No. Continuous Explanatory Vars.: 3

While the bandwidths of the logarithmic quantities of capital and materials are around one, the bandwidth of the logarithmic labor quantity is rather large. These bandwidths indicate that the logarithmic output quantity non-linearly changes with the logarithmic quantities of capital and materials but it changes approximately linearly with the logarithmic labor quantity.

The estimated relationship between each explanatory variable and the dependent variable (holding all other explanatory variables constant at their median values) can be visualized using the `plot` method. We can use argument `plot.errors.method` to add confidence intervals:

```
> plot( prodNP, plot.errors.method = "bootstrap" )
```

The resulting graphs are shown in figure 2.51.

The estimated gradients of the dependent variable with respect to each explanatory variable (holding all other explanatory variables constant at their median values) can be visualized using the `plot` method with argument `gradient` set to `TRUE`:

```
> plot( prodNP, gradients = TRUE, plot.errors.method = "bootstrap" )
```

The resulting graphs are shown in figure 2.52.

Function `npsigtest` can be used to obtain the statistical significance of the explanatory variables:

```
> npsigtest( prodNP )
```

Kernel Regression Significance Test

Type I Test with IID Bootstrap (399 replications, Pivot = TRUE, joint = FALSE)

Explanatory variables tested for significance:

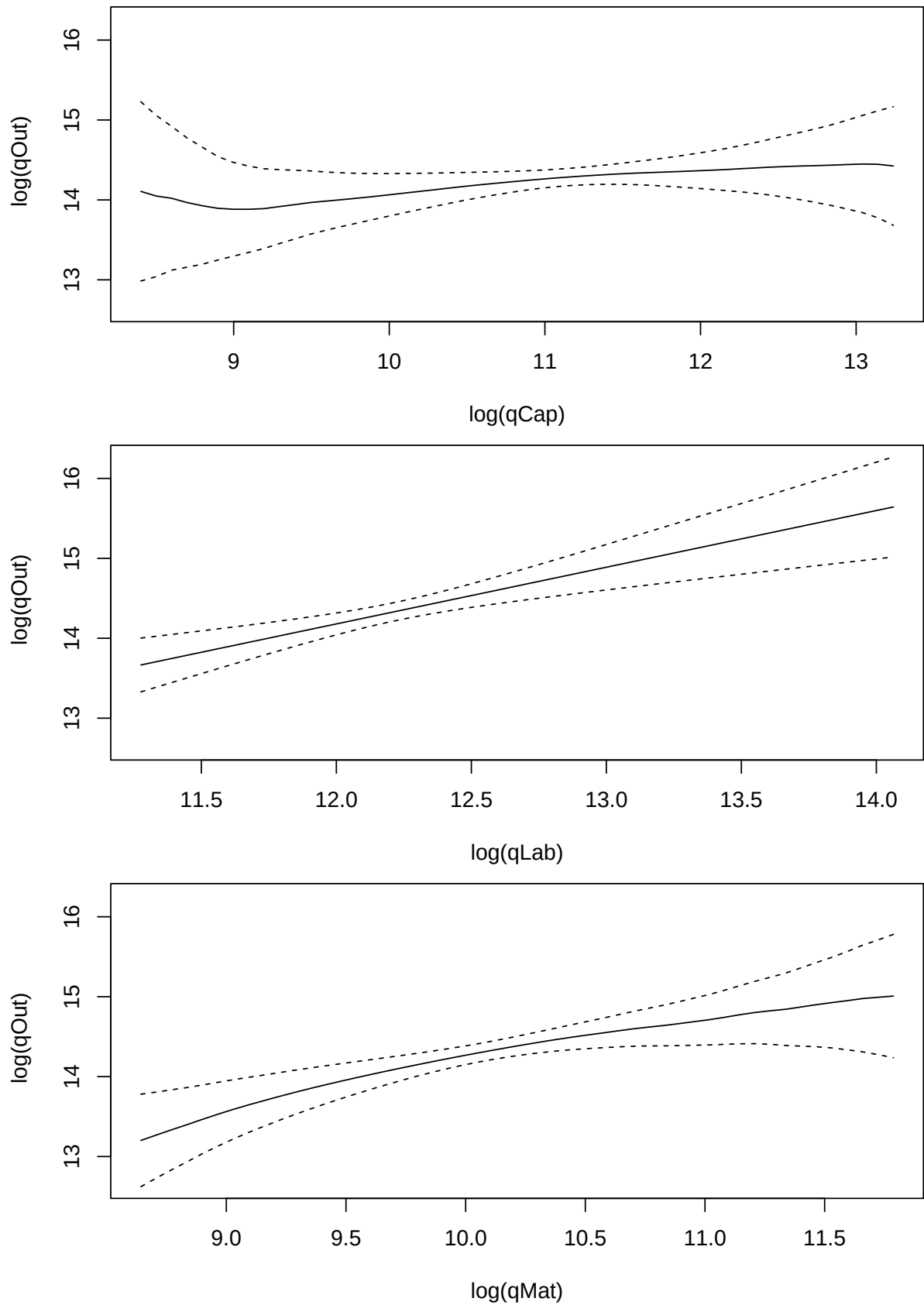


Figure 2.51: Production technology estimated by non-parametric kernel regression

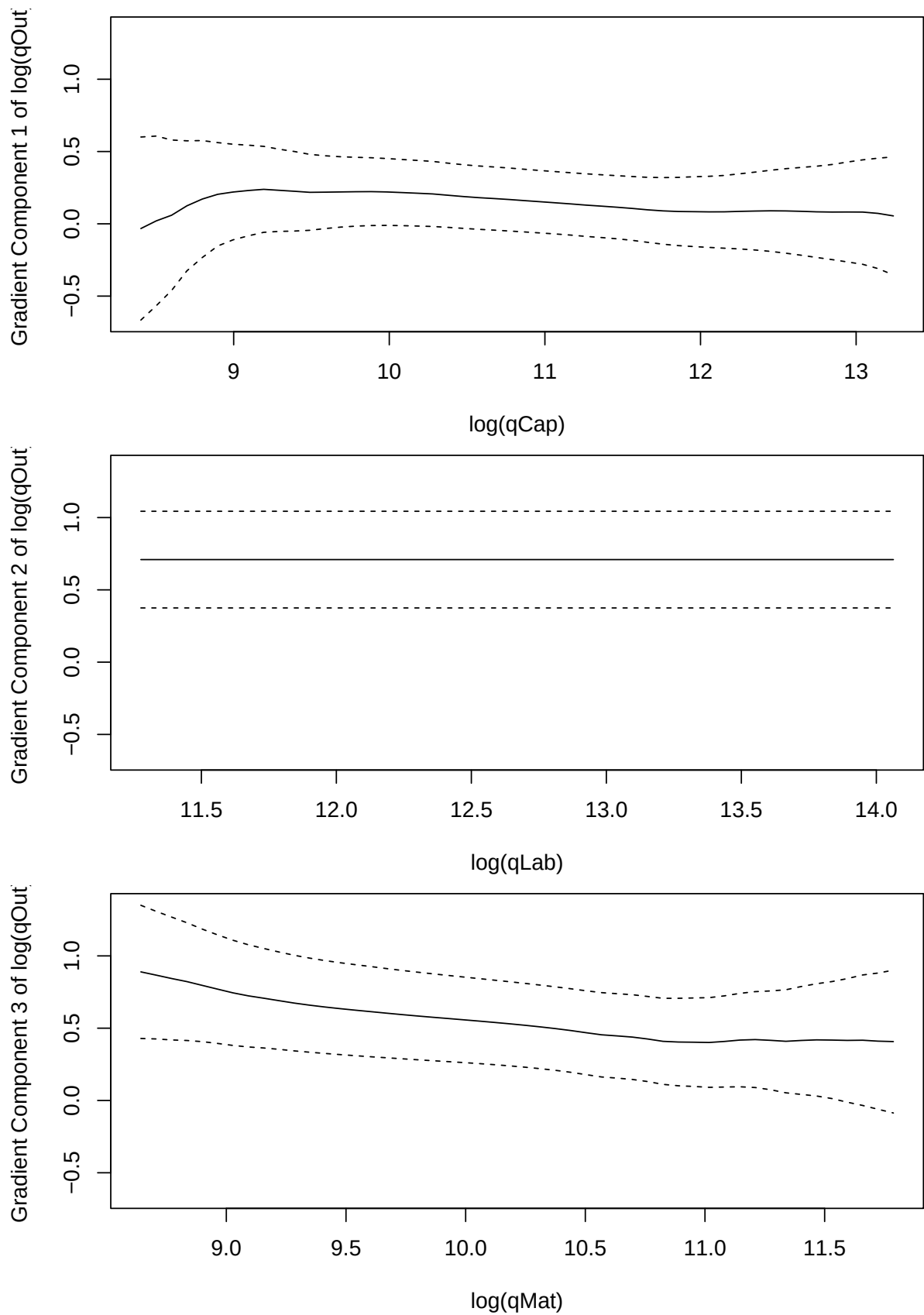


Figure 2.52: Gradients (output elasticities) estimated by non-parametric kernel regression

```
log(qCap) (1), log(qLab) (2), log(qMat) (3)
```

```
          log(qCap) log(qLab) log(qMat)
Bandwidth(s):  1.039647    332644 0.8418465
```

```
Individual Significance Tests
```

```
P Value:
```

```
log(qCap) 0.12281
```

```
log(qLab) < 2e-16 ***
```

```
log(qMat) < 2e-16 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results confirm the results from the parametric regressions that labor and materials have a significant effect on the output while capital does not have a significant effect (at 10% significance level).

The following commands plot histograms of the three output elasticities and the elasticity of scale:

```
> hist( gradients( prodNP )[ ,1] )
> hist( gradients( prodNP )[ ,2] )
> hist( gradients( prodNP )[ ,3] )
> hist( rowSums( gradients( prodNP ) ) )
```

The resulting graphs are shown in figure 2.53. The monotonicity condition is fulfilled at almost all observations, only 1 output elasticity of capital and 0 output elasticity of labor is negative. All firms operate under increasing returns to scale with most farms having an elasticity of scale around 1.4.

Finally, we visualize the relationship between firm size and the elasticity of scale based on our non-parametric estimation results:

```
> plot( dat$qOut, rowSums( gradients( prodNP ) ), log = "x" )
> plot( dat$X, rowSums( gradients( prodNP ) ), log = "x" )
```

The resulting graph is shown in figure 2.54. The smallest firms generally would gain most from increasing their size. However, also the largest firms would still considerably gain from increasing their size—perhaps even more than medium-sized firms but there is probably insufficient evidence to be sure about this.

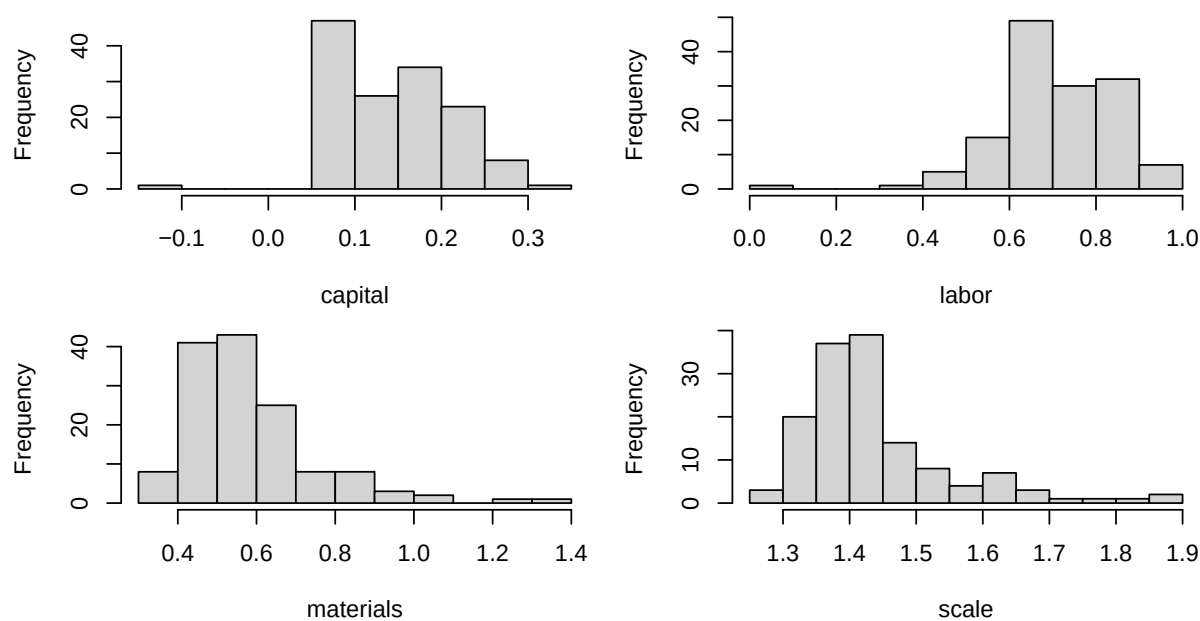


Figure 2.53: Output elasticities and elasticities of scale estimated by non-parametric kernel regression

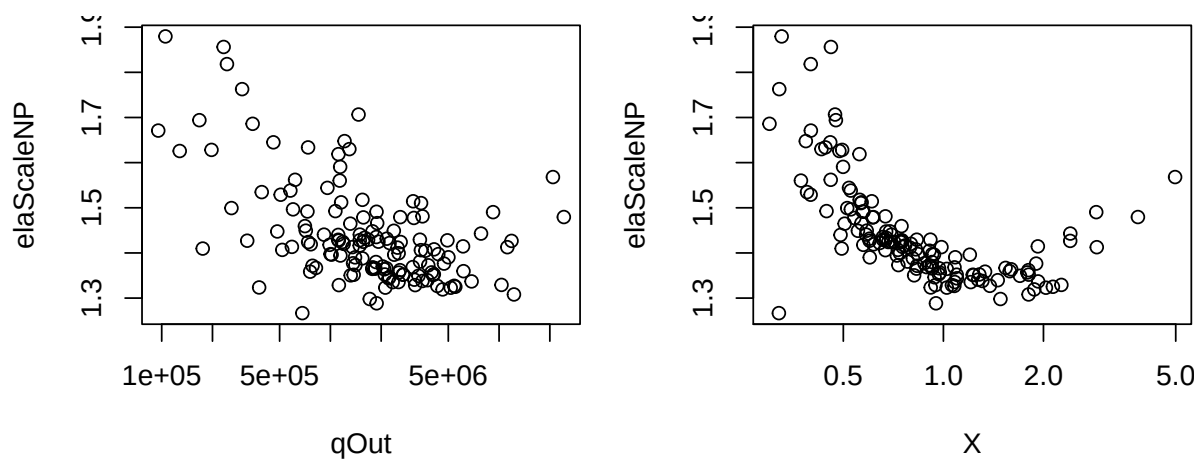


Figure 2.54: Relationship between firm size and elasticities of scale estimated by non-parametric kernel regression

3 Dual Approach: Cost Functions

3.1 Theory

3.1.1 Cost function

Total cost is defined as:

$$c = \sum_i w_i x_i \quad (3.1)$$

The cost function:

$$c(w, y) = \min_x \sum_i w_i x_i, \text{ s.t. } f(x) \geq y \quad (3.2)$$

returns the minimal (total) cost that is required to produce at least the output quantity y given input prices w .

It is important to distinguish the cost definition (3.1) from the cost function (3.2).

3.1.2 Properties of the cost function

It is usually assumed that a cost function $c(w, y)$ defined in (3.2) fulfills the following properties (see, e.g., [Chambers, 1988](#), p. 51-56):

1. $c(w, y)$ is non-negative, i.e., $c(w, y) \geq 0 \forall w \geq 0$ and $y \geq 0$,
and $c(w, y) > 0 \forall w > 0$ and $y > 0$
2. $c(w, y)$ is non-decreasing in w , i.e. $\partial c(w, y) / \partial w_i \geq 0 \forall i$
3. $c(w, y)$ is linearly homogeneous in w , i.e. $c(k w, y) = k c(w, y) \forall k > 0$
4. $c(w, y)$ is concave in w
5. $c(w, y)$ is non-decreasing in y , i.e. $\partial c(w, y) / \partial y \geq 0$
6. no fixed costs, i.e., $c(w, 0) = 0$

3.1.3 Cost flexibility and elasticity of size

The ratio between the relative change in total costs and the relative change in the output quantity is called “cost flexibility:”

$$\frac{\partial c(w, y)}{\partial y} \frac{y}{c(w, y)} \quad (3.3)$$

The inverse of the cost flexibility is called “elasticity of size:”

$$\epsilon^*(w, y) = \frac{\partial y}{\partial c(w, y)} \frac{c(w, y)}{y} \quad (3.4)$$

At the cost-minimizing points, the elasticity of size is equal to the elasticity of scale (Chambers, 1988, p. 71–72). For homothetic production technologies such as the Cobb-Douglas production technology, the elasticity of size is always equal to the elasticity of scale (Chambers, 1988, p. 72–74).¹

3.1.4 Homotheticity of cost functions

A cost function is homothetic if it can be decomposed as:

$$c(w, y) = c_0(c_1(w), c_2(y)), \quad (3.5)$$

where $c_1(w)$ is a function that aggregates the input prices (w) to a single value, $c_2(y)$ is a function that transforms the output quantity (y) (or—in the case of multiple outputs—that aggregates the output quantities to a single value), and $c_0(c_1, c_2)$ is a function that returns the costs based on the aggregated input price and the transformed output quantity. If a cost function is homothetic, there are no interactions between individual input prices and the output quantity, i.e., when the output quantity changes, the cost function is just scaled up or scaled down (proportionally or non-proportionally with the output quantity), while its ‘shape’ remains unchanged. This means, for example, that the optimal cost shares of the inputs are independent of the scale of production (i.e., they do not depend on the output quantities).

3.1.5 Short-run cost functions

As producers often cannot instantly adjust the quantity of the some inputs (e.g. buildings, land, apple trees), we separate the vector of input quantities (x) into two subvectors ($x = (x^1, x^2)'$), where vector x^1 includes the quantities of the variable inputs, i.e. the inputs that can be freely adjusted in the short run, and vector x^2 includes the quantities of the quasi-fixed inputs, i.e. the inputs that cannot be adjusted in the short run. In this case, the production function becomes $y = f(x^1, x^2)$.

In the long run, when all input quantities can be freely adjusted, the cost minimization problem remains:

$$\min_{x^1, x^2} w^{1'} x^1 + w^{2'} x^2, \text{ s.t. } f(x^1, x^2) \geq y \quad (3.6)$$

¹Further details about the relationship between the elasticity of size and the elasticity of scale are available, e.g., in McClelland, Wetzstein, and Musserwetz (1986).

and the cost function remains

$$c(w, y) = c(w^1, w^2, y) = \min_{x^1, x^2} w^1 x^1 + w^2 x^2, \text{ s.t. } f(x^1, x^2) \geq y, \quad (3.7)$$

where w^1 denotes the vector of the prices of the variable inputs and w^2 denotes the vector of the prices of the quasi-fixed inputs.

However, in the short run, when only the variable input quantities (x^1) can be freely adjusted, while the quasi-fixed input quantities (x^2) cannot be adjusted, the costs of the quasi-fixed inputs ($w^2 x^2$) are “sunk costs” and thus are irrelevant for short-run cost minimization. Hence, one only needs to take into account—and minimize—the variable costs:

$$\min_{x^1} w^1 x^1, \text{ s.t. } f(x^1, x^2) \geq y. \quad (3.8)$$

Based on this short-run (variable) cost minimization, we can derive the short-run (variable) cost function:

$$c^v(w^1, y, x^2) = \min_{x^1} w^1 x^1, \text{ s.t. } f(x^1, x^2) \geq y, \quad (3.9)$$

where c^v denotes the variable costs defined in equation (1.3).

Total short-run costs consist of variable costs and sunk costs (see, e.g., [Chambers, 1988](#), p. 103):

$$c^s(w, y, x^2) = c^s(w^1, w^2, y, x^2) = c^v(w^1, y, x^2) + w^2 x^2. \quad (3.10)$$

The long-run cost function can be expressed in terms of the short-run total cost function or the short-run (variable) cost function (see, e.g., [Chambers, 1988](#), p. 103):

$$c(w, y) = \min_{x^2} c^s(w^1, w^2, y, x^2) = \min_{x^2} [c^v(w^1, y, x^2) + w^2 x^2]. \quad (3.11)$$

If producers cannot instantly adjust the quantity of the some inputs, estimating a short-run cost function with some quasi-fixed input quantities might be more appropriate than estimating a (long-run) cost function which assumes that all input quantities can be adjusted instantly.

3.1.6 Cost functions with multiple outputs

The cost function with multiple outputs is very similar to the cost function with a single output (3.2):

$$c(w, y) = \min_x \sum_i w_i x_i, \text{ s.t. } (x, y) \in T, \quad (3.12)$$

where $w = (w_1, \dots, w_N)'$ and $x = (x_1, \dots, x_N)'$ are vectors of input prices and inputs quantities, respectively, as defined before, T is again the technology set, and $y = (y_1, \dots, y_M)'$ is a vector of M output quantities. The multiple-output cost function $c(w, y)$ returns the minimal (total) cost that is required to produce the output quantities $y = (y_1, \dots, y_M)'$ given input prices $w =$

$(w_1, \dots, w_N)'$.

The properties of the multiple-output cost function are basically the same as the properties of the single-output cost function that are described in section 3.1.2. However, the monotonicity property regarding the output quantity applies to all output quantities, i.e.:

- $c(w, y)$ is non-decreasing in all y , i.e. $\partial c(w, y)/\partial y_i \geq 0 \forall i = 1, \dots, M$

and it has one additional property:

- $c(w, y)$ is quasi-convex in output quantities $\Rightarrow$ concave iso-cost curves for different output-output combinations.

In case of multiple outputs, the cost flexibility is:

$$\sum_{i=1}^M \frac{\partial c(w, y)}{\partial y_i} \frac{y_i}{c(w, y)}. \quad (3.13)$$

It indicates by how many percent the (minimum) total cost changes if all output quantities are increased by one percent.

As in the single-input case, the elasticity of size is the inverse of the cost flexibility:

$$\epsilon^*(w, y) = \left(\sum_{i=1}^M \frac{\partial c(w, y)}{\partial y_i} \frac{y_i}{c(w, y)} \right)^{-1}. \quad (3.14)$$

It indicates by how many percent all outputs can be (proportionally) increased if the (minimum) total cost is increased by one percent.

3.1.7 Suitability of the cost function for econometric applications

Given the microeconomic theory of the cost function and the assumptions of standard econometric methods such as ordinary least squares (OLS), it is appropriate to estimate a cost function with these econometric methods if several conditions are fulfilled. The most important and relevant conditions are:

1. The inputs and outputs are rather similar across all firms in the data set (within each input or output category).
2. The production conditions are rather similar for all firms in the data set (unless the empirical specification appropriately accounts for these differences).
3. All firms in the data set minimize their costs to produce the given output quantities given the input prices (note: this implies exogeneously given output quantities and input prices).
4. All input quantities must be “variable”, i.e., must be adjustable by the producer from one observational period to the next observational period (e.g., from year to year in case of annual data).

5. No firm in the data set has all output quantities equal to zero but has strictly positive costs (i.e., the “no fixed costs” assumption must be fulfilled at all observations).
6. The functional form specification of the estimated cost function is sufficiently similar to the ‘true’ cost function in the ‘real world’.
7. Random / representative sample of the firms to be analysed.
8. The data set contains variables that indicate—in reasonably precise ways—the quantities of all relevant outputs, the prices of all relevant inputs, and all relevant ‘environmental’ variables that can be used for controlling for differences in production conditions.
9. All output quantities and input prices (i.e., all explanatory variables) are uncorrelated with the error term, which is fulfilled, e.g., if the output quantities are exogenously given (e.g., by the demand for these outputs if the output prices are exogeneously given or by a non-tradable quota for the outputs) and the firms have no market power and, thus, are price-takers on the input markets.
10. There must be sufficient variation in the input prices across observations, which could particularly be a problem when using a cross-sectional data set (Quiggin and Bui-Lan, 1984), because this requires that the “law of one price” does not hold.

3.2 Cobb-Douglas cost function

3.2.1 Specification

We start with estimating a Cobb-Douglas cost function. It has the following specification:

$$c = A \left(\prod_i w_i^{\alpha_i} \right) y^{\alpha_y} \quad (3.15)$$

This function can be linearized to

$$\ln c = \alpha_0 + \sum_i \alpha_i \ln w_i + \alpha_y \ln y \quad (3.16)$$

with $\alpha_0 = \ln A$.

3.2.2 Estimation

The linearized Cobb-Douglas cost function can be estimated by OLS:

```
> costCD <- lm( log( cost ) ~ log( pCap ) + log( pLab ) + log( pMat ) + log( qOut ),
+   data = dat )
> summary( costCD )
```

Call:

```
lm(formula = log(cost) ~ log(pCap) + log(pLab) + log(pMat) +
    log(qOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.77663	-0.23243	-0.00031	0.24439	0.74339

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	6.75383	0.40673	16.605	< 2e-16 ***
log(pCap)	0.07437	0.04878	1.525	0.12969
log(pLab)	0.46486	0.14694	3.164	0.00193 **
log(pMat)	0.48642	0.08112	5.996	1.74e-08 ***
log(qOut)	0.37341	0.03072	12.154	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3395 on 135 degrees of freedom

Multiple R-squared: 0.6884, Adjusted R-squared: 0.6792

F-statistic: 74.56 on 4 and 135 DF, p-value: < 2.2e-16

3.2.3 Properties

As the coefficients of the (logarithmic) input prices are all non-negative, this cost function is monotonically non-decreasing in input prices. Furthermore, the coefficient of the (logarithmic) output quantity is non-negative so that this cost function is monotonically non-decreasing in output quantities. The Cobb-Douglas cost function always implies no fixed costs, as the costs are always zero if the output quantity is zero. Given that $A = \exp(\alpha_0)$ is always positive, all Cobb-Douglas cost functions that are based on its (estimated) linearized version fulfill the non-negativity condition.

Finally, we check if the Cobb-Douglas cost function is positive linearly homogeneous in input prices. This condition is fulfilled if

$$t c(w, y) = c(t w, y) \quad (3.17)$$

$$\ln(tc) = \alpha_0 + \sum_i \alpha_i \ln(t w_i) + \alpha_y \ln y \quad (3.18)$$

$$\ln t + \ln c = \alpha_0 + \sum_i \alpha_i \ln t + \sum_i \alpha_i \ln w_i + \alpha_y \ln y \quad (3.19)$$

$$\ln c + \ln t = \alpha_0 + \ln t \sum_i \alpha_i + \sum_i \alpha_i \ln w_i + \alpha_y \ln y \quad (3.20)$$

$$\ln c + \ln t = \ln c + \ln t \sum_i \alpha_i \quad (3.21)$$

$$\ln t = \ln t \sum_i \alpha_i \quad (3.22)$$

$$1 = \sum_i \alpha_i \quad (3.23)$$

Hence, the homogeneity condition is only fulfilled if the coefficients of the (logarithmic) input prices sum up to one. As they sum up to 1.03 the homogeneity condition is not fulfilled in our estimated model.

3.2.4 Estimation with linear homogeneity in input prices imposed

In order to estimate a Cobb-Douglas cost function with linear homogeneity imposed, we rearrange the homogeneity condition to get

$$\alpha_N = 1 - \sum_{i=1}^{N-1} \alpha_i \quad (3.24)$$

and replace α_N in the cost function (3.16) by the right-hand side of the above equation:

$$\ln c = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln w_i + \left(1 - \sum_{i=1}^{N-1} \alpha_i\right) \ln w_N + \alpha_y \ln y \quad (3.25)$$

$$\ln c = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i (\ln w_i - \ln w_N) + \ln w_N + \alpha_y \ln y \quad (3.26)$$

$$\ln c - \ln w_N = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i (\ln w_i - \ln w_N) + \alpha_y \ln y \quad (3.27)$$

$$\ln \frac{c}{w_N} = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln \frac{w_i}{w_N} + \alpha_y \ln y \quad (3.28)$$

This Cobb-Douglas cost function with linear homogeneity in input prices imposed can be estimated by following command:

```
> costCDHom <- lm( log( cost / pMat ) ~ log( pCap / pMat ) + log( pLab / pMat ) +
+   log( qOut ), data = dat )
> summary( costCDHom )
```

Call:

```
lm(formula = log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) +
    log(qOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-----	----	--------	----	-----

3 Dual Approach: Cost Functions

```
-0.77096 -0.23022 -0.00154  0.24470  0.74688
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	6.75288	0.40522	16.665	< 2e-16 ***
log(pCap/pMat)	0.07241	0.04683	1.546	0.124
log(pLab/pMat)	0.44642	0.07949	5.616	1.06e-07 ***
log(qOut)	0.37415	0.03021	12.384	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3383 on 136 degrees of freedom

Multiple R-squared: 0.5456, Adjusted R-squared: 0.5355

F-statistic: 54.42 on 3 and 136 DF, p-value: < 2.2e-16

The coefficient of the N th (logarithmic) input price can be obtained by the homogeneity condition (3.24). Hence, the estimate of α_{Mat} is 0.4812 in our model.

As there is no theory that says which input price should be taken for the normalization/deflation, it is desirable that the estimation results do not depend on the price that is used for the normalization/deflation. This desirable property is fulfilled for the Cobb-Douglas cost function and we can verify this by re-estimating the cost function, while using a different input price for the normalization/deflation, e.g. capital:

```
> costCDHomCap <- lm( log( cost / pCap ) ~ log( pLab / pCap ) + log( pMat / pCap ) +
+   log( qOut ), data = dat )
> summary( costCDHomCap )
```

Call:

```
lm(formula = log(cost/pCap) ~ log(pLab/pCap) + log(pMat/pCap) +
    log(qOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.77096	-0.23022	-0.00154	0.24470	0.74688

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	6.75288	0.40522	16.665	< 2e-16 ***
log(pLab/pCap)	0.44642	0.07949	5.616	1.06e-07 ***
log(pMat/pCap)	0.48117	0.07285	6.604	8.26e-10 ***
log(qOut)	0.37415	0.03021	12.384	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3383 on 136 degrees of freedom

Multiple R-squared: 0.8168, Adjusted R-squared: 0.8128

F-statistic: 202.2 on 3 and 136 DF, p-value: < 2.2e-16

The results are identical to the results from the Cobb-Douglas cost function with the price of materials used for the normalization/deflation. The coefficient of the (logarithmic) capital price can be obtained by the homogeneity condition (3.24). Hence, the estimate of α_{Cap} is 0.0724 in our model with the capital price as numéraire, which is identical to the corresponding estimate from the model with the price of materials as numéraire. The following code compares all coefficients obtained in the estimation with the material price as numéraire with the corresponding coefficients obtained with the capital price as numéraire:

```
> all.equal(
+   c( coef( costCDHom )[1:3], 1 - sum( coef( costCDHom )[2:3] ),
+     coef( costCDHom )[4] ),
+   c( coef( costCDHomCap )[1], 1 - sum( coef( costCDHomCap )[2:3] ),
+     coef( costCDHomCap )[2:4] ),
+   check.attributes = FALSE )
```

```
[1] TRUE
```

Both models even have identical residuals:

```
> all.equal( residuals( costCDHom ), residuals( costCDHomCap ) )
```

```
[1] TRUE
```

However, as the two models have different dependent variables (c/p_{Mat} and c/p_{Cap}), the R^2 -values differ between the two models.

We can test the restriction for imposing linear homogeneity in input prices, e.g. by a Wald test or a likelihood ratio test. As the models without and with homogeneity imposed (`costCD` and `costCDHom`) have different dependent variables (c and c/p_{Mat}), we cannot use the function `waldtest` for conducting the Wald test but we have to use the function `linearHypothesis` (package `car`) and specify the homogeneity restriction manually:

```
> library( "car" )
> linearHypothesis( costCD, "log(pCap) + log(pLab) + log(pMat) = 1" )
```

Linear hypothesis test:

```
log(pCap) + log(pLab) + log(pMat) = 1
```

Model 1: restricted model

Model 2: $\log(\text{cost}) \sim \log(\text{pCap}) + \log(\text{pLab}) + \log(\text{pMat}) + \log(\text{qOut})$

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	136	15.563				
2	135	15.560	1	0.0025751	0.0223	0.8814

```
> lrtest( costCDHom, costCD )
```

Likelihood ratio test

Model 1: $\log(\text{cost}/\text{pMat}) \sim \log(\text{pCap}/\text{pMat}) + \log(\text{pLab}/\text{pMat}) + \log(\text{qOut})$

Model 2: $\log(\text{cost}) \sim \log(\text{pCap}) + \log(\text{pLab}) + \log(\text{pMat}) + \log(\text{qOut})$

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	5	-44.878			
2	6	-44.867	1	0.0232	0.879

These two tests clearly show that the data do not contradict linear homogeneity in input prices.

Alternatively, one can test the homogeneity restriction by rearranging the hypothesis $\sum_{i=1}^N \alpha_i = 1$ to $\delta = 0$ with $\delta = \sum_{i=1}^N \alpha_i - 1$. The definition of the auxiliary variable δ can be re-arranged to:

$$\alpha_N = 1 - \sum_{i=1}^{N-1} \alpha_i + \delta \quad (3.29)$$

Replacing α_N in equation (3.16) by the right-hand side of equation (3.29), we get (similarly to the derivation in equations (3.25)–(3.28)):

$$\ln \frac{c}{w_N} = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln \frac{w_i}{w_N} + \delta \ln w_N + \alpha_y \ln y \quad (3.30)$$

This equation can be estimated by OLS and a standard t-test can be used to test the restriction $\delta = 0$:

```
> summary( lm( log( cost / pMat ) ~ log( pCap / pMat ) + log( pLab / pMat ) +
+   log( pMat ) + log( qOut ), data = dat ) )
```

Call:

```
lm(formula = log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) +
    log(pMat) + log(qOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-----	----	--------	----	-----

-0.77663 -0.23243 -0.00031 0.24439 0.74339

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	6.75383	0.40673	16.605	< 2e-16 ***
log(pCap/pMat)	0.07437	0.04878	1.525	0.12969
log(pLab/pMat)	0.46486	0.14694	3.164	0.00193 **
log(pMat)	0.02565	0.17159	0.149	0.88140
log(qOut)	0.37341	0.03072	12.154	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3395 on 135 degrees of freedom

Multiple R-squared: 0.5456, Adjusted R-squared: 0.5322

F-statistic: 40.53 on 4 and 135 DF, p-value: < 2.2e-16

This test is identical to the Wald test conducted above and hence, gives exactly the same P-value.

3.2.5 Checking concavity in input prices

The last property that we have to check is the concavity in input prices. A continuous and twice continuously differentiable function is concave, if its Hessian matrix is negative semidefinite. A necessary and sufficient condition for negative semidefiniteness of a symmetric matrix is that all its i th-order principal minors (not only its leading principal minors) are non-positive for i being odd and non-negative for i being even for all $i \in \{1, \dots, N\}$ (see section 1.5.5). The first derivatives of the Cobb-Douglas cost function with respect to the input prices are:

$$\frac{\partial c}{\partial w_i} = \frac{\partial \ln c}{\partial \ln w_i} \frac{c}{w_i} = \alpha_i \frac{c}{w_i} \quad (3.31)$$

Now, we can calculate the second derivatives as derivatives of the first derivatives (3.31):

$$\frac{\partial^2 c}{\partial w_i \partial w_j} = \frac{\partial}{\partial w_j} \left(\frac{\partial c}{\partial w_i} \right) = \frac{\partial}{\partial w_j} \left(\alpha_i \frac{c}{w_i} \right) \quad (3.32)$$

$$= \frac{\alpha_i}{w_i} \frac{\partial c}{\partial w_j} - \delta_{ij} \alpha_i \frac{c}{w_i^2} \quad (3.33)$$

$$= \frac{\alpha_i}{w_i} \alpha_j \frac{c}{w_j} - \delta_{ij} \alpha_i \frac{c}{w_i^2} \quad (3.34)$$

$$= \alpha_i (\alpha_j - \delta_{ij}) \frac{c}{w_i w_j}, \quad (3.35)$$

where δ_{ij} (again) denotes Kronecker's delta (2.95). Alternative, the second derivatives of the Cobb-Douglas cost function with respect to the input prices can be written as:

$$\frac{\partial^2 c}{\partial w_i \partial w_j} = \frac{f_i f_j}{c} - \delta_{ij} \frac{f_i}{w_i}, \quad (3.36)$$

where $f_i = \partial c / \partial w_i$ indicates the first derivative.

We start with checking concavity in input prices of the Cobb-Douglas cost function without homogeneity imposed. As argued in section 2.4.11.1, we do the calculations with the predicted dependent variables rather than with the observed dependent variables.² We can use following command to obtain the total costs which are predicted by the Cobb-Douglas cost function without homogeneity imposed:

```
> dat$costCD <- exp( fitted( costCD ) )
```

To simplify the calculations, we define short-cuts for the coefficients:

```
> cCap <- coef( costCD )[ "log(pCap)" ]
> cLab <- coef( costCD )[ "log(pLab)" ]
> cMat <- coef( costCD )[ "log(pMat)" ]
```

Using these coefficients, we compute the second derivatives of our estimated Cobb-Douglas cost function:

```
> dat$hCapCap <- cCap * ( cCap - 1 ) * dat$costCD / dat$pCap^2
> dat$hLabLab <- cLab * ( cLab - 1 ) * dat$costCD / dat$pLab^2
> dat$hMatMat <- cMat * ( cMat - 1 ) * dat$costCD / dat$pMat^2
> dat$hCapLab <- cCap * cLab * dat$costCD / ( dat$pCap * dat$pLab )
> dat$hCapMat <- cCap * cMat * dat$costCD / ( dat$pCap * dat$pMat )
> dat$hLabMat <- cLab * cMat * dat$costCD / ( dat$pLab * dat$pMat )
```

Now, we prepare the Hessian matrix for the first observation:

```
> hessian <- matrix( NA, nrow = 3, ncol = 3 )
> hessian[ 1, 1 ] <- dat$hCapCap[1]
> hessian[ 2, 2 ] <- dat$hLabLab[1]
> hessian[ 3, 3 ] <- dat$hMatMat[1]
> hessian[ 1, 2 ] <- hessian[ 2, 1 ] <- dat$hCapLab[1]
> hessian[ 1, 3 ] <- hessian[ 3, 1 ] <- dat$hCapMat[1]
> hessian[ 2, 3 ] <- hessian[ 3, 2 ] <- dat$hLabMat[1]
> print( hessian )
```

²Please note that the selection of c has no effect on the test for concavity, because all elements of the Hessian matrix include c as a multiplicative term and c is always positive so that the value of c does not change the sign of the principal minors and the determinant, as $|c \cdot M| = c \cdot |M|$, where M denotes a quadratic matrix, c denotes a scalar, and the two vertical bars denote the determinant function.

	[,1]	[,2]	[,3]
[1,]	-5031.9274	7323.804	775.3358
[2,]	7323.8043	-152736.317	14046.1549
[3,]	775.3358	14046.155	-1570.0447

As all diagonal elements of this Hessian matrix are non-positive, the conditions regarding the first-order principal minors are all fulfilled. In the following, we calculate the three second-order principal minors and the determinant (third-order principal minor):

```
> det( hessian[-3,-3] )
```

```
[1] 714919939
```

```
> det( hessian[-2,-2] )
```

```
[1] 7299205
```

```
> det( hessian[-1,-1] )
```

```
[1] 42508376
```

```
> det( hessian )
```

```
[1] 121651514835
```

As all three second-order principal minors are non-negative, the conditions regarding the second-order principal minors are all fulfilled. However, the third-order principal minor is strictly positive, which indicates that this Hessian matrix is not negative semidefinite and consequently, the Cobb-Douglas cost function is not concave at the first observation.³

We can check the semidefiniteness of a matrix more conveniently with the command `semidefiniteness` (package *miscTools*), which (by default) checks the signs of the principal minors and returns a logical value indicating whether the necessary and sufficient conditions for negative or positive semidefiniteness are fulfilled:

```
> semidefiniteness( hessian, positive = FALSE )
```

```
[1] FALSE
```

In the following, we will check whether concavity in input prices is fulfilled at each observation in the sample by stacking the Hessian matrices of all observations upon each other to a three-dimensional array:

³Please note that this Hessian matrix is not positive semidefinite either, because the first-order principal minors of this Hessian are negative, while positive semidefiniteness requires that all principal minors are non-negative. Hence, the Cobb-Douglas cost function is neither concave nor convex at the first observation.

```
> hessianCD <- array( NA, c( 3, 3, nrow( dat ) ) )
> hessianCD[ 1, 1, ] <- dat$hCapCap
> hessianCD[ 2, 2, ] <- dat$hLabLab
> hessianCD[ 3, 3, ] <- dat$hMatMat
> hessianCD[ 1, 2, ] <- hessianCD[ 2, 1, ] <- dat$hCapLab
> hessianCD[ 1, 3, ] <- hessianCD[ 3, 1, ] <- dat$hCapMat
> hessianCD[ 2, 3, ] <- hessianCD[ 3, 2, ] <- dat$hLabMat
> dat$concaveCD <- apply( hessianCD, 3, semidefiniteness,
+   positive = FALSE )
> table( dat$concaveCD )
```

FALSE

140

This shows that our Cobb-Douglas cost function without linear homogeneity imposed is concave in input prices not at a single observation.

Now, we will check, whether our Cobb-Douglas cost function with linear homogeneity imposed is concave in input prices. Again, we obtain the predicted total costs:

```
> dat$costCDHom <- exp( fitted( costCDHom ) ) * dat$pMat
```

We create short-cuts for the estimated coefficients:

```
> chCap <- coef( costCDHom )[ "log(pCap/pMat)" ]
> chLab <- coef( costCDHom )[ "log(pLab/pMat)" ]
> chMat <- 1 - chCap - chLab
```

We compute the second derivatives:

```
> dat$hhCapCap <- chCap * ( chCap - 1 ) * dat$costCDHom / dat$pCap^2
> dat$hhLabLab <- chLab * ( chLab - 1 ) * dat$costCDHom / dat$pLab^2
> dat$hhMatMat <- chMat * ( chMat - 1 ) * dat$costCDHom / dat$pMat^2
> dat$hhCapLab <- chCap * chLab * dat$costCDHom /
+   ( dat$pCap * dat$pLab )
> dat$hhCapMat <- chCap * chMat * dat$costCDHom /
+   ( dat$pCap * dat$pMat )
> dat$hhLabMat <- chLab * chMat * dat$costCDHom /
+   ( dat$pLab * dat$pMat )
```

And we prepare the Hessian matrix for the first observation:

```
> hessianHom <- matrix( NA, nrow = 3, ncol = 3 )
> hessianHom[ 1, 1 ] <- dat$hhCapCap[1]
> hessianHom[ 2, 2 ] <- dat$hhLabLab[1]
```

3 Dual Approach: Cost Functions

```
> hessianHom[ 3, 3 ] <- dat$hhMatMat[1]
> hessianHom[ 1, 2 ] <- hessianHom[ 2, 1 ] <- dat$hhCapLab[1]
> hessianHom[ 1, 3 ] <- hessianHom[ 3, 1 ] <- dat$hhCapMat[1]
> hessianHom[ 2, 3 ] <- hessianHom[ 3, 2 ] <- dat$hhLabMat[1]
> print( hessianHom )
```

```
      [,1]      [,2]      [,3]
[1,] -4901.0204  6835.826  745.4417
[2,]  6835.8256 -151446.932 13318.1722
[3,]   745.4417  13318.172 -1566.0312
```

As all diagonal elements of this Hessian matrix are non-positive, the conditions regarding the first-order principal minors are all fulfilled. In the following, we calculate the three second-order principal minors and the determinant (third-order principal minor):

```
> det( hessianHom[-3,-3] )
```

```
[1] 695515989
```

```
> det( hessianHom[-2,-2] )
```

```
[1] 7119468
```

```
> det( hessianHom[-1,-1] )
```

```
[1] 59796915
```

```
> det( hessianHom )
```

```
[1] -0.0003162841
```

All three second-order principal minors are non-negative and the third-order principal minor is close to zero, where it is negative on some computers but positive on other computers. As Hessian matrices of linear homogeneous functions are always singular, it is expected that the determinant of the Hessian matrix (the N th-order principal minor) is zero. However, the computed determinant of our Hessian matrix is not exactly zero due to rounding errors, which are unavoidable on digital computers. Given that the determinant of the Hessian matrix of our Cobb-Douglas cost function with linear homogeneity imposed should always be zero, the condition for negative semidefiniteness regarding the N th-order principal minor (sign of the determinant of the Hessian matrix) should always be fulfilled. Consequently, we can conclude that our Cobb-Douglas cost function with linear homogeneity imposed is concave in input prices at the first observation. Again, we can check the semidefiniteness of a matrix more conveniently with the command `semidefiniteness`:

```
> semidefiniteness( hessianHom, positive = FALSE )
```

```
[1] TRUE
```

In the following, we will check whether concavity in input prices is fulfilled at each observation in the sample:

```
> hessianCDHom <- array( NA, c( 3, 3, nrow( dat) ) )
> hessianCDHom[ 1, 1, ] <- dat$hhCapCap
> hessianCDHom[ 2, 2, ] <- dat$hhLabLab
> hessianCDHom[ 3, 3, ] <- dat$hhMatMat
> hessianCDHom[ 1, 2, ] <- hessianCDHom[ 2, 1, ] <- dat$hhCapLab
> hessianCDHom[ 1, 3, ] <- hessianCDHom[ 3, 1, ] <- dat$hhCapMat
> hessianCDHom[ 2, 3, ] <- hessianCDHom[ 3, 2, ] <- dat$hhLabMat
> dat$concaveCDHom <-
+   apply( hessianCDHom, 3, semidefiniteness, positive = FALSE )
> table( dat$concaveCDHom )
```

```
TRUE
```

```
140
```

This result indicates that the concavity condition is fulfilled at all observations.

In fact, all Cobb-Douglas cost functions that are non-decreasing and linearly homogeneous in all input prices are always concave (e.g. Coelli, 1995, p. 266).⁴

3.2.6 Homotheticity

In order to check whether the Cobb-Douglas cost function as specified in equations (3.15) and (3.16) is homothetic, we check whether we can decompose it as done in equation (3.5). This decomposition of the Cobb-Douglas cost function is always possible with:

$$c_0(c_1, c_2) = A c_1(w) c_2(y) \quad (3.37)$$

$$c_1(w) = \prod_i w_i^{\alpha_i} \quad (3.38)$$

$$c_2(y) = y^{\alpha_y}. \quad (3.39)$$

Hence, we can conclude that the Cobb-Douglas cost function is always homothetic.

⁴A proof and further details are available at <http://www.econ.ucsb.edu/~tedb/Courses/GraduateTheoryUCSB/concavity.pdf>.

3.2.7 Optimal input quantities

Shepard's Lemma says that the cost-minimizing input quantities are equal to the partial derivatives of the cost functions with respect to the respective input prices. Hence, we can use equation (3.31) to calculate the cost-minimizing input quantities predicted by a Cobb-Douglas cost function:

$$x_i = \frac{\partial c}{\partial w_i} = \alpha_i \frac{c}{w_i} \quad (3.40)$$

The following commands compare the observed input quantities with the cost-minimizing input quantities that are predicted by our estimated Cobb-Douglas cost function with linear homogeneity imposed:

```
> compPlot( chCap * dat$costCDHom / dat$pCap, dat$qCap )
> compPlot( chLab * dat$costCDHom / dat$pLab, dat$qLab )
> compPlot( chMat * dat$costCDHom / dat$pMat, dat$qMat )
> compPlot( chCap * dat$costCDHom / dat$pCap, dat$qCap, log = "xy" )
> compPlot( chLab * dat$costCDHom / dat$pLab, dat$qLab, log = "xy" )
> compPlot( chMat * dat$costCDHom / dat$pMat, dat$qMat, log = "xy" )
```

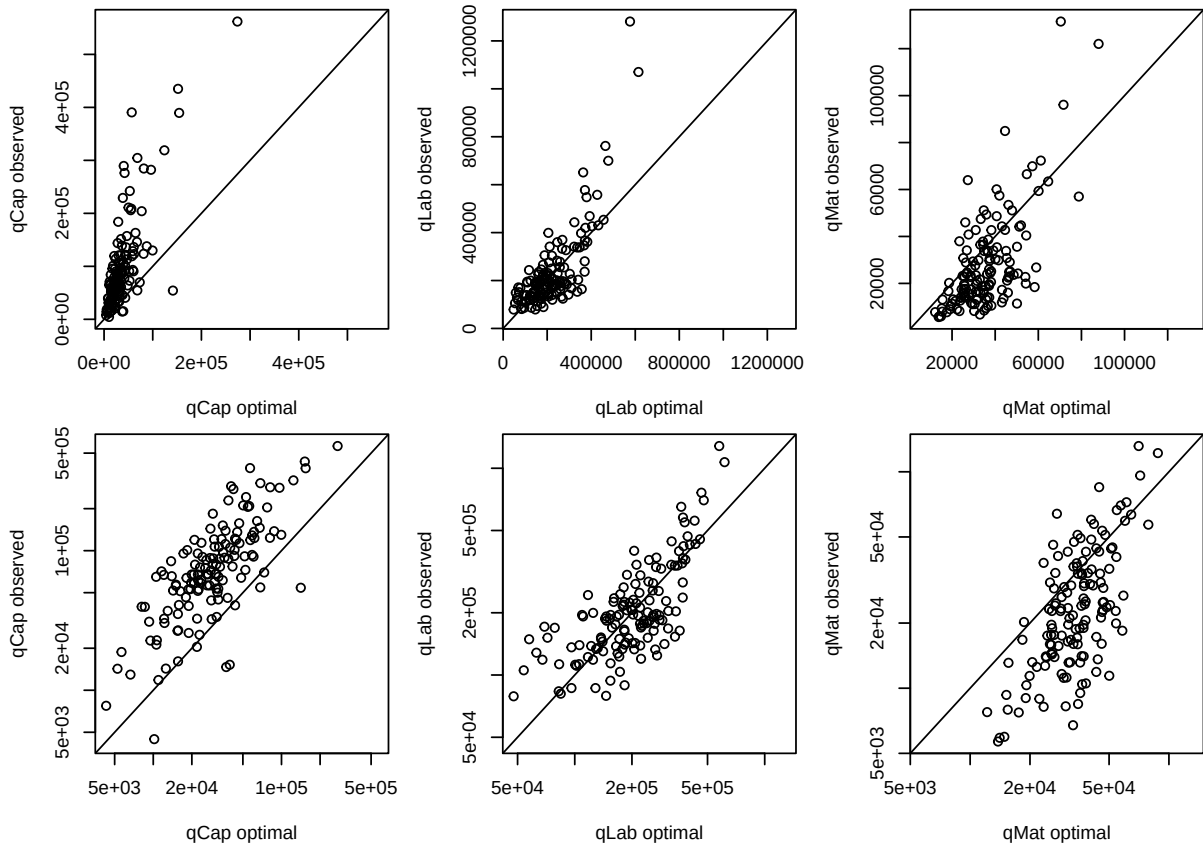


Figure 3.1: Observed and optimal input quantities

The resulting graphs are shown in figure 3.1. These results confirm results based on the production function: most firms should increase the use of materials and decrease the use of capital goods.

3.2.8 Optimal cost shares

Given Shepard's Lemma, the optimal cost shares derived from a Cobb-Douglas cost function are equal to the coefficients of the (logarithmic) input prices:

$$\alpha_i = \frac{\partial \ln c(w, y)}{\partial \ln w_i} = \frac{\partial c(w, y)}{\partial w_i} \frac{w_i}{c(w, y)} = x_i(w, y) \frac{w_i}{c(w, y)} = \frac{w_i x_i(w, y)}{c(w, y)} = s_i(w, y), \quad (3.41)$$

where $s_i = w_i x_i / c$ are the cost shares.

The following commands draw histograms of the observed cost shares and compare them to the optimal cost shares, which are predicted by our Cobb-Douglas cost function with linear homogeneity imposed:

```
> hist( dat$pCap * dat$qCap / dat$cost )
> abline( v = chCap, lwd = 3 )
> hist( dat$pLab * dat$qLab / dat$cost )
> abline( v = chLab, lwd = 3 )
> hist( dat$pMat * dat$qMat / dat$cost )
> abline( v = chMat, lwd = 3 )
```

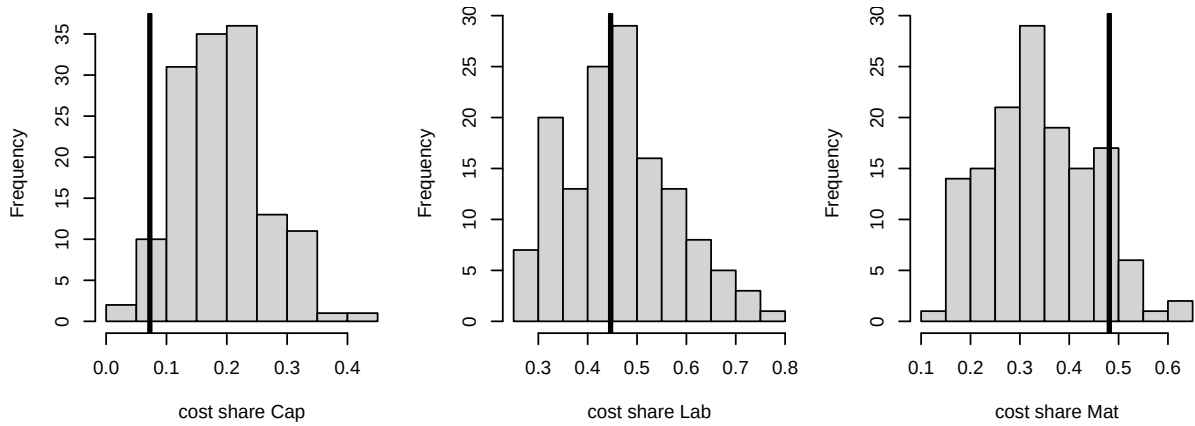


Figure 3.2: Observed and optimal costs shares

The resulting graphs are shown in figure 3.2. These results confirm earlier results: most firms should increase the use of materials and decrease the use of capital goods.

3.2.9 Derived input demand functions

Shepard's Lemma says that the partial derivatives of the cost functions with respect to the input prices are the conditional input demand functions. Therefore, the input demand functions based

3 Dual Approach: Cost Functions

on a Cobb-Douglas cost function are equal to the right-hand side of equation (3.31):

$$x_i(w, y) = \frac{\partial c(w, y)}{\partial w_i} = \alpha_i \frac{c(w, y)}{w_i} \quad (3.42)$$

Input demand functions should be homogeneous of degree zero in input prices:

$$x_i(t w, y) = x_i(w, y) \quad (3.43)$$

This condition is fulfilled for the input demand functions derived from any linearly homogeneous Cobb-Douglas cost function:

$$x_i(t w, y) = \alpha_i \frac{c(t w, y)}{t w_i} = \alpha_i \frac{t c(w, y)}{t w_i} = \alpha_i \frac{c(w, y)}{w_i} = x_i(w, y) \quad (3.44)$$

Furthermore, input demand functions should be symmetric with respect to input prices:

$$\frac{\partial x_i(w, y)}{\partial w_j} = \frac{\partial x_j(w, y)}{\partial w_i} \quad (3.45)$$

This condition is fulfilled for the input demand functions derived from any Cobb-Douglas cost function:

$$\frac{\partial x_i(w, y)}{\partial w_j} = \frac{\alpha_i}{w_i} \frac{\partial c(w, y)}{\partial w_j} = \frac{\alpha_i}{w_i} \alpha_j \frac{c(w, y)}{w_j} = \frac{\alpha_i \alpha_j}{w_i w_j} c(w, y) \quad \forall i \neq j \quad (3.46)$$

$$\frac{\partial x_j(w, y)}{\partial w_i} = \frac{\alpha_j}{w_j} \frac{\partial c(w, y)}{\partial w_i} = \frac{\alpha_j}{w_j} \alpha_i \frac{c(w, y)}{w_i} = \frac{\alpha_i \alpha_j}{w_i w_j} c(w, y) \quad \forall i \neq j \quad (3.47)$$

Finally, input demand functions should fulfill the negativity condition:

$$\frac{\partial x_i(w, y)}{\partial w_i} \leq 0 \quad (3.48)$$

This condition is fulfilled for the input demand functions derived from any linearly homogeneous Cobb-Douglas cost function that is monotonically increasing in all input prices (as this implies $0 \leq \alpha_i \leq 1$):

$$\frac{\partial x_i(w, y)}{\partial w_i} = \frac{\alpha_i}{w_i} \frac{\partial c(w, y)}{\partial w_i} - \alpha_i \frac{c(w, y)}{w_i^2} \quad (3.49)$$

$$= \frac{\alpha_i}{w_i} \alpha_i \frac{c(w, y)}{w_i} - \alpha_i \frac{c(w, y)}{w_i^2} \quad (3.50)$$

$$= \alpha_i \frac{c(w, y)}{w_i^2} (\alpha_i - 1) \leq 0 \quad (3.51)$$

3.2.10 Derived input demand elasticities

Based on the derived input demand functions (3.42), we can derive the conditional input demand elasticities:

$$\epsilon_{ij}(w, y) = \frac{\partial x_i(w, y)}{\partial w_j} \frac{w_j}{x_i(w, y)} \quad (3.52)$$

$$= \frac{\alpha_i}{w_i} \frac{\partial c(w, y)}{\partial w_j} \frac{w_j}{x_i(w, y)} - \delta_{ij} \alpha_i \frac{c(w, y)}{w_j^2} \frac{w_j}{x_i(w, y)} \quad (3.53)$$

$$= \frac{\alpha_i}{w_i} \alpha_j \frac{c(w, y)}{w_j} \frac{w_j}{x_i(w, y)} - \delta_{ij} \alpha_i \frac{c(w, y)}{w_i x_i(w, y)} \quad (3.54)$$

$$= \alpha_i \alpha_j \frac{c(w, y)}{w_i x_i(w, y)} - \delta_{ij} \frac{\alpha_i}{s_i(w, y)} \quad (3.55)$$

$$= \frac{\alpha_i \alpha_j}{s_i(w, y)} - \delta_{ij} \frac{\alpha_i}{s_i(w, y)} \quad (3.56)$$

$$= \alpha_j - \delta_{ij} \quad (3.57)$$

$$\epsilon_{iy}(w, y) = \frac{\partial x_i(w, y)}{\partial y} \frac{y}{x_i(w, y)} \quad (3.58)$$

$$= \frac{\partial c(w, y)}{\partial y} \frac{\alpha_i}{w_i} \frac{y}{x_i(w, y)} \quad (3.59)$$

$$= \alpha_y \frac{c(w, y)}{y} \frac{\alpha_i}{w_i} \frac{y}{x_i(w, y)} \quad (3.60)$$

$$= \alpha_i \alpha_y \frac{c(w, y)}{y} \frac{y}{w_i x_i(w, y)} \quad (3.61)$$

$$= \alpha_i \alpha_y \frac{c(w, y)}{w_i x_i(w, y)} \quad (3.62)$$

$$= \alpha_y \frac{\alpha_i}{s_i(w, y)} \quad (3.63)$$

$$= \alpha_y \quad (3.64)$$

All derived input demand elasticities based on our estimated Cobb-Douglas cost function with linear homogeneity imposed are presented in table 3.1. If the price of capital increases by one percent, the cost-minimizing firm will decrease the use of capital by 0.93% and increase the use of labor and materials by 0.07% each. If the price of labor increases by one percent, the cost-minimizing firm will decrease the use of labor by 0.55% and increase the use of capital and materials by 0.45% each. If the price of materials increases by one percent, the cost-minimizing firm will decrease the use of materials by 0.52% and increase the use of capital and labor by 0.48% each. If the cost-minimizing firm increases the output quantity by one percent, (s)he will increase all input quantities by 0.37%. The price elasticities derived from the Cobb-Douglas cost function with linear homogeneity imposed are rather similar to the price elasticities derived from

3 Dual Approach: Cost Functions

the Cobb-Douglas production function but the elasticities with respect to the output quantity are rather dissimilar (compare Tables 2.1 and 3.1). In theory, elasticities derived from a cost function, which corresponds to a specific production function, should be identical to elasticities which are directly derived from the production function. However, although our production function and cost function are supposed to model the same production technology, their elasticities are not the same. These differences arise from different econometric assumptions (e.g. exogeneity of explanatory variables) and the disturbance terms, which differ between both models so that the production technology is fitted differently.

Table 3.1: Conditional demand elasticities derived from Cobb-Douglas cost function (with linear homogeneity imposed)

	w_{cap}	w_{lab}	w_{mat}	y
x_{cap}	-0.93	0.45	0.48	0.37
x_{lab}	0.07	-0.55	0.48	0.37
x_{mat}	0.07	0.45	-0.52	0.37

Given Euler's theorem and the cost function's homogeneity in input prices, following condition for the price elasticities can be obtained:

$$\sum_j \epsilon_{ij} = 0 \quad \forall i \quad (3.65)$$

The input demand elasticities derived from any linearly homogeneous Cobb-Douglas cost function fulfill the homogeneity condition:

$$\sum_j \epsilon_{ij}(w, y) = \sum_j (\alpha_j - \delta_{ij}) = \sum_j \alpha_j - \sum_j \delta_{ij} = 1 - 1 = 0 \quad \forall i \quad (3.66)$$

As we computed the elasticities in table 3.1 based on the Cobb-Douglas function with linear homogeneity imposed, these conditions are fulfilled for these elasticities, e.g. for $i = cap$: $-0.93 + 0.45 + 0.48 = 0$.

It follows from the necessary conditions for the concavity of the cost function that all own-price elasticities are non-positive:

$$\epsilon_{ii} \leq 0 \quad \forall i \quad (3.67)$$

The input demand elasticities derived from any linearly homogeneous Cobb-Douglas cost function that is monotonically increasing in all input prices fulfill the negativity condition, because linear homogeneity ($\sum_i \alpha_i = 1$) and monotonicity ($\alpha_i \geq 0 \quad \forall i$) together imply that all α_i s (optimal cost shares) are between zero and one ($0 \leq \alpha_i \leq 1 \quad \forall i$):

$$\epsilon_{ii} = \alpha_i - 1 \leq 0 \quad \forall i \quad (3.68)$$

As our Cobb-Douglas function with linear homogeneity imposed fulfills the homogeneity, mono-

tonicity, and concavity condition, the own-price elasticities (on the diagonal) in table 3.1 fulfill the negativity conditions.

The symmetry condition for derived demand elasticities

$$s_i \epsilon_{ij} = s_j \epsilon_{ji} \quad \forall i, j \quad (3.69)$$

is fulfilled for any Cobb-Douglas cost function:

$$s_i \epsilon_{ij} = \alpha_i \alpha_j = \alpha_j \alpha_i = s_j \epsilon_{ji} \quad \forall i \neq j \quad (3.70)$$

$$s_i \epsilon_{ij} = s_i \epsilon_{ii} = s_j \epsilon_{jj} = s_j \epsilon_{ji} \quad \forall i = j \quad (3.71)$$

Hence, the symmetry condition is also fulfilled for the elasticities in table 3.1, e.g. $s_{cap} \epsilon_{cap,lab} = \alpha_{cap} \epsilon_{cap,lab} = 0.07 \cdot 0.45$ is equal to $s_{lab} \epsilon_{lab,cap} = \alpha_{lab} \epsilon_{lab,cap} = 0.45 \cdot 0.07$.

3.2.11 Cost flexibility and elasticity of size

The coefficient of the (logarithmic) output quantity is equal to the “cost flexibility” (3.3). A value of 0.37 (as in our estimated Cobb-Douglas cost function with linear homogeneity in input prices imposed) means that a 1% increase in the output quantity results in a cost increase of 0.37%. The “elasticity of size” is the inverse of the cost flexibility (3.4). A value of 2.67 (as derived from our estimated Cobb-Douglas cost function with linear homogeneity in input prices imposed) means that if minimum costs are increased by 1%, the output quantity increases by 2.67%.

3.2.12 Marginal costs, average costs, and total costs

Marginal costs can be calculated by

$$\frac{\partial c(w, y)}{\partial y} = \alpha_y \frac{c(w, y)}{y} \quad (3.72)$$

These marginal costs should be linearly homogeneous in input prices:

$$\frac{\partial c(t w, y)}{\partial y} = t \frac{\partial c(w, y)}{\partial y} \quad (3.73)$$

This condition is fulfilled for the marginal costs derived from a linearly homogeneous Cobb-Douglas cost function:

$$\frac{\partial c(t w, y)}{\partial y} = \alpha_y \frac{c(t w, y)}{y} = \alpha_y \frac{t c(w, y)}{y} = t \alpha_y \frac{c(w, y)}{y} = t \frac{\partial c(w, y)}{\partial y} \quad (3.74)$$

We can compute the marginal costs by following command:

```
> chOut <- coef( costCDHom )[ "log(qOut)" ]
> dat$margCost <- chOut * dat$costCDHom / dat$qOut
```

We can visualize these marginal costs with a histogram.

```
> hist( dat$margCost, 20 )
```

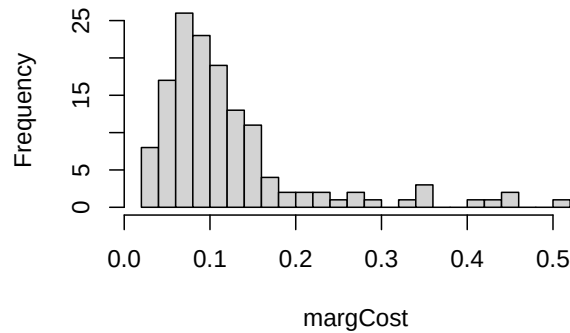


Figure 3.3: Marginal costs

The resulting graph is shown in figure 3.3. It indicates that producing one additional output unit increases the costs of most firms by around 0.08 monetary units.

Furthermore, we can check if the marginal costs are equal to the output prices, which is a first-order condition for profit maximization:

```
> compPlot( dat$pOut, dat$margCost )
> compPlot( dat$pOut, dat$margCost, log = "xy" )
```

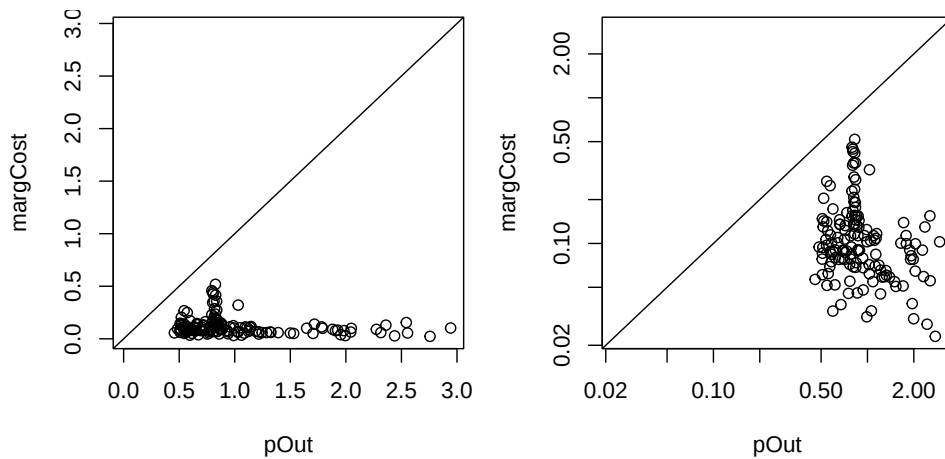


Figure 3.4: Marginal costs and output prices

The resulting graphs are shown in figure 3.4. The marginal costs of all firms are considerably smaller than their output prices. Hence, all firms would gain from increasing their output level. This is not surprising for a technology with large economies of scale.

Now, we analyze, how the marginal costs depend on the output quantity:

```

> plot( dat$qOut, dat$margCost,
+       ylim = range( c( dat$margCost, dat$costCDHom / dat$qOut ) ) )
> points( dat$qOut, dat$costCDHom / dat$qOut, pch = 4 )
> legend( "topright", pch = c( 21, 4 ),
+        legend = c( "marginal costs", "average costs" ) )
> plot( dat$qOut, dat$margCost, log = "xy",
+       ylim = range( c( dat$margCost, dat$costCDHom / dat$qOut ) ) )
> points( dat$qOut, dat$costCDHom / dat$qOut, pch = 4 )
> legend( "topright", pch = c( 21, 4 ),
+        legend = c( "marginal costs", "average costs" ) )

```

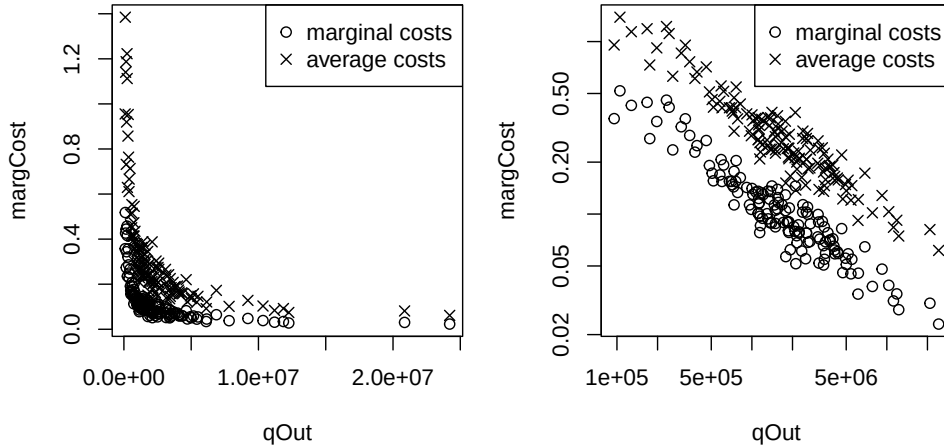


Figure 3.5: Marginal costs depending on output quantity and firm size

The resulting graphs are shown in figure 3.5. Due to the large economies of size, the marginal costs are decreasing with the output quantity.

The relation between output quantity and marginal costs in a Cobb-Douglas cost function can be analyzed by taking the first derivative of the marginal costs (3.72) with respect to the output quantity:

$$\frac{\partial MC}{\partial y} = \frac{\partial \left(\alpha_y \frac{c(w,y)}{y} \right)}{\partial y} \quad (3.75)$$

$$= \frac{\alpha_y}{y} \frac{\partial c(w,y)}{\partial y} - \alpha_y \frac{c(w,y)}{y^2} \quad (3.76)$$

$$= \frac{\alpha_y}{y} \alpha_y \frac{c(w,y)}{y} - \alpha_y \frac{c(w,y)}{y^2} \quad (3.77)$$

$$= \alpha_y \frac{c}{y^2} (\alpha_y - 1) \quad (3.78)$$

As α_y , c , and y^2 should always be positive, the marginal costs are (globally) increasing in the output quantity, if there are decreasing returns to size (i.e. $\alpha_y > 1$) and the marginal costs are

(globally) decreasing in the output quantity, if there are increasing returns to size (i.e. $\alpha_y < 1$).

Now, we illustrate our estimated model by drawing the total cost curve for output quantities between the minimum and the maximum output level in the sample, where we use the sample means of the input prices. Furthermore, we draw the average cost curve and the marginal cost curve for the above-mentioned output quantities and input prices:

```
> y <- seq( min( dat$qOut ), max( dat$qOut ), length.out = 200 )
> chInt <- coef(costCDHom)[ "(Intercept)" ]
> costs <- exp( chInt + chCap * log( mean( dat$pCap ) ) +
+   chLab * log( mean( dat$pLab ) ) + chMat * log( mean( dat$pMat ) ) +
+   chOut * log( y ) )

> plot( y, costs, type = "l" )

> # average costs
> plot( y, costs/y, type = "l" )

> # marginal costs
> lines( y, chOut * costs / y, lty = 2 )
> legend( "right", lty = c( 1, 2 ),
+   legend = c( "average costs", "marginal costs" ) )
```

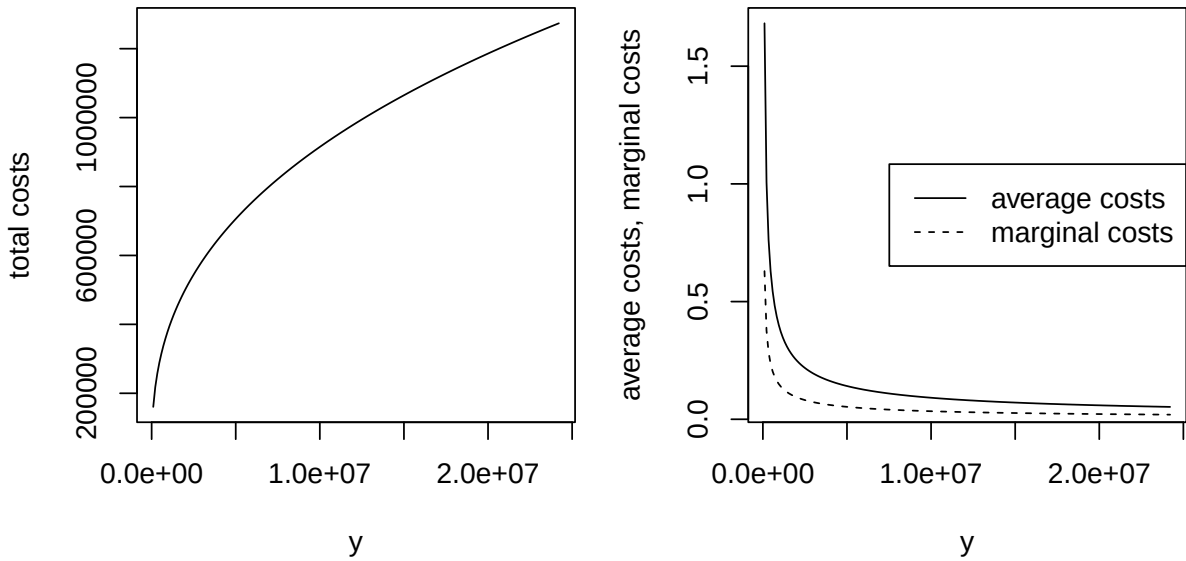


Figure 3.6: Total, marginal, and average costs

The resulting graphs are shown in figure 3.6. As the marginal costs are equal to the average costs multiplied by a fixed factor, α_y (see equation 3.72), the average cost curve and the marginal cost curve of a Cobb-Douglas cost function cannot intersect. Hence, the average costs curve based on a Cobb-Douglas cost function does not have a minimum (for finite output quantities y).

3.3 Cobb-Douglas short-run cost function

3.3.1 Specification

Given the general specification of a short-run cost function (3.9), a Cobb-Douglas short-run cost function is

$$c^v = A \left(\prod_{i \in N^1} w_i^{\alpha_i} \right) \left(\prod_{j \in N^2} x_j^{\alpha_j} \right) y^{\alpha_y}, \quad (3.79)$$

where c^v denotes the variable costs as defined in (1.3), N^1 is a vector of the indices of the variable inputs, and N^2 is a vector of the indices of the quasi-fixed inputs. The Cobb-Douglas short-run cost function can be linearized to

$$\ln c^v = \alpha_0 + \sum_{i \in N^1} \alpha_i \ln w_i + \sum_{j \in N^2} \alpha_j \ln x_j + \alpha_y \ln y \quad (3.80)$$

with $\alpha_0 = \ln A$.

3.3.2 Estimation

The following commands estimate a Cobb-Douglas short-run cost function with capital as a quasi-fixed input and summarize the results:

```
> costCDSR <- lm( log( vCost ) ~ log( pLab ) + log( pMat ) + log( qCap ) + log( qOut ),
+   data = dat )
> summary( costCDSR )
```

Call:

```
lm(formula = log(vCost) ~ log(pLab) + log(pMat) + log(qCap) +
    log(qOut), data = dat)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-0.73935	-0.20934	-0.00571	0.20729	0.71633

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	5.66013	0.42523	13.311	< 2e-16 ***
log(pLab)	0.45683	0.13819	3.306	0.00121 **
log(pMat)	0.44144	0.07715	5.722	6.50e-08 ***
log(qCap)	0.19174	0.04034	4.754	5.05e-06 ***
log(qOut)	0.29127	0.03318	8.778	6.49e-15 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3183 on 135 degrees of freedom

Multiple R-squared: 0.7265, Adjusted R-squared: 0.7183

F-statistic: 89.63 on 4 and 135 DF, p-value: < 2.2e-16

3.3.3 Properties

This short-run cost function is (significantly) increasing in the prices of the variable inputs (labor and materials) as the coefficient of the labor price (0.457) and the coefficient of the materials price (0.441) are both positive. However, this short-run cost function is not linearly homogeneous in input prices, as the coefficient of the labor price and the coefficient of the materials price do not sum up to one ($0.457 + 0.441 = 0.898$). The short-run cost function is increasing in the output quantity with a short-run cost flexibility of 0.291, which corresponds to a short-run elasticity of size of 3.433. However, this short-run cost function is increasing in the quantity of the fixed input (capital), as the corresponding coefficient is (significantly) positive (0.192) which contradicts microeconomic theory. This would mean that the apple producers could reduce variable costs (costs from labor and materials) by reducing the capital input (e.g. by destroying their apple trees and machinery), while still producing the same amount of apples. Producing the same output level with less of *all* inputs is not plausible. The violation of the monotonicity property regarding the capital input is probably caused by model misspecification. For instance, the assumption that the capital input quantity is exogenously given may not hold; the capital input quantity could be positively correlated with the error term, e.g. if poorly performing firms (with higher variable costs) use more capital than well performing firms (with lower variable costs) to produce the same output quantity.

3.3.4 Estimation with linear homogeneity in input prices imposed

We can impose linear homogeneity in the prices of the variable inputs as we did with the (long-run) cost function (see equations 3.24 to 3.28):

$$\ln \frac{c}{w_k} = \alpha_0 + \sum_{i \in N^1 \setminus k} \alpha_i \ln \frac{w_i}{w_k} + \sum_{j \in N^2} \alpha_j \ln x_j + \alpha_y \ln y \quad (3.81)$$

with $k \in N^1$. We can estimate a Cobb-Douglas short-run cost function with capital as a quasi-fixed input and linear homogeneity in input prices imposed by the command:

```
> costCDSRHom <- lm( log( vCost / pMat ) ~ log( pLab / pMat ) +
+   log( qCap ) + log( qOut ), data = dat )
> summary( costCDSRHom )
```

Call:

```
lm(formula = log(vCost/pMat) ~ log(pLab/pMat) + log(qCap) + log(qOut),
```

```
data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.78305	-0.20539	-0.00265	0.19533	0.71792

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	5.67882	0.42335	13.414	< 2e-16 ***
log(pLab/pMat)	0.53487	0.06781	7.888	9.00e-13 ***
log(qCap)	0.18774	0.03978	4.720	5.79e-06 ***
log(qOut)	0.29010	0.03306	8.775	6.33e-15 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3176 on 136 degrees of freedom

Multiple R-squared: 0.5963, Adjusted R-squared: 0.5874

F-statistic: 66.97 on 3 and 136 DF, p-value: < 2.2e-16

We can obtain the coefficient of the materials price from the homogeneity condition (3.24): $1 - 0.535 = 0.465$. We can test the homogeneity restriction by a likelihood ratio test:

```
> lrtest( costCDSRHom, costCDSR )
```

Likelihood ratio test

Model 1: log(vCost/pMat) ~ log(pLab/pMat) + log(qCap) + log(qOut)

Model 2: log(vCost) ~ log(pLab) + log(pMat) + log(qCap) + log(qOut)

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	5	-36.055			
2	6	-35.838	1	0.4356	0.5093

Given the large P -value, we can conclude that the data do not contradict the linear homogeneity in the prices of the variable inputs.

While the linear homogeneity in the prices of all variable inputs is accepted and the short-run cost function is still increasing in the output quantity and the prices of all variable inputs, the estimated short-run cost function is still increasing in the capital quantity, which contradicts microeconomic theory. Therefore, a further microeconomic analysis with this function is not reasonable.

3.4 Cobb-Douglas cost function with multiple outputs

3.4.1 Specification

The Cobb-Douglas cost function with multiple outputs is a minor extension of the Cobb-Douglas cost function with a single output as defined in equations (3.15) and (3.16):

$$c = A \left(\prod_{i=1}^N w_i^{\alpha_i} \right) \left(\prod_{i=1}^M y_i^{\beta_i} \right) \quad (3.82)$$

This function can be linearized to:

$$\ln c = \alpha_0 + \sum_{i=1}^N \alpha_i \ln w_i + \sum_{i=1}^M \beta_i \ln y_i \quad (3.83)$$

with $\alpha_0 = \ln A$.

3.4.2 Estimation

The linearized Cobb-Douglas cost function with multiple outputs can be estimated by OLS:

```
> costMultOutCD <- lm( log( cost ) ~ log( pCap ) + log( pLab ) + log( pMat )
+   + log( qApples ) + log( qOtherOut ), data = dat )
> summary( costMultOutCD )
```

Call:

```
lm(formula = log(cost) ~ log(pCap) + log(pLab) + log(pMat) +
    log(qApples) + log(qOtherOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.78785	-0.26760	-0.00666	0.23298	1.18589

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	11.85635	0.17271	68.650	< 2e-16 ***
log(pCap)	0.05709	0.05673	1.006	0.316073
log(pLab)	0.66763	0.17335	3.851	0.000181 ***
log(pMat)	0.61834	0.09208	6.715	4.88e-10 ***
log(qApples)	0.14147	0.01980	7.147	5.16e-11 ***
log(qOtherOut)	0.15432	0.03417	4.516	1.37e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3931 on 134 degrees of freedom
 Multiple R-squared: 0.5853, Adjusted R-squared: 0.5699
 F-statistic: 37.83 on 5 and 134 DF, p-value: < 2.2e-16

3.4.3 Properties

The Cobb-Douglas multiple-output cost function is never quasi-convex in output quantities because it always has convex iso-cost curves for different output-output combinations.

3.4.4 Estimation with linear homogeneity in input prices imposed

In order to estimate a Cobb-Douglas cost function for multiple outputs with linear homogeneity imposed, we re-arrange the homogeneity condition in the same way as we did for the Cobb-Douglas cost function for a single output. Hence, we can estimate it by following command:

```
> costMultOutCDHom <- lm( log( cost / pMat ) ~ log( pCap / pMat ) + log( pLab / pMat ) +
+   log( qApples ) + log( qOtherOut ), data = dat )
> summary( costMultOutCDHom )
```

Call:

```
lm(formula = log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) +
    log(qApples) + log(qOtherOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.85692	-0.27389	-0.00736	0.21797	1.23278

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	11.97385	0.16011	74.783	< 2e-16 ***
log(pCap/pMat)	0.03191	0.05526	0.577	0.565
log(pLab/pMat)	0.41381	0.09411	4.397	2.21e-05 ***
log(qApples)	0.14405	0.01989	7.243	3.02e-11 ***
log(qOtherOut)	0.14886	0.03428	4.342	2.75e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.396 on 135 degrees of freedom
 Multiple R-squared: 0.3817, Adjusted R-squared: 0.3634
 F-statistic: 20.84 on 4 and 135 DF, p-value: 2.146e-13

The coefficient of the N th (logarithmic) input price can be obtained by the homogeneity condition (3.24). Hence, the estimate of α_{Mat} is 0.5543 in our estimated Cobb-Douglas cost function.

3.4.5 Cost flexibility and elasticity of size

The cost flexibility is:

$$\sum_{i=1}^M \beta_i \quad (3.84)$$

and the elasticity of size is:

$$\epsilon^* = \left(\sum_{i=1}^M \beta_i \right)^{-1}. \quad (3.85)$$

and, thus, our estimate of the elasticity of size is:

```
> 1 / sum( coef( costMultOutCDHom )[ c( "log(qApples)", "log(qOtherOut)" ) ] )
[1] 3.414028
```

3.5 Translog cost function

3.5.1 Specification

The general specification of a Translog cost function is

$$\begin{aligned} \ln c(w, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln w_i + \alpha_y \ln y \\ & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln w_i \ln w_j + \frac{1}{2} \alpha_{yy} (\ln y)^2 \\ & + \sum_{i=1}^N \alpha_{iy} \ln w_i \ln y \end{aligned} \quad (3.86)$$

with $\alpha_{ij} = \alpha_{ji} \forall i, j$.

3.5.2 Estimation

The Translog cost function can be estimated by following command:

```
> costTL <- lm( log( cost ) ~ log( pCap ) + log( pLab ) + log( pMat ) +
+   log( qOut ) + I( 0.5 * log( pCap )^2 ) + I( 0.5 * log( pLab )^2 ) +
+   I( 0.5 * log( pMat )^2 ) + I( log( pCap ) * log( pLab ) ) +
+   I( log( pCap ) * log( pMat ) ) + I( log( pLab ) * log( pMat ) ) +
+   I( 0.5 * log( qOut )^2 ) + I( log( pCap ) * log( qOut ) ) +
+   I( log( pLab ) * log( qOut ) ) + I( log( pMat ) * log( qOut ) ),
```

3 Dual Approach: Cost Functions

```
+ data = dat )
> summary( costTL )
```

Call:

```
lm(formula = log(cost) ~ log(pCap) + log(pLab) + log(pMat) +
    log(qOut) + I(0.5 * log(pCap)^2) + I(0.5 * log(pLab)^2) +
    I(0.5 * log(pMat)^2) + I(log(pCap) * log(pLab)) + I(log(pCap) *
    log(pMat)) + I(log(pLab) * log(pMat)) + I(0.5 * log(qOut)^2) +
    I(log(pCap) * log(qOut)) + I(log(pLab) * log(qOut)) + I(log(pMat) *
    log(qOut)), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.73251	-0.18718	0.02001	0.15447	0.82858

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	25.383429	3.511353	7.229	4.26e-11 ***
log(pCap)	0.198813	0.537885	0.370	0.712291
log(pLab)	-0.024792	2.232126	-0.011	0.991156
log(pMat)	-1.244914	1.201129	-1.036	0.301992
log(qOut)	-2.040079	0.510905	-3.993	0.000111 ***
I(0.5 * log(pCap)^2)	-0.095173	0.105158	-0.905	0.367182
I(0.5 * log(pLab)^2)	-0.503168	0.943390	-0.533	0.594730
I(0.5 * log(pMat)^2)	0.529021	0.337680	1.567	0.119728
I(log(pCap) * log(pLab))	-0.746199	0.244445	-3.053	0.002772 **
I(log(pCap) * log(pMat))	0.182268	0.130463	1.397	0.164865
I(log(pLab) * log(pMat))	0.139429	0.433408	0.322	0.748215
I(0.5 * log(qOut)^2)	0.164075	0.041078	3.994	0.000110 ***
I(log(pCap) * log(qOut))	-0.028090	0.042844	-0.656	0.513259
I(log(pLab) * log(qOut))	0.007533	0.171134	0.044	0.964959
I(log(pMat) * log(qOut))	0.048794	0.092266	0.529	0.597849

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3043 on 125 degrees of freedom

Multiple R-squared: 0.7682, Adjusted R-squared: 0.7423

F-statistic: 29.59 on 14 and 125 DF, p-value: < 2.2e-16

As the Cobb-Douglas cost function is nested in the Translog cost function, we can use statistical

tests to check whether the Cobb-Douglas cost function fits the data as good as the Translog cost function:

```
> waldtest( costCD, costTL )
```

Wald test

```
Model 1: log(cost) ~ log(pCap) + log(pLab) + log(pMat) + log(qOut)
Model 2: log(cost) ~ log(pCap) + log(pLab) + log(pMat) + log(qOut) + I(0.5 *
  log(pCap)^2) + I(0.5 * log(pLab)^2) + I(0.5 * log(pMat)^2) +
  I(log(pCap) * log(pLab)) + I(log(pCap) * log(pMat)) + I(log(pLab) *
  log(pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap) * log(qOut)) +
  I(log(pLab) * log(qOut)) + I(log(pMat) * log(qOut))
Res.Df Df      F      Pr(>F)
```

```
1      135
2      125 10 4.3053 3.586e-05 ***
---
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( costCD, costTL )
```

Likelihood ratio test

```
Model 1: log(cost) ~ log(pCap) + log(pLab) + log(pMat) + log(qOut)
Model 2: log(cost) ~ log(pCap) + log(pLab) + log(pMat) + log(qOut) + I(0.5 *
  log(pCap)^2) + I(0.5 * log(pLab)^2) + I(0.5 * log(pMat)^2) +
  I(log(pCap) * log(pLab)) + I(log(pCap) * log(pMat)) + I(log(pLab) *
  log(pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap) * log(qOut)) +
  I(log(pLab) * log(qOut)) + I(log(pMat) * log(qOut))
#Df LogLik Df Chisq Pr(>Chisq)
```

```
1      6 -44.867
2     16 -24.149 10 41.435 9.448e-06 ***
---
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Given the very small P -values, we can conclude that the Cobb-Douglas cost function is not suitable for analyzing the production technology in our data set.

3.5.3 Linear homogeneity in input prices

Linear homogeneity of a Translog cost function requires

$$\ln(t c(w, y)) = \ln c(t w, y) \quad (3.87)$$

$$\ln t + \ln c(w, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln(t w_i) + \alpha_y \ln y \quad (3.88)$$

$$\begin{aligned} & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(t w_i) \ln(t w_j) + \frac{1}{2} \alpha_{yy} (\ln y)^2 \\ & + \sum_{i=1}^N \alpha_{iy} \ln(t w_i) \ln y \\ & = \alpha_0 + \sum_{i=1}^N \alpha_i \ln(t) + \sum_{i=1}^N \alpha_i \ln(w_i) + \alpha_y \ln y \end{aligned} \quad (3.89)$$

$$\begin{aligned} & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(t) \ln(t) + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(t) \ln(w_j) \\ & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(w_i) \ln(t) + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(w_i) \ln(w_j) \\ & + \frac{1}{2} \alpha_{yy} (\ln y)^2 + \sum_{i=1}^N \alpha_{iy} \ln(t) \ln y + \sum_{i=1}^N \alpha_{iy} \ln(w_i) \ln y \\ & = \alpha_0 + \ln(t) \sum_{i=1}^N \alpha_i + \sum_{i=1}^N \alpha_i \ln(w_i) + \alpha_y \ln y \end{aligned} \quad (3.90)$$

$$\begin{aligned} & + \frac{1}{2} \ln(t) \ln(t) \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} + \frac{1}{2} \ln(t) \sum_{j=1}^N \ln(w_j) \sum_{i=1}^N \alpha_{ij} \\ & + \frac{1}{2} \ln(t) \sum_{i=1}^N \ln(w_i) \sum_{j=1}^N \alpha_{ij} + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(w_i) \ln(w_j) \\ & + \frac{1}{2} \alpha_{yy} (\ln y)^2 + \ln(t) \ln y \sum_{i=1}^N \alpha_{iy} + \sum_{i=1}^N \alpha_{iy} \ln(w_i) \ln y \\ & = \ln c(w, y) + \ln(t) \sum_{i=1}^N \alpha_i \end{aligned} \quad (3.91)$$

$$\begin{aligned} & + \frac{1}{2} \ln(t) \ln(t) \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} + \frac{1}{2} \ln(t) \sum_{j=1}^N \ln(w_j) \sum_{i=1}^N \alpha_{ij} \\ & + \frac{1}{2} \ln(t) \sum_{i=1}^N \ln(w_i) \sum_{j=1}^N \alpha_{ij} + \ln(t) \ln y \sum_{i=1}^N \alpha_{iy} \\ \ln t & = \ln(t) \sum_{i=1}^N \alpha_i + \frac{1}{2} \ln(t) \ln(t) \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} + \frac{1}{2} \ln(t) \sum_{j=1}^N \ln(w_j) \sum_{i=1}^N \alpha_{ij} \end{aligned} \quad (3.92)$$

$$\begin{aligned} & + \frac{1}{2} \ln(t) \sum_{i=1}^N \ln(w_i) \sum_{j=1}^N \alpha_{ij} + \ln(t) \ln y \sum_{i=1}^N \alpha_{iy} \\ 1 & = \sum_{i=1}^N \alpha_i + \frac{1}{2} \ln(t) \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} + \frac{1}{2} \sum_{j=1}^N \ln(w_j) \sum_{i=1}^N \alpha_{ij} \end{aligned} \quad (3.93)$$

$$+ \frac{1}{2} \sum_{i=1}^N \ln(w_i) \sum_{j=1}^N \alpha_{ij} + \ln y \sum_{i=1}^N \alpha_{iy}$$

Hence, the homogeneity condition is only globally fulfilled (i.e. no matter which values t , w , and y have) if the following parameter restrictions hold:

$$\sum_{i=1}^N \alpha_i = 1 \quad (3.94)$$

$$\sum_{i=1}^N \alpha_{ij} = 0 \quad \forall j \xleftrightarrow{\alpha_{ij}=\alpha_{ji}} \sum_{j=1}^N \alpha_{ij} = 0 \quad \forall i \quad (3.95)$$

$$\sum_{i=1}^N \alpha_{iy} = 0 \quad (3.96)$$

We can see from the estimates above that these conditions are not fulfilled in our Translog cost function. For instance, according to condition (3.94), the first-order coefficients of the input prices should sum up to one but our estimates sum up to $0.199 + (-0.025) + (-1.245) = -1.071$. Hence, the homogeneity condition is not fulfilled in our estimated Translog cost function.

3.5.4 Estimation with linear homogeneity in input prices imposed

In order to impose linear homogeneity in input prices, we can rearrange these restrictions to get

$$\alpha_N = 1 - \sum_{i=1}^{N-1} \alpha_i \quad (3.97)$$

$$\alpha_{Nj} = - \sum_{i=1}^{N-1} \alpha_{ij} \quad \forall j \quad (3.98)$$

$$\alpha_{iN} = - \sum_{j=1}^{N-1} \alpha_{ij} \quad \forall i \quad (3.99)$$

$$\alpha_{Ny} = - \sum_{i=1}^{N-1} \alpha_{iy} \quad (3.100)$$

Replacing α_N , α_{Ny} and all α_{iN} and α_{jN} in equation (3.86) by the right-hand sides of equations (3.97) to (3.100) and re-arranging, we get

$$\begin{aligned} \ln \frac{c(w, y)}{w_N} &= \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln \frac{w_i}{w_N} + \alpha_y \ln y \\ &+ \frac{1}{2} \sum_{j=1}^{N-1} \sum_{i=1}^{N-1} \alpha_{ij} \ln \frac{w_i}{w_N} \ln \frac{w_j}{w_N} + \frac{1}{2} \alpha_{yy} (\ln y)^2 \\ &+ \sum_{i=1}^{N-1} \alpha_{iy} \ln \frac{w_i}{w_N} \ln y. \end{aligned} \quad (3.101)$$

3 Dual Approach: Cost Functions

This Translog cost function with linear homogeneity imposed can be estimated by following command:

```
> costTLHom <- lm( log( cost / pMat ) ~ log( pCap / pMat ) +
+   log( pLab / pMat ) + log( qOut ) +
+   I( 0.5 * log( pCap / pMat )^2 ) + I( 0.5 * log( pLab / pMat )^2 ) +
+   I( log( pCap / pMat ) * log( pLab / pMat ) ) +
+   I( 0.5 * log( qOut )^2 ) + I( log( pCap / pMat ) * log( qOut ) ) +
+   I( log( pLab / pMat ) * log( qOut ) ),
+   data = dat )
> summary( costTLHom )
```

Call:

```
lm(formula = log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) +
    log(qOut) + I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) +
    I(log(pCap/pMat) * log(pLab/pMat)) + I(0.5 * log(qOut)^2) +
    I(log(pCap/pMat) * log(qOut)) + I(log(pLab/pMat) * log(qOut)),
    data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.6860	-0.2086	0.0192	0.1978	0.8281

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	23.714976	3.445289	6.883	2.24e-10	***
log(pCap/pMat)	0.306159	0.525789	0.582	0.561383	
log(pLab/pMat)	1.093860	1.169160	0.936	0.351216	
log(qOut)	-1.933605	0.501090	-3.859	0.000179	***
I(0.5 * log(pCap/pMat)^2)	0.025951	0.089977	0.288	0.773486	
I(0.5 * log(pLab/pMat)^2)	0.716467	0.338049	2.119	0.035957	*
I(log(pCap/pMat) * log(pLab/pMat))	-0.292889	0.142710	-2.052	0.042144	*
I(0.5 * log(qOut)^2)	0.158662	0.039866	3.980	0.000114	***
I(log(pCap/pMat) * log(qOut))	-0.048274	0.040025	-1.206	0.229964	
I(log(pLab/pMat) * log(qOut))	0.008363	0.096490	0.087	0.931067	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3089 on 130 degrees of freedom

Multiple R-squared: 0.6377, Adjusted R-squared: 0.6126

F-statistic: 25.43 on 9 and 130 DF, p-value: < 2.2e-16

We can use a Wald test or a likelihood ratio test to compare this function with the unconstrained Translog cost function (3.86):

```
> linearHypothesis( costTL, c( "log(pCap) + log(pLab) + log(pMat) = 1",
+   "I(0.5 * log(pCap)^2) + I(log(pCap) * log(pLab)) + I(log(pCap) * log(pMat)) = 0",
+   "I(0.5 * log(pLab)^2) + I(log(pCap) * log(pLab)) + I(log(pLab) * log(pMat)) = 0",
+   "I(0.5 * log(pMat)^2) + I(log(pCap) * log(pMat)) + I(log(pLab) * log(pMat)) = 0",
+   "I(log(pCap) * log(qOut)) + I(log(pLab) * log(qOut)) + I(log(pMat) * log(qOut)) = 0" ) )
```

Linear hypothesis test:

```
log(pCap) + log(pLab) + log(pMat) = 1
I(0.5 * log(pCap)^2) + I(log(pCap) * log(pLab)) + I(log(pCap) * log(pMat)) = 0
I(0.5 * log(pLab)^2) + I(log(pCap) * log(pLab)) + I(log(pLab) * log(pMat)) = 0
I(0.5 * log(pMat)^2) + I(log(pCap) * log(pMat)) + I(log(pLab) * log(pMat)) = 0
I(log(pCap) * log(qOut)) + I(log(pLab) * log(qOut)) + I(log(pMat) * log(qOut)) = 0
```

Model 1: restricted model

```
Model 2: log(cost) ~ log(pCap) + log(pLab) + log(pMat) + log(qOut) + I(0.5 *
  log(pCap)^2) + I(0.5 * log(pLab)^2) + I(0.5 * log(pMat)^2) +
  I(log(pCap) * log(pLab)) + I(log(pCap) * log(pMat)) + I(log(pLab) *
  log(pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap) * log(qOut)) +
  I(log(pLab) * log(qOut)) + I(log(pMat) * log(qOut))
```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	130	12.407				
2	125	11.574	5	0.83308	1.7995	0.1177

```
> lrtest( costTL, costTLHom )
```

Likelihood ratio test

```
Model 1: log(cost) ~ log(pCap) + log(pLab) + log(pMat) + log(qOut) + I(0.5 *
  log(pCap)^2) + I(0.5 * log(pLab)^2) + I(0.5 * log(pMat)^2) +
  I(log(pCap) * log(pLab)) + I(log(pCap) * log(pMat)) + I(log(pLab) *
  log(pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap) * log(qOut)) +
  I(log(pLab) * log(qOut)) + I(log(pMat) * log(qOut))
```

```
Model 2: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
  I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
  log(pLab/pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap/pMat) *
  log(qOut)) + I(log(pLab/pMat) * log(qOut))
```

```
#Df LogLik Df Chisq Pr(>Chisq)
```

```
1 16 -24.149
2 11 -29.014 -5 9.7309 0.08323 .
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The Wald test does not reject the null hypothesis, i.e., linear homogeneity in input prices, while the LR test rejects the null hypothesis at the 10% significance level but not at the 5% level. Given the importance of microeconomic consistency and that 5% is the standard significance level, we continue our analysis with the Translog cost function with linear homogeneity in input prices imposed.

Furthermore, we can use a Wald test and a likelihood ratio test to compare this function with the Cobb-Douglas cost function with homogeneity imposed (3.28):

```
> waldtest( costCDHom, costTLHom )
```

Wald test

```
Model 1: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut)
Model 2: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
      I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
      log(pLab/pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap/pMat) *
      log(qOut)) + I(log(pLab/pMat) * log(qOut))
```

	Res.Df	Df	F	Pr(>F)
1	136			
2	130	6	5.5111	4.1e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( costCDHom, costTLHom )
```

Likelihood ratio test

```
Model 1: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut)
Model 2: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
      I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
      log(pLab/pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap/pMat) *
      log(qOut)) + I(log(pLab/pMat) * log(qOut))
```

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	5	-44.878			
2	11	-29.014	6	31.727	1.84e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Again, the Cobb-Douglas functional form is clearly rejected by the data in favor of the Translog functional form.

We can compute the total costs that are predicted by our estimated Translog cost function and compare them to the observed costs by following commands:

```
> dat$costTLHom <- exp( fitted( costTLHom ) ) * dat$pMat

> compPlot( dat$cost, dat$costTLHom )
> compPlot( dat$cost, dat$costTLHom, log = "xy" )
```

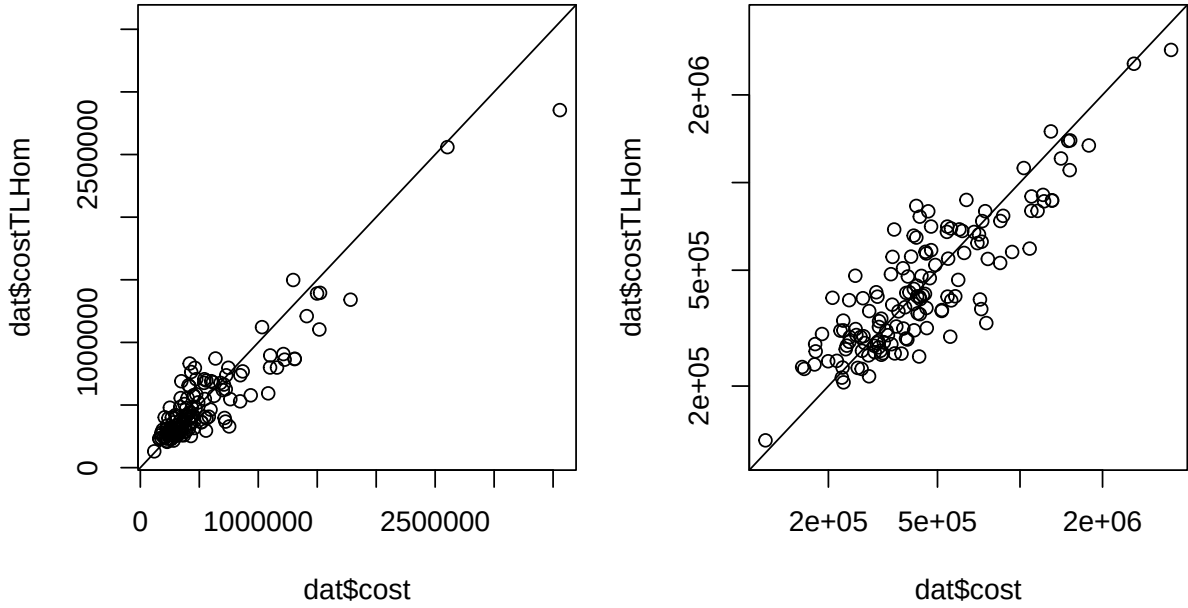


Figure 3.7: Translog cost function: observed vs. predicted costs

The resulting graphs are presented in figure 3.7.

Some parameters of the Translog cost function with linear homogeneity imposed (3.101) have not been directly estimated (α_N , α_{Ny} , all α_{iN} , all α_{jN}) but they can be retrieved from the (directly) estimated parameters and equations (3.97) to (3.100). Please note that the specification in equation (3.101) is used for the econometric estimation only; after retrieving the non-estimated parameters, we can do our analysis based on equation (3.86). To facilitate the further analysis, we create short-cuts of all estimated parameters and obtain the parameters that have not been directly estimated:

```
> ch0 <- coef( costTLHom )[ "(Intercept)" ]
> ch1 <- coef( costTLHom )[ "log(pCap/pMat)" ]
> ch2 <- coef( costTLHom )[ "log(pLab/pMat)" ]
> ch3 <- 1 - ch1 - ch2
> chy <- coef( costTLHom )[ "log(qOut)" ]
```

```
> ch11 <- coef( costTLHom )[ "I(0.5 * log(pCap/pMat)^2)" ]
> ch22 <- coef( costTLHom )[ "I(0.5 * log(pLab/pMat)^2)" ]
> chyy <- coef( costTLHom )[ "I(0.5 * log(qOut)^2)" ]
> ch12 <- ch21 <- coef( costTLHom )[ "I(log(pCap/pMat) * log(pLab/pMat))" ]
> ch13 <- ch31 <- 0 - ch11 - ch12
> ch23 <- ch32 <- 0 - ch12 - ch22
> ch33 <- 0 - ch13 - ch23
> ch1y <- coef( costTLHom )[ "I(log(pCap/pMat) * log(qOut))" ]
> ch2y <- coef( costTLHom )[ "I(log(pLab/pMat) * log(qOut))" ]
> ch3y <- 0 - ch1y - ch2y
```

Hence, our estimated Translog cost function has following parameters:

```
> # alpha_0, alpha_i, alpha_y
> unname( c( ch0, ch1, ch2, ch3, chy ) )

[1] 23.7149761 0.3061589 1.0938598 -0.4000187 -1.9336052

> # alpha_ij
> matrix( c( ch11, ch12, ch13, ch21, ch22, ch23, ch31, ch32, ch33 ), ncol=3 )

      [,1]      [,2]      [,3]
[1,] 0.02595083 -0.2928892 0.2669384
[2,] -0.29288920 0.7164670 -0.4235778
[3,] 0.26693837 -0.4235778 0.1566394

> # alpha_iy, alpha_yy
> unname( c( ch1y, ch2y, ch3y, chyy ) )

[1] -0.048274484 0.008362717 0.039911768 0.158661757
```

3.5.5 Cost flexibility and elasticity of size

Based on the estimated parameters, we can calculate the cost flexibilities and the elasticities of size. The cost flexibilities derived from a Translog cost function (3.86) are

$$\frac{\partial \ln c(w, y)}{\partial \ln y} = \alpha_y + \sum_{i=1}^N \alpha_{iy} \ln w_i + \alpha_{yy} \ln y \quad (3.102)$$

and the elasticities of size are—as always—their inverses:

$$\frac{\partial \ln y}{\partial \ln c} = \left(\frac{\partial \ln c(w, y)}{\partial \ln y} \right)^{-1} \quad (3.103)$$

We can calculate the cost flexibilities and the elasticities of size with following commands:

3 Dual Approach: Cost Functions

```
> dat$costFlex <- with( dat, chy + ch1y * log( pCap ) +
+   ch2y * log( pLab ) + ch3y * log( pMat ) + chyy * log( qOut ) )
> dat$elaSize <- 1 / dat$costFlex
```

Now, we can visualize these values using histograms:

```
> hist( dat$costFlex )
> hist( dat$elaSize )
> hist( dat$elaSize[ dat$elaSize > 0 & dat$elaSize < 10 ] )
```

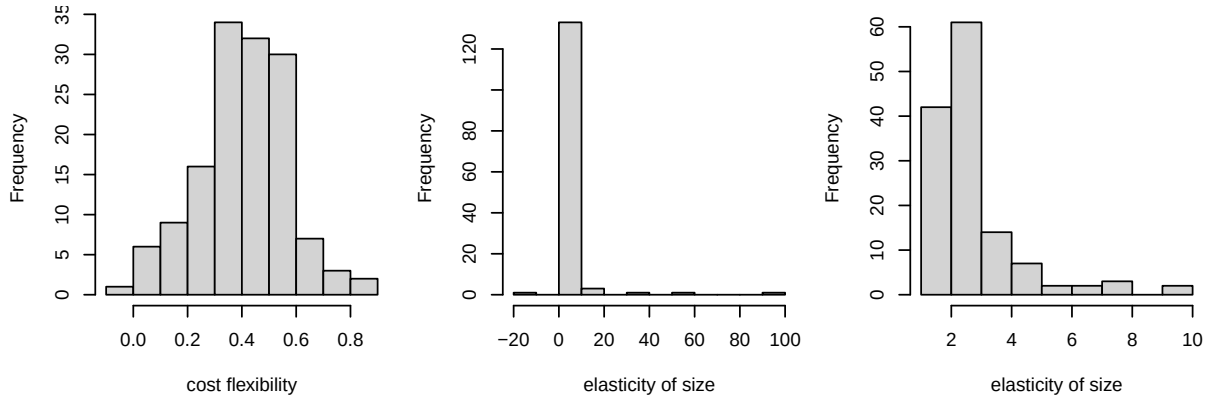


Figure 3.8: Translog cost function: cost flexibility and elasticity of size

The resulting graphs are presented in figure 3.8. Only 1 out of 140 cost flexibilities is negative. Hence, the estimated Translog cost function is to a very large extent increasing in the output quantity. All cost flexibilities are lower than one, which indicates that all apple producers operate under increasing returns to size. Most cost flexibilities are around 0.5, which corresponds to an elasticity of size of 2. Hence, if the apple producers increase their output quantity by one percent, the total costs of most producers increases by around 0.5 percent. Or—the other way round—if the apple producers increase their input use so that their costs increase by one percent, the output quantity of most producers would increase by around two percent.

With the following commands, we visualize the relationship between output quantity and elasticity of size

```
> plot( dat$qOut, dat$elaSize )
> abline( 1, 0 )
> plot( dat$qOut, dat$elaSize, ylim = c( 0, 10 ) )
> abline( 1, 0 )
> plot( dat$qOut, dat$elaSize, ylim = c( 0, 10 ), log = "x" )
> abline( 1, 0 )
```

The resulting graphs are shown in figure 3.9. With increasing output quantity, the elasticity of size approaches one (from above). Hence, small apple producers could gain a lot from increasing

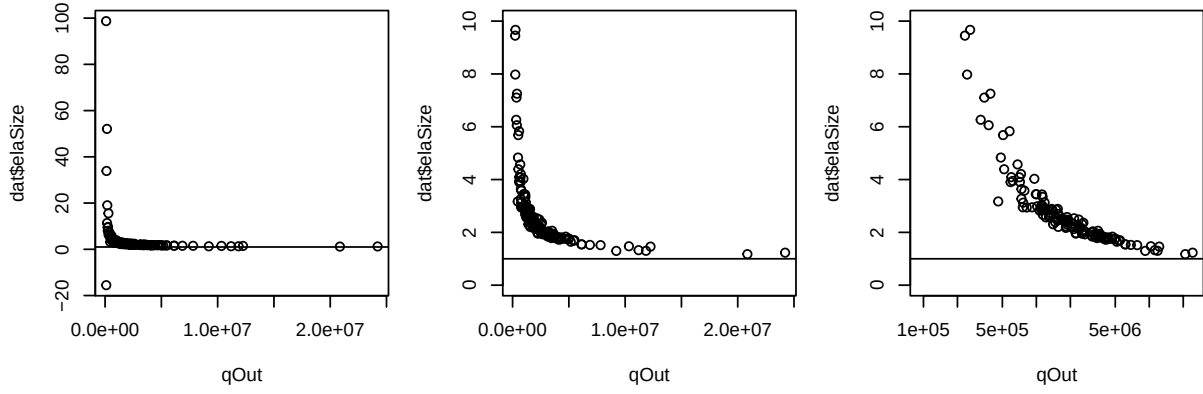


Figure 3.9: Translog cost function: output quantity and elasticity of size

their size, while large apple producers would gain much less from increasing their size. However, even the largest producers still gain from increasing their size so that the optimal firm size is larger than the largest firm in the sample.

3.5.6 Homotheticity

In order to check whether the Translog cost function as specified in equation (3.86) is homothetic, we check whether we can decompose it as done in equation (3.5). This decomposition of the Translog cost function is possible if $\alpha_{iy} = 0 \forall i$ so that it can be decomposed in the following way:

$$c_0(c_1, c_2) = \exp(\alpha_0 + c_1(w) + c_2(y)) \quad (3.104)$$

$$c_1(w) = \sum_{i=1}^N \alpha_i \ln w_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln w_i \ln w_j \quad (3.105)$$

$$c_2(y) = \alpha_y \ln y + \frac{1}{2} \alpha_{yy} (\ln y)^2. \quad (3.106)$$

Hence, we can conclude that the Translog cost function is homothetic if $\alpha_{iy} = 0 \forall i$.

The following command estimates a Translog cost function with both linear homogeneity in input prices and homotheticity imposed:

```
> costTLHomHom <- lm( log( cost / pMat ) ~ log( pCap / pMat ) +
+   log( pLab / pMat ) + log( qOut ) +
+   I( 0.5 * log( pCap / pMat )^2 ) + I( 0.5 * log( pLab / pMat )^2 ) +
+   I( log( pCap / pMat ) * log( pLab / pMat ) ) +
+   I( 0.5 * log( qOut )^2 ),
+   data = dat )
> summary( costTLHomHom )
```

Call:

3 Dual Approach: Cost Functions

```
lm(formula = log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) +
    log(qOut) + I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) +
    I(log(pCap/pMat) * log(pLab/pMat)) + I(0.5 * log(qOut)^2),
    data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.69257	-0.20604	0.00191	0.20321	0.82527

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	23.13551	3.40391	6.797	3.34e-10 ***
log(pCap/pMat)	-0.28784	0.20635	-1.395	0.16539
log(pLab/pMat)	1.22386	0.44113	2.774	0.00633 **
log(qOut)	-1.91286	0.47646	-4.015	9.93e-05 ***
I(0.5 * log(pCap/pMat)^2)	0.01609	0.08951	0.180	0.85761
I(0.5 * log(pLab/pMat)^2)	0.67114	0.28180	2.382	0.01866 *
I(log(pCap/pMat) * log(pLab/pMat))	-0.23524	0.13214	-1.780	0.07734 .
I(0.5 * log(qOut)^2)	0.16235	0.03352	4.844	3.51e-06 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3085 on 132 degrees of freedom

Multiple R-squared: 0.6331, Adjusted R-squared: 0.6137

F-statistic: 32.54 on 7 and 132 DF, p-value: < 2.2e-16

The following two commands apply a Wald test and a likelihood-ratio test, respectively, to test the homotheticity property:

```
> waldtest( costTLHom, costTLHomHom )
```

Wald test

```
Model 1: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
    I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
    log(pLab/pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap/pMat) *
    log(qOut)) + I(log(pLab/pMat) * log(qOut))
```

```
Model 2: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
    I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
    log(pLab/pMat)) + I(0.5 * log(qOut)^2)
```

Res.Df	Df	F	Pr(>F)
--------	----	---	--------

```
1      130
2      132 -2 0.8259 0.4401
```

```
> lrtest( costTLHom, costTLHomHom )
```

Likelihood ratio test

```
Model 1: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
      I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
      log(pLab/pMat)) + I(0.5 * log(qOut)^2) + I(log(pCap/pMat) *
      log(qOut)) + I(log(pLab/pMat) * log(qOut))
Model 2: log(cost/pMat) ~ log(pCap/pMat) + log(pLab/pMat) + log(qOut) +
      I(0.5 * log(pCap/pMat)^2) + I(0.5 * log(pLab/pMat)^2) + I(log(pCap/pMat) *
      log(pLab/pMat)) + I(0.5 * log(qOut)^2)
#Df  LogLik Df  Chisq Pr(>Chisq)
1  11 -29.014
2   9 -29.898 -2  1.7676    0.4132
```

None of the two tests can reject homotheticity of the estimated Translog cost function.

3.5.7 Marginal costs and average costs

Marginal costs derived from a Translog cost function are

$$\frac{\partial c(w, y)}{\partial y} = \frac{\partial \ln c(w, y)}{\partial \ln y} \frac{c(w, y)}{y} = \left(\alpha_y + \sum_{i=1}^N \alpha_{iy} \ln w_i + \alpha_{yy} \ln y \right) \frac{c(w, y)}{y}. \quad (3.107)$$

Hence, they are—as always—equal to the cost flexibility multiplied by total costs and divided by the output quantity. Hence, we can compute the marginal costs by:

```
> dat$margCostTL <- with( dat, costFlex * costTLHom / qOut )
```

We can visualize these marginal costs with a histogram.

```
> hist( dat$margCostTL, 15 )
```

The resulting graph is shown in figure 3.10. It indicates that producing one additional output unit increases the costs of most firms by around 0.09 monetary units.

Furthermore, we can check if the marginal costs are equal to the output prices, which is a first-order condition for profit maximization:

```
> compPlot( dat$pOut, dat$margCostTL )
> compPlot( dat$pOut[ dat$margCostTL > 0 ],
+   dat$margCostTL[ dat$margCostTL > 0 ], log = "xy" )
```

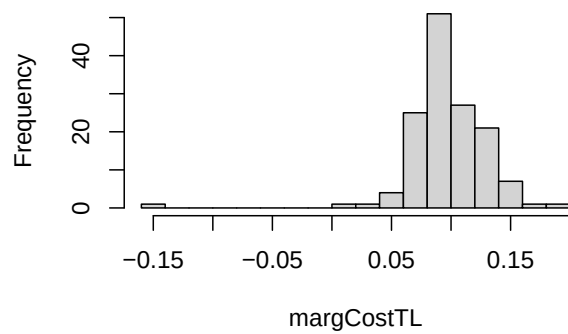


Figure 3.10: Translog cost function: Marginal costs

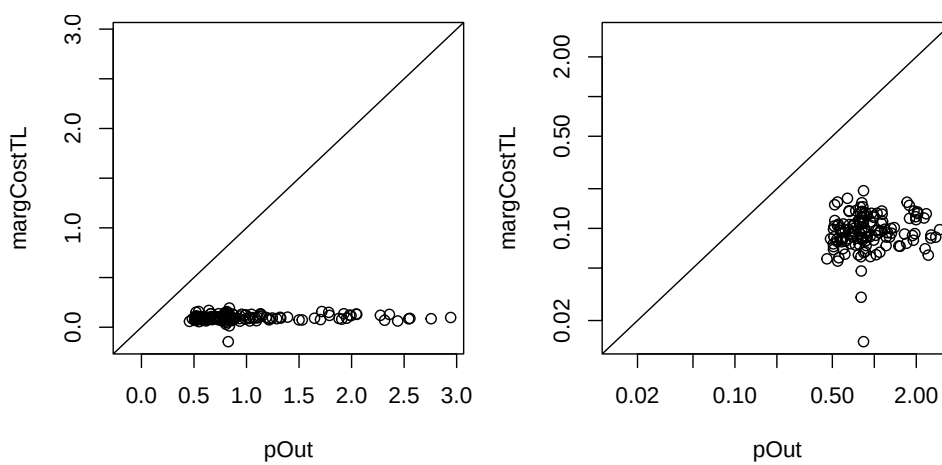


Figure 3.11: Translog cost function: marginal costs and output prices

The resulting graphs are shown in figure 3.11. The marginal costs of all firms are considerably smaller than their output prices. Hence, all firms would gain from increasing their output level. This is not surprising for a technology with large economies of scale.

Now, we analyze, how the marginal costs depend on the output quantity:

```
> plot( dat$qOut, dat$margCostTL,
+       ylim = range( c( dat$margCostTL, dat$costTLHom / dat$qOut ) ) )
> points( dat$qOut, dat$costTLHom / dat$qOut, pch = 4 )
> legend( "topright", pch = c( 21, 4 ),
+       legend = c( "marginal costs", "average costs" ) )
> plot( dat$qOut, dat$margCostTL, log = "x",
+       ylim = range( c( dat$margCostTL, dat$costTLHom / dat$qOut ) ) )
> points( dat$qOut, dat$costTLHom / dat$qOut, pch = 4 )
> legend( "topright", pch = c( 21, 4 ),
+       legend = c( "marginal costs", "average costs" ) )
```

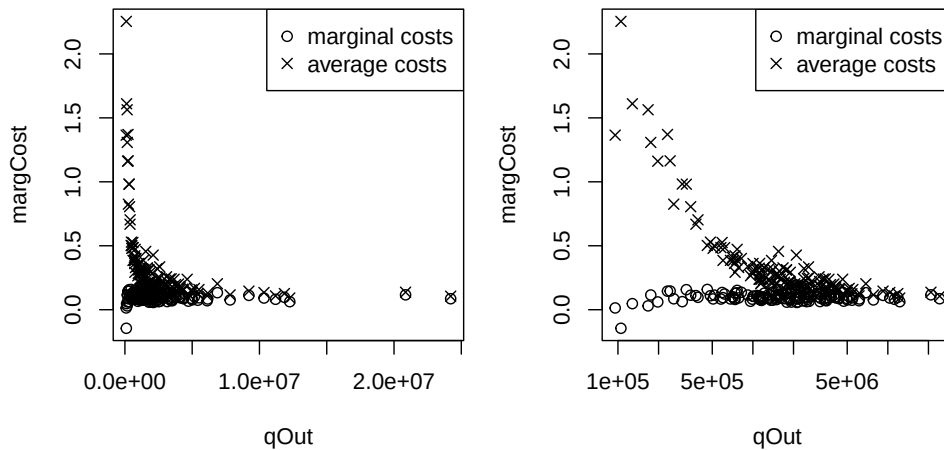


Figure 3.12: Translog cost function: Marginal costs depending on output quantity

The resulting graphs are shown in figure 3.12. There is no clear relationship between marginal costs and the output quantity.

Now, we illustrate our estimated model by drawing the average cost curve and the marginal cost curve for output quantities between 0 and five times the maximum output level in the sample, where we use the sample means of the input prices.

```
> y <- seq( 0, 5 * max( dat$qOut ), length.out = 200 )
> lpCap <- log( mean( dat$pCap ) )
> lpLab <- log( mean( dat$pLab ) )
> lpMat <- log( mean( dat$pMat ) )
> totalCost <- exp( ch0 + ch1 * lpCap + ch2 * lpLab + ch3 * lpMat +
+   chy * log( y ) + 0.5 * chy * log( y )^2 +
```

```

+ 0.5 * ch11 * lpCap^2 + 0.5 * ch22 * lpLab^2 + 0.5 * ch33 * lpMat^2 +
+ ch12 * lpCap * lpLab + ch13 * lpCap * lpMat + ch23 * lpLab * lpMat +
+ ch1y * lpCap * log( y ) + ch2y * lpLab * log( y ) + ch3y * lpMat * log( y ) )
> margCost <- ( chy + chy * log( y ) +
+ ch1y * lpCap + ch2y * lpLab + ch3y * lpMat ) * totalCost / y

> # average costs
> plot( y, totalCost/y, type = "l" )

> # marginal costs
> lines( y, margCost, lty = 2 )
> # minimum and maximum output level in the sample
> abline( v = min( dat$qOut ) )
> abline( v = max( dat$qOut ) )
> legend( "topright", lty = c( 1, 2 ),
+ legend = c( "average costs", "marginal costs" ) )

> # average costs
> plot( y, totalCost/y, type = "l", ylim = c( 0.07, 0.10 ) )

> # marginal costs
> lines( y, margCost, lty = 2 )
> # maximum output level in the sample
> abline( v = max( dat$qOut ) )
> legend( "topright", lty = c( 1, 2 ),
+ legend = c( "average costs", "marginal costs" ) )

```

The resulting graphs are shown in figure 3.13. The average costs are decreasing until an output level of around 70,000,000 units (1 unit $\approx$ 1 Euro) and they are increasing for larger output quantities. The average cost curve intersects the marginal cost curve (of course) at its minimum. However, as the maximum output level in the sample (approx. 25,000,000 units) is considerably lower than the minimum of the average cost curve (approx. 70,000,000 units), the estimated minimum of the average cost curve cannot be reliably determined because there are no data in this region.

3.5.8 Derived input demand functions

We can derive the cost-minimizing input quantities from the Translog cost function using Shepard's lemma:

$$x_i(w, y) = \frac{\partial c(w, y)}{\partial w_i} \quad (3.108)$$

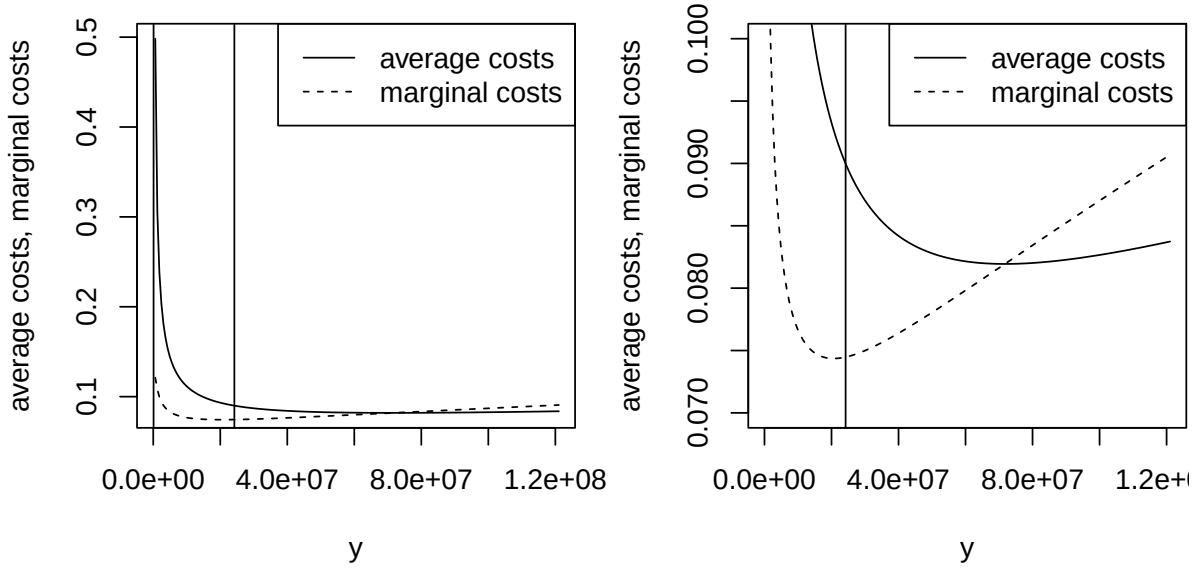


Figure 3.13: Translog cost function: marginal and average costs

$$= \frac{\partial \ln c(w, y)}{\partial \ln w_i} \frac{c}{w_i} \quad (3.109)$$

$$= \left(\alpha_i + \sum_{j=1}^N \alpha_{ij} \ln w_j + \alpha_{iy} \ln y \right) \frac{c}{w_i} \quad (3.110)$$

And we can re-arrange these derived input demand functions in order to obtain the cost-minimizing cost shares:

$$s_i(w, y) \equiv \frac{w_i x_i(w, y)}{c} = \alpha_i + \sum_{j=1}^N \alpha_{ij} \ln w_j + \alpha_{iy} \ln y \quad (3.111)$$

We can calculate the cost-minimizing cost shares based on our estimated Translog cost function by following commands:

```
> dat$shCap <- with( dat, ch1 + ch11 * log( pCap ) +
+   ch12 * log( pLab ) + ch13 * log( pMat ) + ch1y * log( qOut ) )
> dat$shLab <- with( dat, ch2 + ch21 * log( pCap ) +
+   ch22 * log( pLab ) + ch23 * log( pMat ) + ch2y * log( qOut ) )
> dat$shMat <- with( dat, ch3 + ch31 * log( pCap ) +
+   ch32 * log( pLab ) + ch33 * log( pMat ) + ch3y * log( qOut ) )
```

We visualize the cost-minimizing cost shares with histograms:

```
> hist( dat$shCap )
> hist( dat$shLab )
> hist( dat$shMat )
```

The resulting graphs are shown in figure 3.14. As the signs of the derived optimal cost shares are equal to the signs of the first derivatives of the cost function with respect to the input prices, we

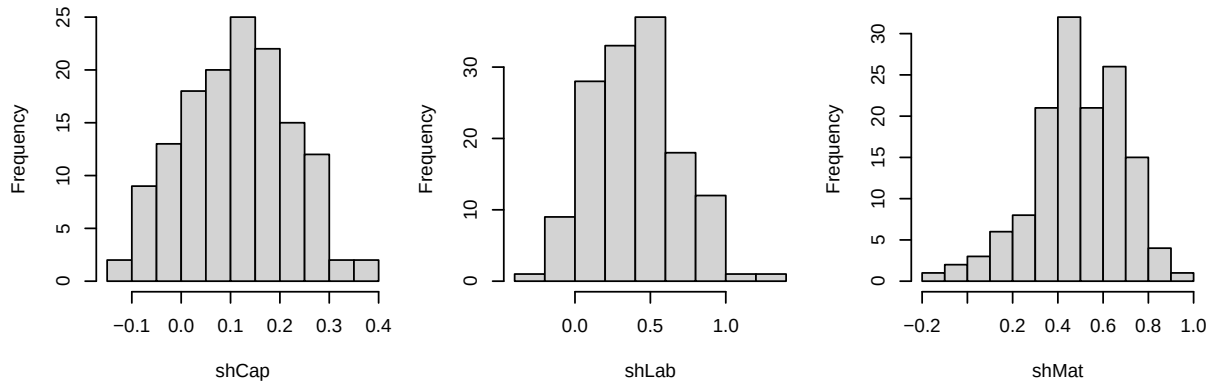


Figure 3.14: Translog cost function: cost-minimizing cost shares

can check whether the cost function is non-decreasing in input prices by checking if the derived optimal cost shares are non-negative. Counting the negative derived optimal cost shares, we find that our estimated cost function is decreasing in the capital price at 24 observations, decreasing in the labor price at 10 observations, and decreasing in the materials price at 3 observations. Given that our data set has 140 observations, our estimated cost function is to a large extent non-decreasing in input prices.

As our estimated cost function is (forced to be) linearly homogeneous in all input prices, the derived optimal cost shares always sum up to one:

```
> range( with( dat, shCap + shLab + shMat ) )
[1] 1 1
```

We can use the following commands to compare the observed cost shares with the derived cost-minimizing cost shares:

```
> compPlot( dat$shCap, dat$vCap / dat$cost )
> compPlot( dat$shLab, dat$vLab / dat$cost )
> compPlot( dat$shMat, dat$vMat / dat$cost )
```

The resulting graphs are shown in figure 3.15. Most firms use less than optimal materials, while there is a tendency to use more than optimal capital and a very slight tendency to use more than optimal labor.

Similarly, we can compare the observed input quantities with the cost-minimizing input quantities:

```
> compPlot( dat$shCap * dat$costTLHom / dat$pCap,
+   dat$vCap / dat$pCap )
> compPlot( dat$shLab * dat$costTLHom / dat$pLab,
+   dat$vLab / dat$pLab )
```

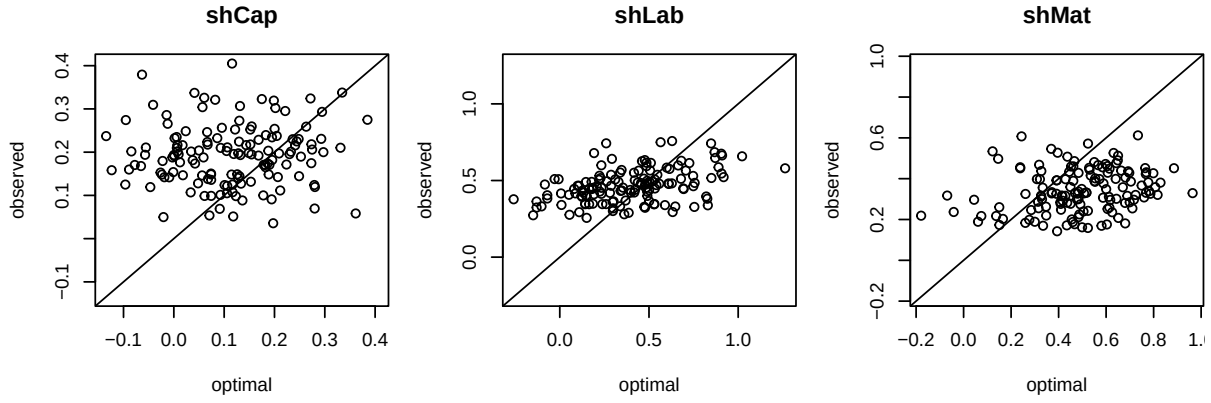


Figure 3.15: Translog cost function: observed and cost-minimizing cost shares

```
> compPlot( dat$shMat * dat$costTLHom / dat$pMat,
+   dat$vmMat / dat$pMat )
```

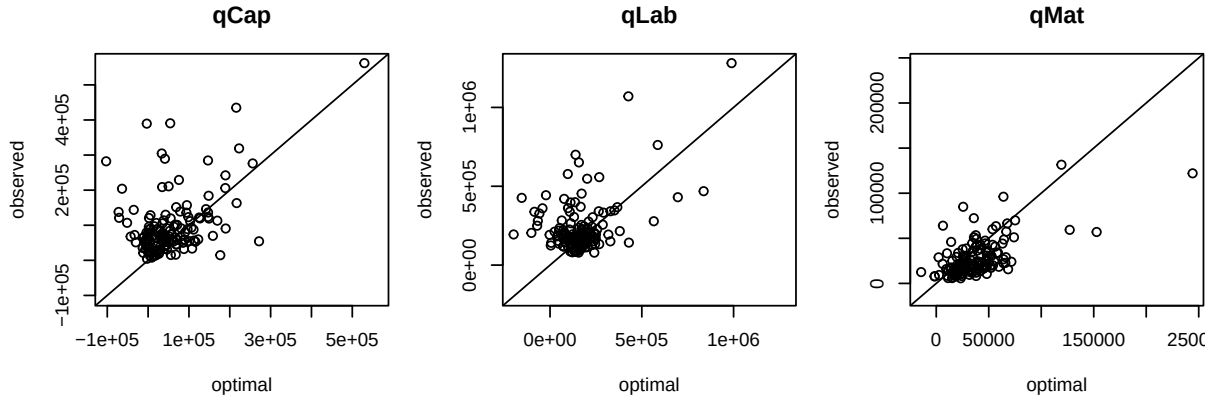


Figure 3.16: Translog cost function: observed and cost-minimizing input quantities

The resulting graphs are shown in figure 3.16. Of course, the conclusions derived from these graphs are the same as conclusions derived from figure 3.15.

3.5.9 Derived input demand elasticities

Based on the derived input demand functions (3.110), we can derive the input demand elasticities with respect to input prices:

$$\epsilon_{ij}(w, y) = \frac{\partial x_i(w, y)}{\partial w_j} \frac{w_j}{x_i(w, y)} \quad (3.112)$$

$$= \frac{\partial \left(\left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{c}{w_i} \right)}{\partial w_j} \frac{w_j}{x_i} \quad (3.113)$$

3 Dual Approach: Cost Functions

$$= \left[\frac{\alpha_{ij}}{w_j} \frac{c}{w_i} + \left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{x_j}{w_i} - \delta_{ij} \left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{c}{w_i^2} \right] \frac{w_j}{x_i} \quad (3.114)$$

$$= \left[\frac{\alpha_{ij}}{w_i} \frac{c}{w_j} + \frac{x_i w_i}{c} \frac{x_j}{w_i} - \delta_{ij} \frac{x_i w_i}{c} \frac{c}{w_i^2} \right] \frac{w_j}{x_i} \quad (3.115)$$

$$= \frac{\alpha_{ij}}{w_i} \frac{c}{x_i} + \frac{w_j x_j}{c} - \delta_{ij} \frac{w_j}{w_i} \quad (3.116)$$

$$= \frac{\alpha_{ij}}{s_i} + s_j - \delta_{ij}, \quad (3.117)$$

where δ_{ij} (again) denotes Kronecker's delta (2.95), and the input demand elasticities with respect to the output quantity:

$$\epsilon_{iy}(w, y) = \frac{\partial x_i(w, y)}{\partial y} \frac{y}{x_i(w, y)} \quad (3.118)$$

$$= \frac{\partial \left(\left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{c}{w_i} \right)}{\partial y} \frac{y}{x_i} \quad (3.119)$$

$$= \left[\frac{\alpha_{iy}}{y} \frac{c}{w_i} + \left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{\partial c}{\partial y} \frac{1}{w_i} \right] \frac{y}{x_i} \quad (3.120)$$

$$= \left[\frac{\alpha_{iy}}{w_i} \frac{c}{y} + \frac{w_i x_i}{c} \frac{\partial c}{\partial y} \frac{1}{w_i} \right] \frac{y}{x_i} \quad (3.121)$$

$$= \frac{\alpha_{iy}}{w_i} \frac{c}{x_i} + \frac{\partial c}{\partial y} \frac{y}{c} \quad (3.122)$$

$$= \frac{\alpha_{iy}}{s_i} + \frac{\partial \ln c}{\partial \ln y}, \quad (3.123)$$

where $\partial \ln c / \partial \ln y$ is the cost flexibility (see section 3.1.3).

With the following commands, we compute the input demand elasticities at the first observation:

```
> ela <- matrix( NA, nrow = 3, ncol = 4 )
> rownames( ela ) <- c( "qCap", "qLab", "qMat" )
> colnames( ela ) <- c( "wCap", "wLab", "wMat", "y" )
> ela[ 1, 1 ] <- ch11 / dat$shCap[1] + dat$shCap[1] - 1
> ela[ 1, 2 ] <- ch12 / dat$shCap[1] + dat$shLab[1]
> ela[ 1, 3 ] <- ch13 / dat$shCap[1] + dat$shMat[1]
> ela[ 1, 4 ] <- ch1y / dat$shCap[1] + dat$costFlex[1]
> ela[ 2, 1 ] <- ch21 / dat$shLab[1] + dat$shCap[1]
> ela[ 2, 2 ] <- ch22 / dat$shLab[1] + dat$shLab[1] - 1
> ela[ 2, 3 ] <- ch23 / dat$shLab[1] + dat$shMat[1]
> ela[ 2, 4 ] <- ch2y / dat$shLab[1] + dat$costFlex[1]
```

```
> ela[ 3, 1 ] <- ch31 / dat$shMat[1] + dat$shCap[1]
> ela[ 3, 2 ] <- ch32 / dat$shMat[1] + dat$shLab[1]
> ela[ 3, 3 ] <- ch33 / dat$shMat[1] + dat$shMat[1] - 1
> ela[ 3, 4 ] <- ch3y / dat$shMat[1] + dat$costFlex[1]
> ela
```

	wCap	wLab	wMat	y
qCap	-0.6383107	-1.1835104	1.821821136	0.1653394
qLab	4.4484591	-11.3084023	6.859943173	0.2293691
qMat	0.5938258	-0.5948896	0.001063746	0.3983336

These demand elasticities indicate that when the capital price increases by one percent, the demand for capital decreases by 0.638 percent, the demand for labor increases by 4.448 percent, and the demand for materials increases by 0.594 percent. When the labor price increases by one percent, the elasticities indicate that the demand for all inputs decreases, which is not possible when the output quantity should be maintained. Furthermore, the symmetry condition for the elasticities (3.69) indicates that the cross-price elasticities of each input pair must have the same sign. However, this is not the case for the pairs capital–labor and materials–labor. The reason for this is the negative predicted input share of labor:

```
> dat[ 1, c( "shCap", "shLab", "shMat" ) ]
```

	shCap	shLab	shMat
1	0.2630271	-0.06997825	0.8069511

Finally, the negativity constraint (3.67) is violated, because the own-price elasticity of materials is positive (0.001).

When the output quantity is increased by one percent, the demand for capital increases by 0.165 percent, the demand for labor increases by 0.229 percent, and the demand for materials increases by 0.398 percent.

Now, we create a three-dimensional array and compute the demand elasticities for all observations:

```
> elaAll <- array( NA, c( 3, 4, nrow( dat ) ) )
> dimnames( elaAll ) <- list( c( "qCap", "qLab", "qMat" ),
+   c( "wCap", "wLab", "wMat", "y" ), 1:nrow(dat) )
> elaAll[ 1, 1, ] <- ch11 / dat$shCap + dat$shCap - 1
> elaAll[ 1, 2, ] <- ch12 / dat$shCap + dat$shLab
> elaAll[ 1, 3, ] <- ch13 / dat$shCap + dat$shMat
> elaAll[ 1, 4, ] <- ch1y / dat$shCap + dat$costFlex
> elaAll[ 2, 1, ] <- ch21 / dat$shLab + dat$shCap
> elaAll[ 2, 2, ] <- ch22 / dat$shLab + dat$shLab - 1
```

```
> elaAll[ 2, 3, ] <- ch23 / dat$shLab + dat$shMat
> elaAll[ 2, 4, ] <- ch2y / dat$shLab + dat$costFlex
> elaAll[ 3, 1, ] <- ch31 / dat$shMat + dat$shCap
> elaAll[ 3, 2, ] <- ch32 / dat$shMat + dat$shLab
> elaAll[ 3, 3, ] <- ch33 / dat$shMat + dat$shMat - 1
> elaAll[ 3, 4, ] <- ch3y / dat$shMat + dat$costFlex
```

The following commands present the median values of the elasticities over all observations, at which the monotonicity conditions are fulfilled; this makes sure that the optimal input shares are always positive.

```
> dat$monoCostTLHom <- with( dat, shCap >= 0 & shLab >= 0 & shMat >= 0 &
+      costFlex >= 0 )
> apply( elaAll[ , , dat$monoCostTLHom ], c(1,2), median )
```

	wCap	wLab	wMat	y
qCap	-0.6486401	-2.0250830	2.6857495	-0.007520786
qLab	-0.7529717	1.4426349	-0.6992232	0.467598439
qMat	0.6641220	-0.5086388	-0.1645389	0.506421497

We can visualize the variations of the elasticities using histograms, where we—again—only include observations, at which the monotonicity conditions are fulfilled.

```
> hist( elaAll[1,1,dat$monoCostTLHom] )
> hist( elaAll[1,2,dat$monoCostTLHom] )
> hist( elaAll[1,3,dat$monoCostTLHom] )
> hist( elaAll[2,1,dat$monoCostTLHom] )
> hist( elaAll[2,2,dat$monoCostTLHom] )
> hist( elaAll[2,3,dat$monoCostTLHom] )
> hist( elaAll[3,1,dat$monoCostTLHom] )
> hist( elaAll[3,2,dat$monoCostTLHom] )
> hist( elaAll[3,3,dat$monoCostTLHom] )
> hist( elaAll[1,4,dat$monoCostTLHom] )
> hist( elaAll[2,4,dat$monoCostTLHom] )
> hist( elaAll[3,4,dat$monoCostTLHom] )
```

The resulting graphs are shown in figure 3.17. While the conditional own-price elasticities of capital and materials are negative at almost all observations, the conditional own-price elasticity of labor is positive at almost all observations. These violations of the negativity constraint (3.67) originate from the violation of the concavity condition. As all conditional elasticities of the capital demand with respect to the materials price as well as all conditional elasticities of the materials demand with respect to the capital price are positive, we can conclude that capital and materials

3 Dual Approach: Cost Functions

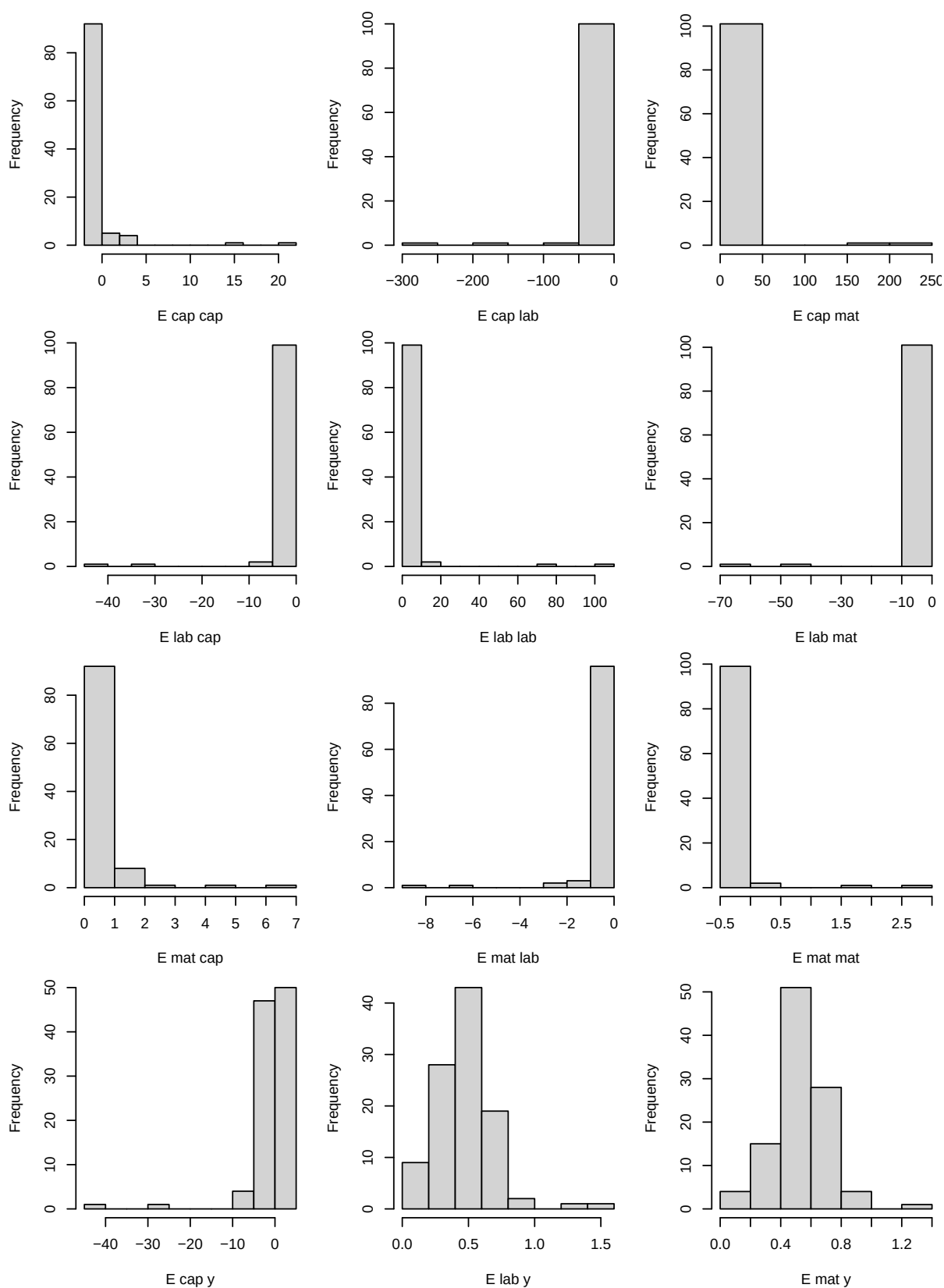


Figure 3.17: Translog cost function: derived demand elasticities

are net substitutes. In contrast, all cross-price elasticities between capital and labor as well as between labor and materials are negative. This indicates that the two pairs capital and labor as well as labor and materials are net complements.

When the output quantity is increased by one percent, most farms would increase both the labor quantity and the materials quantity by around 0.5% and either increase or decrease the capital quantity.

3.5.10 Theoretical consistency

The Translog cost function (3.86) is always continuous for positive input prices and a positive output quantity.

The non-negativity is always fulfilled for the Translog cost function, because the predicted cost is equal to the exponential function of the right-hand side of equation (3.86) and the exponential function always returns a non-negative value (also when the right-hand side of equation (3.86) is negative).

If the output quantity approaches zero (from above), the right-hand side of the Translog cost functions (equation 3.86) approaches:

$$\lim_{y \rightarrow 0+} \left(\alpha_0 + \sum_{i=1}^N \alpha_i \ln w_i + \alpha_y \ln y + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln w_i \ln w_j + \frac{1}{2} \alpha_{yy} (\ln y)^2 + \sum_{i=1}^N \alpha_{iy} \ln w_i \ln y \right) \quad (3.124)$$

$$= \lim_{y \rightarrow 0+} \left(\alpha_y \ln y + \frac{1}{2} \alpha_{yy} (\ln y)^2 + \sum_{i=1}^N \alpha_{iy} \ln w_i \ln y \right) \quad (3.125)$$

$$= \lim_{y \rightarrow 0+} \left(\left(\alpha_y + \frac{1}{2} \alpha_{yy} \ln y + \sum_{i=1}^N \alpha_{iy} \ln w_i \right) \ln y \right) \quad (3.126)$$

$$= \lim_{y \rightarrow 0+} \left(\alpha_y + \frac{1}{2} \alpha_{yy} \ln y + \sum_{i=1}^N \alpha_{iy} \ln w_i \right) \lim_{y \rightarrow 0+} \ln y \quad (3.127)$$

$$= \lim_{y \rightarrow 0+} (\alpha_{yy} \ln y) \lim_{y \rightarrow 0+} \ln y = (-\alpha_{yy} \infty)(-\infty) = \alpha_{yy} \infty \quad (3.128)$$

Hence, if coefficient α_{yy} is negative and the output quantity approaches zero (from above), the predicted cost (exponential function of the right-hand side of equation 3.86) approaches zero so that the “no fixed costs” property is asymptotically fulfilled.

Our estimated Translog cost function with linear homogeneity in input prices imposed (of course) is linearly homogeneous in input prices. Hence, the linear homogeneity property is globally fulfilled.

A cost function is non-decreasing in the output quantity if the cost flexibility and the elasticity of size are non-negative. As we can see from figure 3.8, only a single cost flexibility and thus, only a single elasticity of size is negative. Hence, our estimated Translog cost function with linear homogeneity in input prices imposed violates the monotonicity condition regarding the output

quantity only at a single observation.

Given Shepard's lemma, a cost function is non-decreasing in input prices if the derived cost-minimizing input quantities and the corresponding cost shares are non-negative. As we can see from figure 3.14, our estimated Translog cost function with linear homogeneity in input prices imposed predicts that 24 cost shares of capital, 10 cost shares of labor, and 3 cost shares of materials are negative. In total, the monotonicity condition regarding the input prices is violated at 36 observations:

```
> table( dat$shCap >= 0 & dat$shLab >= 0 & dat$shMat >= 0 )
```

```
FALSE  TRUE
    36   104
```

Concavity in input prices of the cost function requires that the Hessian matrix of the cost function with respect to the input prices is negative semidefinite. The elements of the Hessian matrix are:

$$H_{ij} = \frac{\partial^2 c(w, y)}{\partial w_i \partial w_j} = \frac{\partial x_i(w, y)}{\partial w_j} \quad (3.129)$$

$$= \frac{\partial \left(\left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{c}{w_i} \right)}{\partial w_j} \quad (3.130)$$

$$= \frac{\alpha_{ij}}{w_j} \frac{c}{w_i} + \left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{x_j}{w_i} - \delta_{ij} \left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln w_k + \alpha_{iy} \ln y \right) \frac{c}{w_i^2} \quad (3.131)$$

$$= \frac{\alpha_{ij}}{w_i w_j} + \frac{x_i w_i}{c} \frac{x_j}{w_i} - \delta_{ij} \frac{x_i w_i}{c} \frac{c}{w_i^2} \quad (3.132)$$

$$= \frac{\alpha_{ij}}{w_i w_j} + \frac{x_i x_j}{c} - \delta_{ij} \frac{x_i}{w_i}, \quad (3.133)$$

where δ_{ij} (again) denotes Kronecker's delta (2.95). As the elements of the Hessian matrix have the same sign as the corresponding elasticities ($H_{ij} = \epsilon_{ij}(w, y) x_i/w_j$), the positive own-price elasticities of labor in figure 3.17 indicate that the element $H_{lab,lab}$ is positive at all observations, where the monotonicity conditions regarding the input prices are fulfilled. As negative semidefiniteness requires that all diagonal elements of the (Hessian) matrix are negative, we can conclude that the estimated Translog cost function is concave at not a single observation where the monotonicity conditions regarding the input prices are fulfilled.

This means that our estimated Translog cost function is inconsistent with microeconomic theory at all observations.

3.6 Translog cost function with multiple outputs

3.6.1 Specification

The Translog cost function with multiple outputs is a minor extension of the Translog cost function with a single output as defined in equation (3.86):

$$\begin{aligned} \ln c(w, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln w_i + \sum_{i=1}^M \beta_i \ln y_i \\ & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln w_i \ln w_j + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln y_i \ln y_j \\ & + \sum_{i=1}^N \sum_{j=1}^M \gamma_{ij} \ln w_i \ln y_j \end{aligned} \quad (3.134)$$

with $\alpha_{ij} = \alpha_{ji} \forall i, j = 1, \dots, N$ and $\beta_{ij} = \beta_{ji} \forall i, j = 1, \dots, M$.

3.6.2 Properties

The Translog multiple-output cost function cannot be globally quasi-convex in output quantities because it cannot have globally concave iso-cost curves for different output-output combinations. In fact, both the Cobb-Douglas and the Translog multiple-output cost functions predict that when decreasing one output quantity towards zero, the other output quantities can be increased towards infinity while still holding total costs constant.

4 Dual Approach: Profit Function

4.1 Theory

4.1.1 Profit functions

The profit function:

$$\pi(p, w) = \max_{y, x} p y - \sum_i w_i x_i, \text{ s.t. } y = f(x) \quad (4.1)$$

returns the maximum profit that is attainable given the output price p and input prices w .

It is important to distinguish the profit definition (1.4) from the profit function (4.1).

4.1.2 Short-run profit functions

As producers often cannot instantly adjust the quantity of the some inputs (e.g. capital, land, apple trees), estimating a short-run profit function with some quasi-fixed input quantities might be more appropriate than a (long-run) profit function which assumes that all input quantities and output quantities can be adjusted instantly. Furthermore, a short-run profit function can model technologies with increasing returns to scale, if the sum over the output elasticities of the *variable* inputs is lower than one.

In general, a short-run profit function is defined as

$$\pi^v(p, w^1, x^2) = \max_{y \geq 0} \{ p y - c^s(w^1, y, x^2) \}, \quad (4.2)$$

where w^1 denotes the vector of the prices of all variable inputs, x^2 denotes the vector of the quantities of all quasi-fixed inputs, $c^s(w^1, y, x^2)$ is the short-run cost function (see section 3.3), π^v denotes the gross margin defined in equation (1.5), and N^1 is a vector of the indices of the variable inputs.

4.2 Graphical illustration of profit and gross margin

We use the following commands to visualize the variation of the profits and the relationship between profits and firm size:

```
> hist( dat$profit, 30 )  
> plot( dat$X, dat$profit, log = "xy" )
```

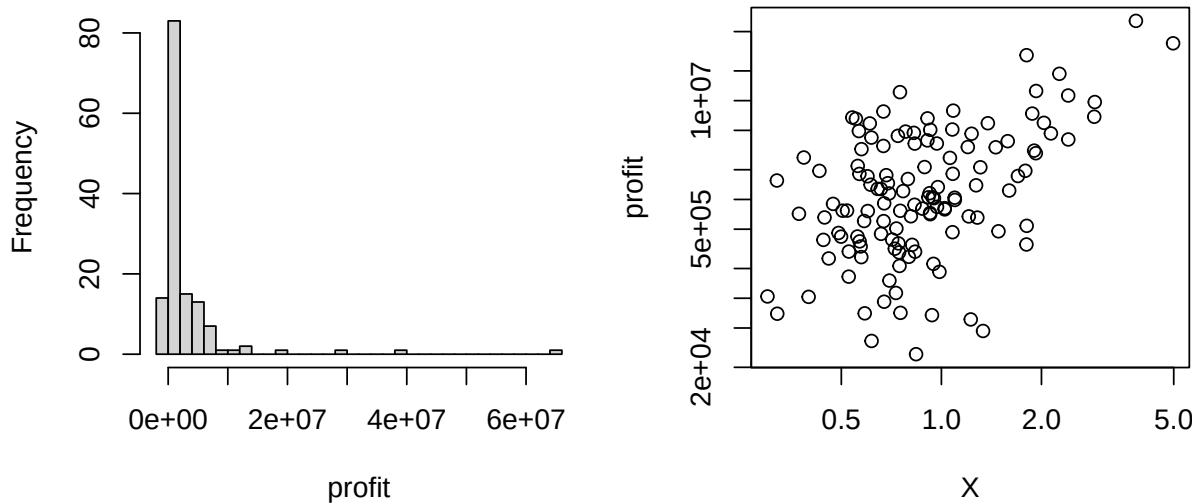



Figure 4.1: Profit

The resulting graphs are shown in figure 4.1. The histogram shows that 14 out of 140 apple producers (10%) have (slightly) negative profits. Although this seems to be not unrealistic, this contradicts the non-negativity condition of the profit function. However, the observed negative profits might have been caused by deviations from the theoretical assumptions that we have made to derive the profit function, e.g. that all inputs can be instantly adjusted and that there are no unexpected events such as severe weather conditions or pests. We will deal with these deviations from our assumptions later and for now just ignore the observations with negative profits in our analyses with the profit function. The right part of figure 4.1 shows that the profit clearly increases with firm size.

The following commands graphically illustrate the variation of the gross margins and their relationship to the firm size and the quantity of the quasi-fixed input:

```
> hist( dat$vProfit, 30 )
> plot( dat$X, dat$vProfit, log = "xy" )
> plot( dat$qCap, dat$vProfit, log = "xy" )
```

The resulting graphs are shown in figure 4.2. The histogram on the left shows that 8 out of 140 apple producers (6%) have (slightly) negative gross margins. Although this does not seem to be unrealistic, this contradicts the non-negativity condition of the short-run profit function. However, the observed negative gross margins might have been caused by deviations from the theoretical assumptions, e.g. that there are no unexpected events such as severe weather conditions or pests. The center part of figure 4.2 shows that the gross margin clearly increases with the firm size (as expected). However, the right part of this figure shows that the gross margin is only weakly positively correlated with the fixed input.

Please note that according to microeconomic theory, the short-run total profit π^s —in contrast

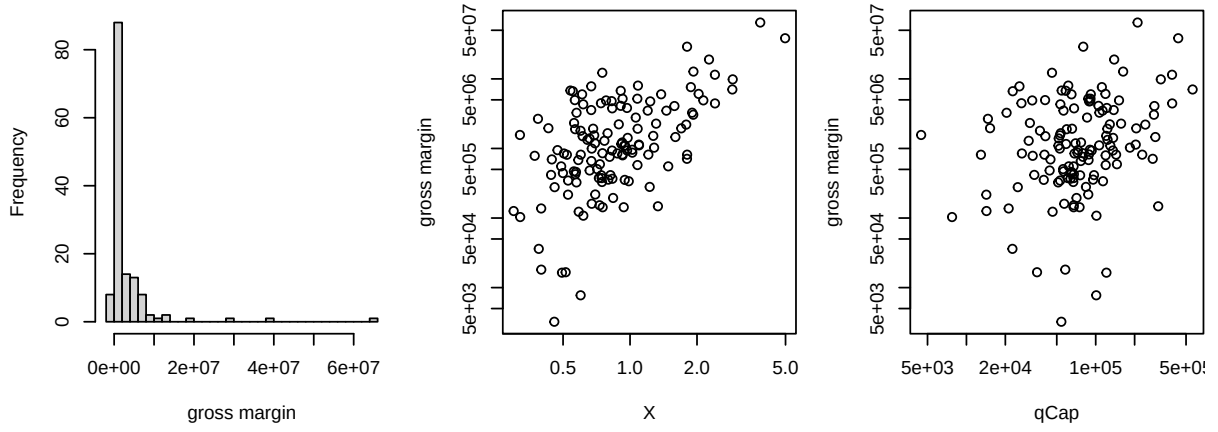


Figure 4.2: Gross margins

to the gross margin π^v —might be negative due to fixed costs:

$$\pi^s(p, w, x^2) = \pi^v(p, w^1, x^2) - \sum_{j \in N^2} w_j x_j, \quad (4.3)$$

where N^2 is a vector of the indices of the quasi-fixed inputs. However, in the long-run, profit must be non-negative:

$$\pi(p, w) = \max_{x^2} \pi^s(p, w, x^2) \geq 0, \quad (4.4)$$

as all costs are variable in the long run.

4.3 Cobb-Douglas profit function

4.3.1 Specification

The Cobb-Douglas profit function¹ has the following specification:

$$\pi = Ap^{\alpha_p} \left(\prod_i w_i^{\alpha_i} \right), \quad (4.5)$$

This function can be linearized to

$$\ln \pi = \alpha_0 + \alpha_p \ln p + \sum_i \alpha_i \ln w_i \quad (4.6)$$

with $\alpha_0 = \ln A$.

¹Please note that the Cobb-Douglas profit function is used as a simple example here but that it is much too restrictive for most “real” empirical applications (Chand and Kaul, 1986).

4.3.2 Estimation

The linearized Cobb-Douglas profit function can be estimated by OLS. As the logarithm of a negative number is not defined and function `lm` automatically removes observations with missing data, we do not have to remove the observations (apple producers) with negative profits manually.

```
> profitCD <- lm( log( profit ) ~ log( pOut ) + log( pCap ) + log( pLab ) +
+   log( pMat ), data = dat )
> summary( profitCD )
```

Call:

```
lm(formula = log(profit) ~ log(pOut) + log(pCap) + log(pLab) +
    log(pMat), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-3.6183	-0.2778	0.1261	0.5986	2.0442

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	13.9380	0.4921	28.321	< 2e-16 ***
log(pOut)	2.7117	0.2340	11.590	< 2e-16 ***
log(pCap)	-0.7298	0.1752	-4.165	5.86e-05 ***
log(pLab)	-0.1940	0.4623	-0.420	0.676
log(pMat)	0.1612	0.2543	0.634	0.527

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9815 on 121 degrees of freedom

(14 observations deleted due to missingness)

Multiple R-squared: 0.5911, Adjusted R-squared: 0.5776

F-statistic: 43.73 on 4 and 121 DF, p-value: < 2.2e-16

As expected, `lm` reports that 14 observations have been removed due to missing data (logarithms of negative numbers).

4.3.3 Properties

A Cobb-Douglas profit function is always continuous and twice continuously differentiable for all $p > 0$ and $w_i > 0 \forall i$. Furthermore, a Cobb-Douglas profit function automatically fulfills the non-negativity property, because the profit predicted by equation (4.5) is always positive as long as coefficient A is positive (given that all input prices and the output price are positive). As A

is usually obtained by applying the exponential function to the estimate of α_0 , i.e. $A = \exp(\alpha_0)$, A and hence, also the predicted profit, are always positive (even if α_0 is non-positive).

The estimated coefficients of the output price and the input prices indicate that profit is increasing in the output price and decreasing in the capital and labor price but it is increasing in the price of materials, which contradicts microeconomic theory. However, the positive coefficient of the (logarithmic) price of materials is statistically not significantly different from zero.

The Cobb-Douglas profit function is linearly homogeneous in all prices (output price and all input prices) if the following condition is fulfilled:

$$t \pi(p, w) = \pi(tp, tw) \quad (4.7)$$

$$\ln(t \pi) = \alpha_0 + \alpha_p \ln(t p) + \sum_i \alpha_i \ln(t w_i) \quad (4.8)$$

$$\ln t + \ln \pi = \alpha_0 + \alpha_p \ln t + \alpha_p \ln p + \sum_i \alpha_i \ln t + \sum_i \alpha_i \ln w_i \quad (4.9)$$

$$\ln t + \ln \pi = \alpha_0 + \alpha_p \ln p + \sum_i \alpha_i \ln w_i + \ln t \left(\alpha_p + \sum_i \alpha_i \right) \quad (4.10)$$

$$\ln \pi = \ln \pi + \ln t \left(\alpha_p + \sum_i \alpha_i - 1 \right) \quad (4.11)$$

$$0 = \ln t \left(\alpha_p + \sum_i \alpha_i - 1 \right) \quad (4.12)$$

$$0 = \alpha_p + \sum_i \alpha_i - 1 \quad (4.13)$$

$$1 = \alpha_p + \sum_i \alpha_i \quad (4.14)$$

Hence, the homogeneity condition is only fulfilled if the coefficient of the (logarithmic) output price and the coefficients of the (logarithmic) input prices sum up to one. As they sum up to $2.71 + (-0.73) + (-0.19) + 0.16 = 1.95$, the homogeneity condition is not fulfilled in our estimated model.

4.3.4 Estimation with linear homogeneity in all prices imposed

In order to derive a Cobb-Douglas profit function with linear homogeneity in input prices imposed, we re-arrange the homogeneity condition (4.14) to get

$$\alpha_p = 1 - \sum_{i=1}^N \alpha_i \quad (4.15)$$

and replace α_p in the profit function (4.6) by the right-hand side of the above equation:

$$\ln \pi = \alpha_0 + \left(1 - \sum_i \alpha_i \right) \ln p + \sum_i \alpha_i \ln w_i \quad (4.16)$$

4 Dual Approach: Profit Function

$$\ln \pi = \alpha_0 + \ln p - \sum_i \alpha_i \ln p + \sum_i \alpha_i \ln w_i \quad (4.17)$$

$$\ln \pi - \ln p = \alpha_0 + \sum_i \alpha_i (\ln w_i - \ln p) \quad (4.18)$$

$$\ln \frac{\pi}{p} = \alpha_0 + \sum_i \alpha_i \ln \frac{w_i}{p} \quad (4.19)$$

This Cobb-Douglas profit function with linear homogeneity imposed can be estimated by following command:

```
> profitCDHom <- lm( log( profit / pOut ) ~ log( pCap / pOut ) +
+   log( pLab / pOut ) + log( pMat / pOut ), data = dat )
> summary( profitCDHom )
```

Call:

```
lm(formula = log(profit/pOut) ~ log(pCap/pOut) + log(pLab/pOut) +
    log(pMat/pOut), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-3.6045	-0.2724	0.0972	0.6013	2.0385

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	14.27961	0.45962	31.068	< 2e-16 ***
log(pCap/pOut)	-0.82114	0.16953	-4.844	3.78e-06 ***
log(pLab/pOut)	-0.90068	0.25591	-3.519	0.000609 ***
log(pMat/pOut)	-0.02469	0.23530	-0.105	0.916610

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9909 on 122 degrees of freedom

(14 observations deleted due to missingness)

Multiple R-squared: 0.3568, Adjusted R-squared: 0.341

F-statistic: 22.56 on 3 and 122 DF, p-value: 1.091e-11

The coefficient of the (logarithmic) output price can be obtained by the homogeneity restriction (4.15). Hence, it is $1 - (-0.82) - (-0.9) - (-0.02) = 2.75$. Now, all monotonicity conditions are fulfilled: profit is increasing in the output price and decreasing in all input prices. We can use a Wald test or a likelihood-ratio test to test whether the model and the data contradict the homogeneity assumption:

```
> library( "car" )
> linearHypothesis( profitCD, "log(pOut) + log(pCap) + log(pLab) + log(pMat) = 1" )
```

Linear hypothesis test:

```
log(pOut) + log(pCap) + log(pLab) + log(pMat) = 1
```

Model 1: restricted model

Model 2: log(profit) ~ log(pOut) + log(pCap) + log(pLab) + log(pMat)

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	122	119.78				
2	121	116.57	1	3.2183	3.3407	0.07005 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( profitCD, profitCDHom )
```

Likelihood ratio test

Model 1: log(profit) ~ log(pOut) + log(pCap) + log(pLab) + log(pMat)

Model 2: log(profit/pOut) ~ log(pCap/pOut) + log(pLab/pOut) + log(pMat/pOut)

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	6	-173.88			
2	5	-175.60	-1	3.4316	0.06396 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Both tests reject the null hypothesis, linear homogeneity in all prices, at the 10% significance level but not at the 5% level. Given the importance of microeconomic consistency and that 5% is the standard significance level, we continue our analysis with the Cobb-Douglas profit function with linear homogeneity imposed.

4.3.5 Checking convexity in all prices

The last property that we have to check is the convexity in all prices. A continuous and twice continuously differentiable function is convex, if its Hessian matrix is positive semidefinite. A necessary and sufficient condition for positive semidefiniteness is that all its principal minors (not only its leading principal minors) are non-negative (see section 1.5.5). The first derivatives of the Cobb-Douglas profit function with respect to the input prices are:

$$\frac{\partial \pi}{\partial w_i} = \frac{\partial \ln \pi}{\partial \ln w_i} \frac{\pi}{w_i} = \alpha_i \frac{\pi}{w_i} \quad (4.20)$$

and the first derivative with respect to the output price is:

$$\frac{\partial \pi}{\partial p} = \frac{\partial \ln \pi}{\partial \ln p} \frac{\pi}{p} = \alpha_p \frac{\pi}{p} \quad (4.21)$$

Now, we can calculate the second derivatives as derivatives of the first derivatives (4.20) and (4.21):

$$\frac{\partial^2 \pi}{\partial w_i \partial w_j} = \frac{\partial \frac{\partial \pi}{\partial w_i}}{\partial w_j} = \frac{\partial \left(\alpha_i \frac{\pi}{w_i} \right)}{\partial w_j} \quad (4.22)$$

$$= \frac{\alpha_i}{w_i} \frac{\partial \pi}{\partial w_j} - \delta_{ij} \alpha_i \frac{\pi}{w_i^2} \quad (4.23)$$

$$= \frac{\alpha_i}{w_i} \alpha_j \frac{\pi}{w_j} - \delta_{ij} \alpha_i \frac{\pi}{w_i^2} \quad (4.24)$$

$$= \alpha_i (\alpha_j - \delta_{ij}) \frac{\pi}{w_i w_j} \quad (4.25)$$

$$\frac{\partial^2 \pi}{\partial w_i \partial p} = \frac{\partial \frac{\partial \pi}{\partial w_i}}{\partial p} = \frac{\partial \left(\alpha_i \frac{\pi}{w_i} \right)}{\partial p} \quad (4.26)$$

$$= \frac{\alpha_i}{w_i} \frac{\partial \pi}{\partial p} \quad (4.27)$$

$$= \frac{\alpha_i}{w_i} \alpha_p \frac{\pi}{p} \quad (4.28)$$

$$= \alpha_i \alpha_p \frac{\pi}{w_i p} \quad (4.29)$$

$$\frac{\partial^2 \pi}{\partial p^2} = \frac{\partial \frac{\partial \pi}{\partial p}}{\partial p} = \frac{\partial \left(\alpha_p \frac{\pi}{p} \right)}{\partial p} \quad (4.30)$$

$$= \frac{\alpha_p}{p} \frac{\partial \pi}{\partial p} - \alpha_p \frac{\pi}{p^2} \quad (4.31)$$

$$= \frac{\alpha_p}{p} \alpha_p \frac{\pi}{p} - \alpha_p \frac{\pi}{p^2} \quad (4.32)$$

$$= \alpha_p (\alpha_p - 1) \frac{\pi}{p^2}, \quad (4.33)$$

where δ_{ij} (again) denotes Kronecker's delta (2.95).

As *all* elements of the Hessian matrix include π as a multiplicative term, we can ignore this variable in the calculation of the Hessian matrix, because the value π neither changes the signs of the (diagonal) elements of the matrix, nor the signs of the principal minors and the determinant (as long as π is positive, i.e. the non-negativity condition is fulfilled) given the general rule that $|\pi \cdot M| = \pi \cdot |M|$, where M denotes a quadratic matrix, π denotes a scalar, and the two vertical bars denote the determinant function.

We start with checking convexity in all prices of the Cobb-Douglas profit function without homogeneity imposed.

To simplify the calculations, we define short-cuts for the coefficients:

```
> gCap <- coef( profitCD )[ "log(pCap)" ]
```

```
> gLab <- coef( profitCD )[ "log(pLab)" ]
> gMat <- coef( profitCD )[ "log(pMat)" ]
> gOut <- coef( profitCD )[ "log(pOut)" ]
```

Using these coefficients, we compute the second derivatives of our estimated Cobb-Douglas profit function:

```
> hpCapCap <- gCap * ( gCap - 1 ) / dat$pCap^2
> hpLabLab <- gLab * ( gLab - 1 ) / dat$pLab^2
> hpMatMat <- gMat * ( gMat - 1 ) / dat$pMat^2
> hpCapLab <- gCap * gLab / ( dat$pCap * dat$pLab )
> hpCapMat <- gCap * gMat / ( dat$pCap * dat$pMat )
> hpLabMat <- gLab * gMat / ( dat$pLab * dat$pMat )
> hpCapOut <- gCap * gOut / ( dat$pCap * dat$pOut )
> hpLabOut <- gLab * gOut / ( dat$pLab * dat$pOut )
> hpMatOut <- gMat * gOut / ( dat$pMat * dat$pOut )
> hpOutOut <- gOut * ( gOut - 1 ) / dat$pOut^2
```

Now, we prepare the Hessian matrix for the first observation:

```
> hessian <- matrix( NA, nrow = 4, ncol = 4 )
> hessian[ 1, 1 ] <- hpCapCap[1]
> hessian[ 2, 2 ] <- hpLabLab[1]
> hessian[ 3, 3 ] <- hpMatMat[1]
> hessian[ 1, 2 ] <- hessian[ 2, 1 ] <- hpCapLab[1]
> hessian[ 1, 3 ] <- hessian[ 3, 1 ] <- hpCapMat[1]
> hessian[ 2, 3 ] <- hessian[ 3, 2 ] <- hpLabMat[1]
> hessian[ 1, 4 ] <- hessian[ 4, 1 ] <- hpCapOut[1]
> hessian[ 2, 4 ] <- hessian[ 4, 2 ] <- hpLabOut[1]
> hessian[ 3, 4 ] <- hessian[ 4, 3 ] <- hpMatOut[1]
> hessian[ 4, 4 ] <- hpOutOut[1]
> print( hessian )
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.185633270	0.060331020	-0.005072286	-1.14920901
[2,]	0.060331020	0.286060178	-0.003907371	-0.88527867
[3,]	-0.005072286	-0.003907371	-0.001709437	0.07442915
[4,]	-1.149209014	-0.885278673	0.074429148	10.64451706

As the third element on the diagonal of this Hessian matrix is negative, the conditions for positive semidefiniteness regarding the first-order principal minors are not fulfilled. Hence, we do not need to calculate the higher-order principal minors of the Hessian matrix, as we already can conclude

that the Hessian matrix is not positive semidefinite and hence, the estimated profit function is not convex at the first observation.²

We can check whether the third element on the diagonal of the Hessian matrix is non-negative at other observations:

```
> table( hpMatMat >= 0 )
```

```
FALSE
```

```
140
```

As it is non-negative not at a single observation, we must conclude that the estimated Cobb-Douglas profit function without homogeneity imposed violates the convexity property at all observations.

Now, we will check, whether our Cobb-Douglas profit function with linear homogeneity imposed is convex in all prices. Again, we create short-cuts for the estimated coefficients:

```
> ghCap <- coef( profitCDHom )["log(pCap/pOut)"]
> ghLab <- coef( profitCDHom )["log(pLab/pOut)"]
> ghMat <- coef( profitCDHom )["log(pMat/pOut)"]
> ghOut <- 1- ghCap - ghLab - ghMat
```

We compute the second derivatives:

```
> hphCapCap <- ghCap * ( ghCap - 1 ) / dat$pCap^2
> hphLabLab <- ghLab * ( ghLab - 1 ) / dat$pLab^2
> hphMatMat <- ghMat * ( ghMat - 1 ) / dat$pMat^2
> hphCapLab <- ghCap * ghLab / ( dat$pCap * dat$pLab )
> hphCapMat <- ghCap * ghMat / ( dat$pCap * dat$pMat )
> hphLabMat <- ghLab * ghMat / ( dat$pLab * dat$pMat )
> hphCapOut <- ghCap * ghOut / ( dat$pCap * dat$pOut )
> hphLabOut <- ghLab * ghOut / ( dat$pLab * dat$pOut )
> hphMatOut <- ghMat * ghOut / ( dat$pMat * dat$pOut )
> hphOutOut <- ghOut * ( ghOut - 1 ) / dat$pOut^2
```

And we prepare the Hessian matrix for the first observation:

```
> hessianHom <- matrix( NA, nrow = 4, ncol = 4 )
> hessianHom[ 1, 1 ] <- hphCapCap[1]
> hessianHom[ 2, 2 ] <- hphLabLab[1]
> hessianHom[ 3, 3 ] <- hphMatMat[1]
```

²Please note that this Hessian matrix is not negative semidefinite either, because the other two first-order principal minors are positive, while negative semidefiniteness requires that all first-order principal minors are non-positive. Hence, the Cobb-Douglas profit function is neither concave nor convex at the first observation.

```
> hessianHom[ 1, 2 ] <- hessianHom[ 2, 1 ] <- hphCapLab[1]
> hessianHom[ 1, 3 ] <- hessianHom[ 3, 1 ] <- hphCapMat[1]
> hessianHom[ 2, 3 ] <- hessianHom[ 3, 2 ] <- hphLabMat[1]
> hessianHom[ 1, 4 ] <- hessianHom[ 4, 1 ] <- hphCapOut[1]
> hessianHom[ 2, 4 ] <- hessianHom[ 4, 2 ] <- hphLabOut[1]
> hessianHom[ 3, 4 ] <- hessianHom[ 4, 3 ] <- hphMatOut[1]
> hessianHom[ 4, 4 ] <- hphOutOut[1]
> print( hessianHom )
```

```
          [,1]          [,2]          [,3]          [,4]
[1,] 0.2198994186 0.315188197 0.0008740851 -1.30964735
[2,] 0.3151881974 2.114366248 0.0027786041 -4.16320062
[3,] 0.0008740851 0.002778604 0.0003198275 -0.01154546
[4,] -1.3096473550 -4.163200623 -0.0115454561 11.00022565
```

As all diagonal elements of this Hessian matrix are non-negative, the conditions regarding the first-order principal minors are all fulfilled. In the following, we calculate the six second-order principal minors, the four third-order principal minors, and the determinant (fourth-order principal minor):

```
> det( hessianHom[ c(1:2), c(1:2)] )
```

```
[1] 0.3656043
```

```
> det( hessianHom[ c(1:3), c(1:3)] )
```

```
[1] 0.0001151481
```

```
> det( hessianHom[ c(1:4), c(1:4)] )
```

```
[1] -5.245992e-20
```

```
> det( hessianHom[ c(2:3), c(2:3)] )
```

```
[1] 0.0006685118
```

```
> det( hessianHom[ c(2:4), c(2:4)] )
```

```
[1] 0.001795727
```

```
> det( hessianHom[ c(3:4), c(3:4)] )
```

```
[1] 0.003384877
```

```
> det( hessianHom[-4,-4] )
```

```
[1] 0.0001151481

> det( hessianHom[-3,-3] )

[1] 0.02088665

> det( hessianHom[-2,-2] )

[1] 0.0002138007

> det( hessianHom[-1,-1] )

[1] 0.001795727

> det( hessianHom )

[1] -5.245992e-20
```

These principal minors are all either positive or very close to zero. For instance, the determinant (fourth-order principal minor) of the Hessian is close to zero, where it is negative on some computers but positive on other computers. As Hessian matrices of linear homogeneous functions are always singular, it is expected that the determinant of the Hessian matrix (the N th-order principal minor) is zero. However, the computed determinant of our Hessian matrix is not exactly zero due to rounding errors, which are unavoidable on digital computers. Rounding errors may also have caused that our calculations of other principle minors resulted in values that are negative but very close to zero, although they are in fact zero or slightly positive. Assuming that the principal minors that are very close to zero are in fact zero or positive, we can conclude that our Cobb-Douglas profit function with linear homogeneity imposed is convex in all prices at the first observation. We can check the semidefiniteness of a matrix more conveniently with the command `semidefiniteness` (package *miscTools*), which (by default) checks the signs of the principal minors and returns a logical value indicating whether the necessary and sufficient conditions for negative or positive semidefiniteness are fulfilled:

We can check the semidefiniteness of a matrix more conveniently with the command `semidefiniteness` (package *miscTools*), which (by default) checks the signs of the principal minors and returns a logical value indicating whether the necessary and sufficient conditions for negative or positive semidefiniteness are fulfilled:

```
> semidefiniteness( hessianHom, positive = TRUE )

[1] TRUE
```

In the following, we will check whether convexity in all prices is fulfilled at each observation in the sample:

```

> dat$convexCDHom <- NA
> for( obs in 1:nrow( dat ) ) {
+   hessianLoop <- matrix( NA, nrow = 4, ncol = 4 )
+   hessianLoop[ 1, 1 ] <- hphCapCap[obs]
+   hessianLoop[ 2, 2 ] <- hphLabLab[obs]
+   hessianLoop[ 3, 3 ] <- hphMatMat[obs]
+   hessianLoop[ 1, 2 ] <- hessianLoop[ 2, 1 ] <- hphCapLab[obs]
+   hessianLoop[ 1, 3 ] <- hessianLoop[ 3, 1 ] <- hphCapMat[obs]
+   hessianLoop[ 2, 3 ] <- hessianLoop[ 3, 2 ] <- hphLabMat[obs]
+   hessianLoop[ 1, 4 ] <- hessianLoop[ 4, 1 ] <- hphCapOut[obs]
+   hessianLoop[ 2, 4 ] <- hessianLoop[ 4, 2 ] <- hphLabOut[obs]
+   hessianLoop[ 3, 4 ] <- hessianLoop[ 4, 3 ] <- hphMatOut[obs]
+   hessianLoop[ 4, 4 ] <- hphOutOut[obs]
+   dat$convexCDHom[obs] <-
+     semidefiniteness( hessianLoop, positive = TRUE, tol = 1e-10 )
+ }
> table( dat$convexCDHom )

```

TRUE

140

This result indicates that the convexity condition is violated not at a single observation. Consequently, our Cobb-Douglas profit function with linear homogeneity imposed is convex in all prices at all observations.

4.3.6 Predicted profit

As the dependent variable of the Cobb-Douglas profit function without homogeneity imposed is $\ln(\pi)$, we have to apply the exponential function to the fitted dependent variable, in order obtain the fitted profit π . Furthermore, we have to be aware of that the `fitted` method only returns the predicted values for the observations that were included in the estimation. Hence, we have to make sure that the predicted profits are only assigned to the observations that have a positive profit and hence, were included in the estimation:

```

> dat$profitCD[ dat$profit > 0 ] <- exp( fitted( profitCD ) )

```

We obtain the predicted profit from the Cobb-Douglas profit function with homogeneity imposed by:

```

> dat$profitCDHom[ dat$profit > 0 ] <- exp( fitted( profitCDHom ) ) * dat$pOut

```

4.3.7 Optimal profit shares

Given Hotelling's Lemma, the coefficients of the (logarithmic) output price and the (logarithmic) input prices are equal to the optimal "profit shares" derived from a Cobb-Douglas profit function:

$$\alpha_p = \frac{\partial \ln \pi(p, w)}{\partial \ln p} = \frac{\partial \pi(p, w)}{\partial p} \frac{p}{\pi(p, w)} = y(p, w) \frac{p}{\pi(p, w)} = \frac{p y(p, w)}{\pi(p, w)} \equiv r(p, w) \geq 1 \quad (4.34)$$

$$\alpha_i = \frac{\partial \ln \pi(p, w)}{\partial \ln w_i} = \frac{\partial \pi(p, w)}{\partial w_i} \frac{w_i}{\pi(p, w)} = -x_i(p, w) \frac{w_i}{\pi(p, w)} = -\frac{w_i x_i(p, w)}{\pi(p, w)} \equiv r_i(p, w) \leq 0 \quad (4.35)$$

In contrast to "real" shares, these "profit shares" are never between zero and one but they sum up to one, as do "real" shares:

$$r + \sum_i r_i = \frac{p y}{\pi} + \sum_i \left(-\frac{w_i x_i}{\pi} \right) = \frac{p y - \sum_i w_i x_i}{\pi} = \frac{\pi}{\pi} = 1 \quad (4.36)$$

For instance, an optimal profit share of the output of $\alpha_p = 2.75$ means that profit maximization would result in a total revenue that is 2.75 times as large as the profit, which corresponds to a return on sales of $1/2.75 = 36\%$. Similarly, an optimal profit share of the capital input of $\alpha_{cap} = -0.82$ means that profit maximization would result in total capital costs that are 0.82 times as large as the profit.

The following commands draw histograms of the observed profit shares and compare them to the optimal profit shares, which are predicted by our Cobb-Douglas profit function with linear homogeneity imposed:

```
> hist( ( dat$pOut * dat$qOut / dat$profit ) [
+   dat$profit > 0 ], 30 )
> abline( v = ghOut, lwd = 3 )
> hist( ( - dat$pCap * dat$qCap / dat$profit ) [
+   dat$profit > 0 ], 30 )
> abline( v = ghCap, lwd = 3 )
> hist( ( - dat$pLab * dat$qLab / dat$profit ) [
+   dat$profit > 0 ], 30 )
> abline( v = ghLab, lwd = 3 )
> hist( ( - dat$pMat * dat$qMat / dat$profit ) [
+   dat$profit > 0 ], 30 )
> abline( v = ghMat, lwd = 3 )
```

The resulting graphs are shown in figure 4.3. These results somewhat contradict previous results. While the results based on production functions and cost functions indicate that the apple producers on average use too much capital and too few materials, the results of the Cobb-Douglas profit function indicate that almost all apple producers use too much materials and most apple producers use too less capital and labor. However, the results of the Cobb-Douglas profit function

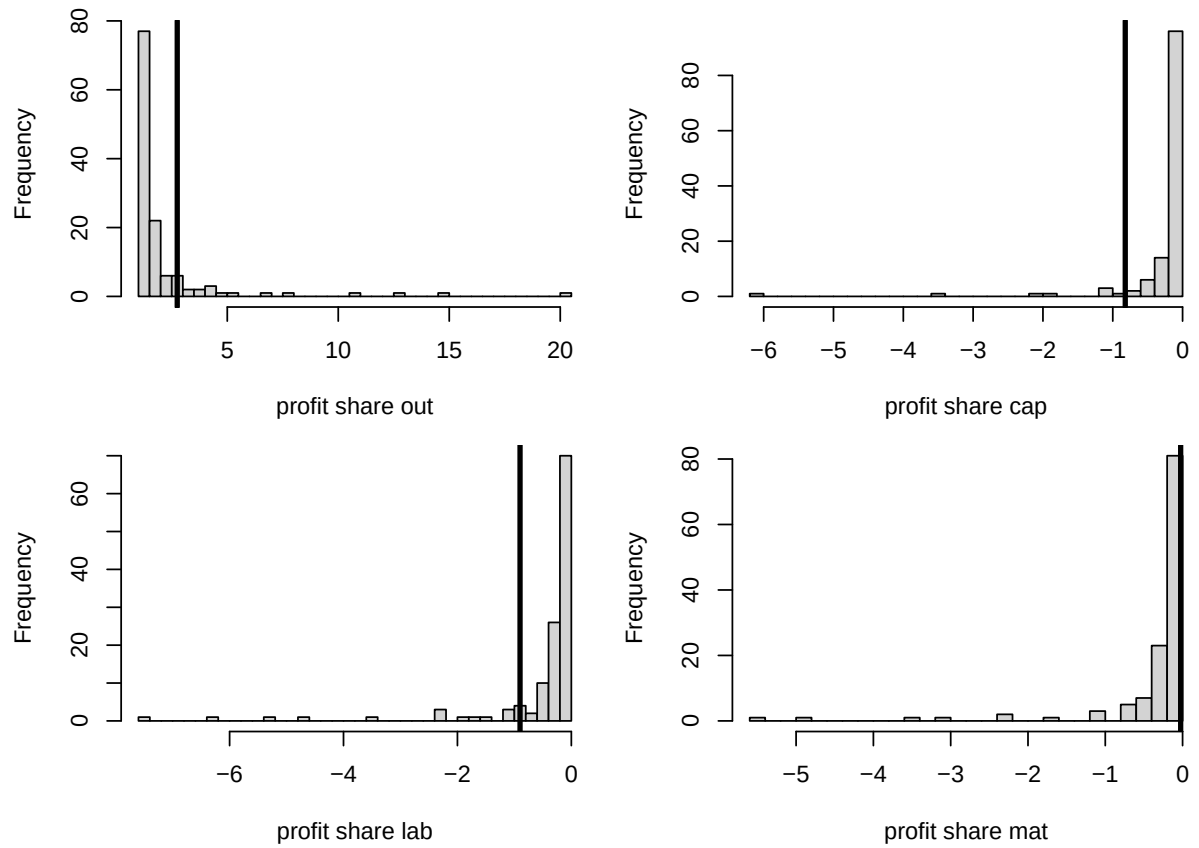


Figure 4.3: Cobb-Douglas profit function: observed and optimal profit shares

are consistent with previous results regarding the output quantity: all results suggest that most apple producers should produce more output.

4.3.8 Derived output supply and input demand functions

Hotelling's Lemma says that the partial derivative of a profit function with respect to the output price is the output supply function and that the partial derivatives of a profit function with respect to the input prices are the negative (unconditional) input demand functions:

$$y(p, w) = \frac{\partial \pi(p, w)}{\partial p} = \alpha_p \frac{\pi(p, w)}{p} \quad (4.37)$$

$$x_i(p, w) = -\frac{\partial \pi(p, w)}{\partial w_i} = -\alpha_i \frac{\pi(p, w)}{w_i} \quad (4.38)$$

These output supply and input demand functions should be homogeneous of degree zero in all prices:

$$y(t p, t w) = y(p, w) \quad (4.39)$$

$$x_i(t p, t w) = x_i(p, w) \quad (4.40)$$

This condition is fulfilled for the output supply and input demand functions derived from a linearly homogeneous Cobb-Douglas profit function:

$$y(t p, t w) = \alpha_p \frac{\pi(t p, t w)}{t p} = \alpha_p \frac{t \pi(p, w)}{t p} = \alpha_p \frac{\pi(p, w)}{p} = y(p, w) \quad (4.41)$$

$$x_i(t p, t w) = -\alpha_i \frac{\pi(t p, t w)}{t w_i} = -\alpha_i \frac{t \pi(p, w)}{t w_i} = -\alpha_i \frac{\pi(p, w)}{w_i} = x_i(p, w) \quad (4.42)$$

4.3.9 Derived output supply and input demand elasticities

Based on the derived output supply function (4.37) and the derived input demand functions (4.38), we can derive the output supply elasticities and the (unconditional) input demand elasticities:

$$\epsilon_{yp}(p, w) = \frac{\partial y(p, w)}{\partial p} \frac{p}{y(p, w)} \quad (4.43)$$

$$= \frac{\alpha_p}{p} \frac{\partial \pi(p, w)}{\partial p} \frac{p}{y(p, w)} - \alpha_p \frac{\pi(p, w)}{p^2} \frac{p}{y(p, w)} \quad (4.44)$$

$$= \frac{\alpha_p}{p} y(p, w) \frac{p}{y(p, w)} - \alpha_p \frac{\pi(p, w)}{p y(p, w)} \quad (4.45)$$

$$= \alpha_p - \frac{\alpha_p}{r(w, y)} \quad (4.46)$$

$$= \alpha_p - 1 \quad (4.47)$$

$$\epsilon_{yj}(p, w) = \frac{\partial y(p, w)}{\partial w_j} \frac{w_j}{y(p, w)} \quad (4.48)$$

$$= \frac{\alpha_p}{p} \frac{\partial \pi(p, w)}{\partial w_j} \frac{w_j}{y(p, w)} \quad (4.49)$$

$$= -\frac{\alpha_p}{p} x_j(p, w) \frac{w_j}{y(p, w)} \quad (4.50)$$

$$= -\alpha_p \frac{\pi(p, w)}{p y(p, w)} \frac{w_j x_j(p, w)}{\pi(p, w)} \quad (4.51)$$

$$= \frac{\alpha_p r_j(w, y)}{r(w, y)} \quad (4.52)$$

$$= \alpha_j \quad (4.53)$$

$$\epsilon_{ip}(p, w) = \frac{\partial x_i(p, w)}{\partial p} \frac{p}{x_i(p, w)} \quad (4.54)$$

$$= -\frac{\alpha_i}{w_i} \frac{\partial \pi(p, w)}{\partial p} \frac{p}{x_i(p, w)} \quad (4.55)$$

$$= -\frac{\alpha_i}{w_i} y(p, w) \frac{p}{x_i(p, w)} \quad (4.56)$$

$$= -\alpha_i \frac{\pi(p, w)}{w_i x_i(p, w)} \frac{p y(p, w)}{\pi(p, w)} \quad (4.57)$$

$$= \frac{\alpha_i \alpha_p}{r_i(w, y)} \quad (4.58)$$

$$= \alpha_p \quad (4.59)$$

$$\epsilon_{ij}(p, w) = \frac{\partial x_i(p, w)}{\partial w_j} \frac{w_j}{x_i(p, w)} \quad (4.60)$$

$$= -\frac{\alpha_i}{w_i} \frac{\partial \pi(p, w)}{\partial w_j} \frac{w_j}{x_i(p, w)} + \delta_{ij} \alpha_i \frac{\pi(p, w)}{w_j^2} \frac{w_j}{x_i(p, w)} \quad (4.61)$$

$$= \frac{\alpha_i}{w_i} x_j(p, w) \frac{w_j}{x_i(p, w)} + \delta_{ij} \alpha_i \frac{\pi(p, w)}{w_i x_i(p, w)} \quad (4.62)$$

$$= \alpha_i \frac{\pi(p, w)}{w_i x_i(p, w)} \frac{w_j x_j(p, w)}{\pi(p, w)} - \delta_{ij} \frac{\alpha_i}{r_i(w, y)} \quad (4.63)$$

$$= \frac{\alpha_i r_j(w, y)}{r_i(w, y)} - \delta_{ij} \frac{\alpha_i}{r_i(w, y)} \quad (4.64)$$

$$= \alpha_j - \delta_{ij} \quad (4.65)$$

All derived input demand elasticities based on our Cobb-Douglas profit function with linear homogeneity imposed are presented in table 4.1. If the output price increases by one percent, the

profit-maximizing firm will increase the use of capital, labor, and materials by 2.75% each, which increases the production by 1.75%. The proportional increase of the input quantities (+2.75%) results in a less than proportional increase in the output quantity (+1.75%). This indicates that the model exhibits decreasing returns to scale, which is not surprising, because a profit maximum cannot be in an area of increasing returns to scale (if all inputs are variable and all markets function perfectly). If the price of capital increases by one percent, the profit-maximizing firm will decrease the use of capital by 1.82% and decrease the use of labor and materials by 0.82% each, which decreases the production by 0.82%. If the price of labor increases by one percent, the profit-maximizing firm will decrease the use of labor by 1.9% and decrease the use of capital and materials by 0.9% each, which decreases the production by 0.9%. If the price of materials increases by one percent, the profit-maximizing firm will decrease the use of materials by 1.02% and decrease the use of capital and labor by 0.02% each, which will decrease the production by 0.02%.

Table 4.1: Output supply and input demand elasticities derived from Cobb-Douglas profit function (with linear homogeneity imposed)

	p	w_{cap}	w_{lab}	w_{mat}
y	1.75	-0.82	-0.9	-0.02
x_{cap}	2.75	-1.82	-0.9	-0.02
x_{lab}	2.75	-0.82	-1.9	-0.02
x_{mat}	2.75	-0.82	-0.9	-1.02

4.4 Cobb-Douglas short-run profit function

4.4.1 Specification

The specification of a Cobb-Douglas short-run profit function is

$$\pi^v = A p^{\alpha_p} \left(\prod_{i \in N^1} w_i^{\alpha_i} \right) \left(\prod_{j \in N^2} x_j^{\alpha_j} \right), \quad (4.66)$$

This Cobb-Douglas short-run profit function can be linearized to

$$\ln \pi^v = \alpha_0 + \alpha_p \ln p + \sum_{i \in N^1} \alpha_i \ln w_i + \sum_{j \in N^2} \alpha_j \ln x_j \quad (4.67)$$

with $\alpha_0 = \ln A$.

4.4.2 Estimation

We can estimate a Cobb-Douglas short-run profit function with capital as a quasi-fixed input using the following commands. Again, function `lm` automatically removes the observations (apple

producers) with negative gross margin:

```
> profitCDSR <- lm( log( vProfit ) ~ log( pOut ) + log( pLab ) + log( pMat ) +
+   log( qCap ), data = dat )
> summary( profitCDSR )
```

Call:

```
lm(formula = log(vProfit) ~ log(pOut) + log(pLab) + log(pMat) +
    log(qCap), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-4.7422	-0.0646	0.2578	0.4931	0.8989

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	3.2739	1.2261	2.670	0.008571	**
log(pOut)	3.1745	0.2263	14.025	< 2e-16	***
log(pLab)	-1.6188	0.4434	-3.651	0.000381	***
log(pMat)	-0.7637	0.2687	-2.842	0.005226	**
log(qCap)	1.0960	0.1245	8.802	8.31e-15	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9659 on 127 degrees of freedom

(8 observations deleted due to missingness)

Multiple R-squared: 0.6591, Adjusted R-squared: 0.6484

F-statistic: 61.4 on 4 and 127 DF, p-value: < 2.2e-16

4.4.3 Properties

This short-run profit function fulfills all microeconomic monotonicity conditions: it is increasing in the output price, it is decreasing in the prices of all variable inputs, and it is increasing in the quasi-fixed input quantity. However, the homogeneity condition is not fulfilled, as the coefficient of the output price and the coefficients of the prices of the variable inputs do not sum up to one but to $3.17 + (-1.62) + (-0.76) = 0.79$.

4.4.4 Estimation with linear homogeneity in all prices imposed

We can impose the homogeneity condition on the Cobb-Douglas short-run profit function using the same method as for the Cobb-Douglas (long-run) profit function:

```
> profitCDSRHom <- lm( log( vProfit / pOut ) ~ log( pLab / pOut ) +
+   log( pMat / pOut ) + log( qCap ), data = dat )
> summary( profitCDSRHom )
```

Call:

```
lm(formula = log(vProfit/pOut) ~ log(pLab/pOut) + log(pMat/pOut) +
    log(qCap), data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-4.7302	-0.0677	0.2598	0.5160	0.8916

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	3.3145	1.2184	2.720	0.00743 **
log(pLab/pOut)	-1.4574	0.2252	-6.471	1.88e-09 ***
log(pMat/pOut)	-0.7156	0.2427	-2.949	0.00380 **
log(qCap)	1.0847	0.1212	8.949	3.50e-15 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9628 on 128 degrees of freedom

(8 observations deleted due to missingness)

Multiple R-squared: 0.5227, Adjusted R-squared: 0.5115

F-statistic: 46.73 on 3 and 128 DF, p-value: < 2.2e-16

We can obtain the coefficient of the output price from the homogeneity condition (4.15): $1 - (-1.457) - (-0.716) = 3.173$. All microeconomic monotonicity conditions are still fulfilled: the Cobb-Douglas short-run profit function with homogeneity imposed is increasing in the output price, decreasing in the prices of all variable inputs, and increasing in the quasi-fixed input quantity.

We can test the homogeneity restriction by a likelihood ratio test:

```
> lrtest( profitCDSRHom, profitCDSR )
```

Likelihood ratio test

Model 1: $\log(vProfit/pOut) \sim \log(pLab/pOut) + \log(pMat/pOut) + \log(qCap)$

Model 2: $\log(vProfit) \sim \log(pOut) + \log(pLab) + \log(pMat) + \log(qCap)$

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	5	-180.27			
2	6	-180.17	1	0.1859	0.6664

Given the large P -value, we can conclude that the data do not contradict the linear homogeneity in the output price and the prices of the variable inputs.

4.4.5 Returns to scale

The sum over the coefficients of all quasi-fixed inputs indicates the percentage change of the gross margin if the quantities of all quasi-fixed inputs are increased by one percent: If this sum is larger than one, the increase in gross margin is more than proportional to the increase in the quasi-fixed inputs. Hence, the technology has increasing returns to scale. If this sum over the coefficients of all quasi-fixed inputs is smaller than one, the increase in gross margin is less than proportional to the increase in the quasi-fixed inputs and the technology has decreasing returns to scale. As the coefficient of our (single) quasi-fixed input is larger than one (1.085), we can conclude that the technology has increasing returns to scale.

4.4.6 Shadow prices of quasi-fixed inputs

The partial derivatives of the short-run profit function with respect to the quantities of the quasi-fixed inputs denote the additional gross margins that can be earned by an additional unit of these quasi-fixed inputs. These internal marginal values of the quasi-fixed inputs are usually called “shadow prices”. In case of the Cobb-Douglas short-run profit function, the shadow prices can be computed by

$$\frac{\partial \pi^v}{\partial x_j} = \frac{\partial \ln \pi^v}{\partial \ln x_j} \frac{\pi^v}{x_j} = \alpha_j \frac{\pi^v}{x_j} \quad (4.68)$$

Before we can calculate the shadow price of the capital input, we need to calculate the predicted gross margin π^v . As the dependent variable of the Cobb-Douglas short-run profit function with homogeneity imposed is $\ln(\pi^v / \ln p)$, we have to apply the exponential function to the fitted dependent variable and then we have to multiply the result with p , in order obtain the fitted gross margins π^v . Furthermore, we have to be aware of that the `fitted` method only returns the predicted values for the observations that were included in the estimation. Hence, we have to make sure that the predicted gross margins are only assigned to the observations that have a positive gross margin and hence, were included in the estimation:

```
> dat$vProfitCDHom[ dat$vProfit > 0 ] <-  
+   exp( fitted( profitCDSRHom ) ) * dat$pOut[ dat$vProfit > 0 ]
```

Now, we can calculate the shadow price of the capital input for each apple producer who has a positive gross margin and hence, was included in the estimation:

```
> dat$pCapShadow <- with( dat, coef(profitCDSRHom)["log(qCap)"] *  
+   vProfitCDHom / qCap )
```

The following commands show the variation of the shadow prices of capital and compare them to the observed capital prices:

```
> hist( dat$pCapShadow, 30 )
> hist( dat$pCapShadow[ dat$pCapShadow < 30 ], 30 )
> compPlot( dat$pCap, dat$pCapShadow, log = "xy" )
```

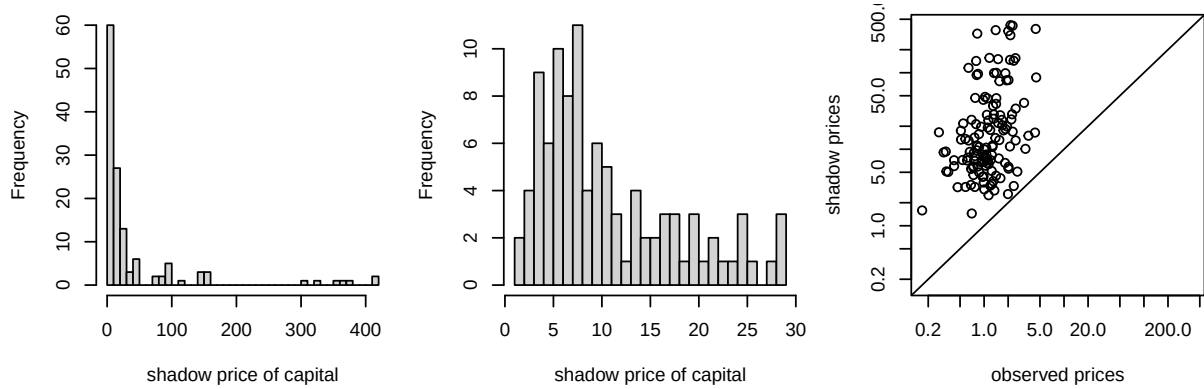


Figure 4.4: Shadow prices of capital

The resulting graphs are shown in figure 4.4. The two histograms show that most shadow prices are below 30 and many shadow prices are between 3 and 11 but there are also some apple producers who would gain much more from increasing their capital input. Indeed, all apple producers have a higher shadow price of capital than the observed price of capital, where the difference is small for some producers and large for other producers. These differences can be explained by risk aversion and market failures on the credit market or land market (e.g. marginal prices are not equal to average prices).

5 Efficiency Measures

5.1 Technical efficiency

5.1.1 Output-oriented technical efficiency with one output

The output-oriented technical efficiency according to Shepard is defined as

$$TE = \frac{y}{y^*} \Leftrightarrow y = TE \cdot y^* \quad 0 \leq TE \leq 1, \quad (5.1)$$

where y is the observed output quantity and y^* is the maximum output quantity that can be produced with the observed input quantities x .

The output-oriented technical efficiency according to Farrell is defined as

$$TE = \frac{y^*}{y} \Leftrightarrow y^* = TE \cdot y \quad TE \geq 1. \quad (5.2)$$

These efficiency measures are graphically illustrated in [Bogetoft and Otto \(2011, p. 26, figure 2.2\)](#).

5.1.2 Input-oriented technical efficiency with one input

The input-oriented technical efficiency according to Shepard is defined as

$$TE = \frac{x}{x^*} \Leftrightarrow x = TE \cdot x^* \quad TE \geq 1, \quad (5.3)$$

where x is the observed input quantity and x^* is the minimum input quantity at which the observed output quantities y can be produced.

The input-oriented technical efficiency according to Farrell is defined as

$$TE = \frac{x^*}{x} \Leftrightarrow x^* = TE \cdot x \quad 0 \leq TE \leq 1. \quad (5.4)$$

These efficiency measures are graphically illustrated in [Bogetoft and Otto \(2011, p. 26, figure 2.2\)](#).

5.1.3 Output-oriented technical efficiency with two or more outputs

The output-oriented technical efficiencies according to Shepard and Farrell assume a proportional increase of all output quantities, while all input quantities are held constant.

Hence, the output-oriented technical efficiency according to Shepard is defined as

$$TE = \frac{y_1}{y_1^*} = \frac{y_2}{y_2^*} = \dots = \frac{y_M}{y_M^*} \Leftrightarrow y_i = TE \cdot y_i^* \forall i \quad 0 \leq TE \leq 1, \quad (5.5)$$

where $y_1, y_2, \dots, y_M$ are the observed output quantities, $y_1^*, y_2^*, \dots, y_M^*$ are the maximum output quantities (given a proportional increase of all output quantities) that can be produced with the observed input quantities x , and M is the number of outputs.

The output-oriented technical efficiency according to Farrell is defined as

$$TE = \frac{y_1^*}{y_1} = \frac{y_2^*}{y_2} = \dots = \frac{y_M^*}{y_M} \Leftrightarrow y_i^* = TE \cdot y_i \forall i \quad TE \geq 1. \quad (5.6)$$

These efficiency measures are graphically illustrated in [Bogetoft and Otto \(2011, p. 27, figure 2.3, right panel\)](#).

5.1.4 Input-oriented technical efficiency with two or more inputs

The input-oriented technical efficiencies according to Shepard and Farrell assume a proportional reduction of all inputs, while all outputs are held constant.

Hence, the input-oriented technical efficiency according to Shepard is defined as

$$TE = \frac{x_1}{x_1^*} = \frac{x_2}{x_2^*} = \dots = \frac{x_N}{x_N^*} \Leftrightarrow x_i = TE \cdot x_i^* \forall i \quad TE \geq 1 \quad (5.7)$$

where $x_1, x_2, \dots, x_N$ are the observed input quantities, $x_1^*, x_2^*, \dots, x_N^*$ are the minimum input quantities (given a proportional decrease of all input quantities) at which the observed output quantities y can be produced, and N is the number of inputs.

The input-oriented technical efficiency according to Farrell is defined as

$$TE = \frac{x_1^*}{x_1} = \frac{x_2^*}{x_2} = \dots = \frac{x_N^*}{x_N} \Leftrightarrow x_i^* = TE \cdot x_i \forall i \quad 0 \leq TE \leq 1. \quad (5.8)$$

These efficiency measures are graphically illustrated in [Bogetoft and Otto \(2011, p. 27, figure 2.3, left panel\)](#).

5.2 Allocative efficiency, revenue efficiency, cost efficiency

5.2.1 Output-oriented allocative efficiency and revenue efficiency

According to equation (5.6), the output-oriented technical efficiency according to Farrell is

$$TE = \frac{\tilde{y}_1}{y_1} = \frac{\tilde{y}_2}{y_2} = \dots = \frac{\tilde{y}_M}{y_M} = \frac{p \cdot \tilde{y}}{p \cdot y}, \quad (5.9)$$

where $\tilde{y}$ is the vector of technically efficient output quantities and p is the vector of output prices.

Proof of the last equality in equation (5.9):

$$\frac{p \tilde{y}}{p y} = \frac{\sum_i p_i \tilde{y}_i}{\sum_i p_i y_i} = \frac{\sum_i p_i TE y_i}{\sum_i p_i y_i} = TE \frac{\sum_i p_i y_i}{\sum_i p_i y_i} = TE \quad (5.10)$$

The output-oriented allocative efficiency according to Farrell is defined as

$$AE = \frac{p y^*}{p \tilde{y}} = \frac{p \hat{y}}{p \tilde{y}}, \quad (5.11)$$

where y^* is the vector of technically efficient and allocatively efficient output quantities and $\hat{y}$ is the vector of output quantities so that $p \hat{y} = p y^*$ and $\hat{y}_i/\tilde{y}_i = AE \forall i$.

Finally, the revenue efficiency according to Farrell is

$$RE = \frac{p y^*}{p y} = \frac{p y^*}{p \tilde{y}} \frac{p \tilde{y}}{p y} = AE \cdot TE \quad (5.12)$$

All these efficiency measures can also be specified according to Shepard by just taking the inverse of the Farrell specifications. These efficiency measures are graphically illustrated in [Bogetoft and Otto \(2011, p. 40, figure 2.11\)](#).

5.2.2 Input-oriented allocative efficiency and cost efficiency

According to equation (5.8), the input-oriented technical efficiency according to Farrell is

$$TE = \frac{\tilde{x}_1}{x_1} = \frac{\tilde{x}_2}{x_2} = \dots = \frac{\tilde{x}_N}{x_N} = \frac{w \tilde{x}}{w x}, \quad (5.13)$$

where $\tilde{x}$ is the vector of technically efficient input quantities and w is the vector of output prices.

The input-oriented allocative efficiency according to Farrell is defined as

$$AE = \frac{w x^*}{w \tilde{x}} = \frac{w \hat{x}}{w \tilde{x}}, \quad (5.14)$$

where x^* is the vector of technically efficient and allocatively efficient input quantities and $\hat{x}$ is the vector of output quantities so that $w \hat{x} = w x^*$ and $\hat{x}_i/\tilde{x}_i = AE \forall i$.

Finally, the cost efficiency according to Farrell is

$$CE = \frac{w x^*}{w x} = \frac{w x^*}{w \tilde{x}} \frac{w \tilde{x}}{w x} = AE \cdot TE \quad (5.15)$$

All these efficiency measures can also be specified according to Shepard by just taking the inverse of the Farrell specifications. These efficiency measures are graphically illustrated in [Bogetoft and Otto \(2011, p. 36, figure 2.9\)](#).

5.3 Profit efficiency

The profit efficiency according to Farrell is defined as

$$PE = \frac{p y^* - w x^*}{p y - w x}, \quad (5.16)$$

where y^* and x^* denote the profit maximizing output quantities and input quantities, respectively (assuming full technical efficiency). The profit efficiency according to Shepard is just the inverse of the Farrell specifications.

5.4 Scale efficiency

In case of one input x and one output $y = f(x)$, the scale efficiency according to Farrell is defined as

$$SE = \frac{AP^*}{AP}, \quad (5.17)$$

where $AP = f(x)/x$ is the observed average product $AP^* = f(x^*)/x^*$ is the maximum average product, and x^* is the input quantity that results in the maximum average product.

The first-order condition for a maximum of the average product is

$$\frac{\partial AP}{\partial x} = \frac{\partial f(x)}{\partial x} \frac{1}{x} - \frac{f(x)}{x^2} = 0 \quad (5.18)$$

This condition can be re-arranged to

$$\frac{\partial f(x)}{\partial x} \frac{x}{f(x)} = 1 \quad (5.19)$$

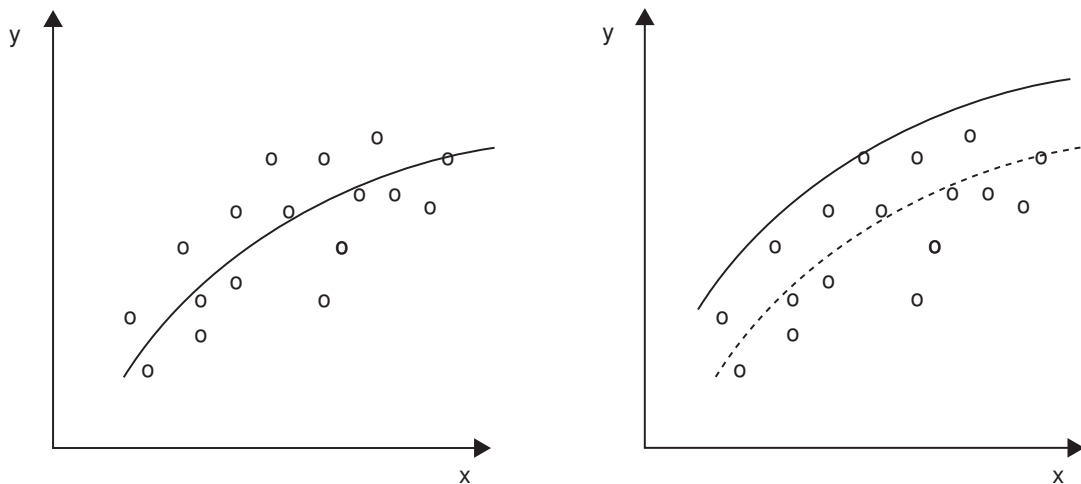
Hence, a necessary (but not sufficient) condition for a maximum of the average product is an elasticity of scale equal to one.

6 Stochastic Frontier Analysis

6.1 Stochastic production frontiers

6.1.1 Specification

In section 2, we have estimated *average* production functions, where about half of the observations were below the estimated production function and about half of the observations were above the estimated production function (see left panel of figure 6.1). However, in microeconomic theory, the production function indicates the maximum output quantity for each given set of input quantities. Hence, theoretically, no observation could be above the production function and an observations below the production function would indicate technical inefficiency.



Source: [Bogetoft and Otto \(2011\)](#)

Figure 6.1: Production function estimation: ordinary regression and with intercept correction

This means that all residuals must be negative or zero. A production function with only non-positive residuals could look like:

$$\ln y = \ln f(x) - u \quad \text{with} \quad u \geq 0, \quad (6.1)$$

where $-u \leq 0$ are the non-positive residuals. One solution to achieve this could be to estimate an average production function by ordinary least squares and then simply shift the production function up until all residuals are negative or zero (see right panel of figure 6.1). However, this

procedure does not account for statistical noise and is very sensitive to positive outliers.¹ As virtually all data sets and models are flawed with statistical noise, e.g. due to measurement errors, omitted variables, and approximation errors, [Meeusen and van den Broeck \(1977\)](#) and [Aigner, Lovell, and Schmidt \(1977\)](#) independently proposed the stochastic frontier model that simultaneously accounts for statistical noise and technical inefficiency:

$$\ln y = \ln f(x) - u + v \quad \text{with} \quad u \geq 0, \quad (6.2)$$

where $-u \leq 0$ accounts for technical inefficiency and v accounts for statistical noise. This model can be re-written (see, e.g. [Coelli et al., 2005](#), p. 243):

$$y = f(x) e^{-u} e^v \quad (6.3)$$

As explained in section 5.1, Shepard's output-oriented technical efficiencies are defined as the ratio between the observed output and the frontier output (see also, e.g., [Coelli et al., 2005](#), p. 244):

$$TE = \frac{y}{f(x) e^v} = \frac{f(x) e^{-u} e^v}{f(x) e^v} = e^{-u}, \quad (6.4)$$

while Farrell's output-oriented technical efficiencies are defined as the ratio between the frontier output and the observed output:

$$TE = \frac{f(x) e^v}{y} = \frac{f(x) e^v}{f(x) e^{-u} e^v} = e^u. \quad (6.5)$$

As Shepard's output-oriented technical efficiencies are much more frequently used than Farrell's output-oriented technical efficiencies, we will continue using Shepard's output-oriented technical efficiencies.

The stochastic frontier model is usually estimated by an econometric maximum likelihood estimation, which requires distributional assumptions of the error terms. Most often, it is assumed that the noise term v follows a normal distribution with zero mean and constant variance σ_v^2 , the inefficiency term u follows a positive half-normal distribution or a positive truncated normal distribution with constant scale parameter σ_u^2 , and all vs and all us are independent:

$$v \sim N(0, \sigma_v^2) \quad (6.6)$$

$$u \sim N^+(\mu, \sigma_u^2), \quad (6.7)$$

where $\mu = 0$ for a positive half-normal distribution and $\mu \neq 0$ for a positive truncated normal distribution. These assumptions result in a left-skewed distribution of the total error terms $\varepsilon = -u + v$, i.e. the density function is flat on the left and steep on the right. Hence, it is very rare that a firm has a large positive residual (much higher output than the production function) but

¹This is also true for the frequently-used Data Envelopment Analysis (DEA).

it is not so rare that a firm has a large negative residual (much lower output than the production function).

6.1.1.1 Marginal products and output elasticities in SFA models

Given the multiplicative specification of stochastic production frontier models (6.3) and assuming that the random error v is zero, we can see that the marginal products are downscaled by the level of the technical efficiency:

$$\frac{\partial y}{\partial x_i} = \frac{\partial f(x)}{\partial x_i} e^{-u} = TE \frac{\partial f(x)}{\partial x_i} \quad (6.8)$$

However, the partial production elasticities are unaffected by the efficiency level:

$$\frac{\partial y}{\partial x_i} \frac{x_i}{y} = \frac{\partial f(x)}{\partial x_i} e^{-u} \frac{x_i}{f(x)e^{-u}} = \frac{\partial f(x)}{\partial x_i} \frac{x_i}{f(x)} = \frac{\partial \ln f(x)}{\partial \ln x_i} \quad (6.9)$$

As the output elasticities do not depend on the firm's technical efficiency, also the elasticity of scale does not depend on the firm's technical efficiency.

6.1.2 Skewness of residuals from OLS estimations

The following commands plot histograms of the residuals taken from the Cobb-Douglas and the Translog production function:

```
> hist( residuals( prodCD ), 15 )
> hist( residuals( prodTL ), 15 )
```

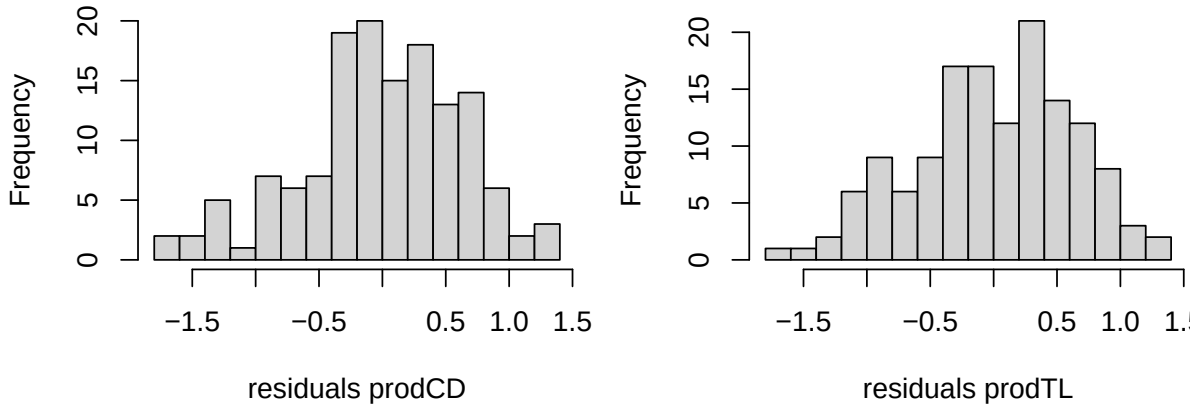


Figure 6.2: Residuals of Cobb-Douglas and Translog production functions

The resulting graphs are shown in figure 6.2. The residuals of both production functions are left-skewed. This visual assessment of the skewness can be confirmed by calculating the skewness using the function `skewness` that is available in the package *moments*:

```
> library( "moments" )
> skewness( residuals( prodCD ) )
```

```
[1] -0.4191323
```

```
> skewness( residuals( prodTL ) )
```

```
[1] -0.3194211
```

As a negative skewness means that the residuals are left-skewed, it is likely that not all apple producers are fully technically efficient.

However, the distribution of the residuals does not always have the expected skewness. Possible reasons for an unexpected skewness of OLS residuals are explained in section 6.2.2.

6.1.3 Cobb-Douglas stochastic production frontier

6.1.3.1 Estimation

We can use the command `sfa` (package *frontier*) to estimate stochastic production frontiers. The basic syntax of the command `sfa` is similar to the syntax of the command `lm`. The following command estimates a Cobb-Douglas stochastic production frontier assuming that the inefficiency term u follows a positive halfnormal distribution:

```
> library( "frontier" )
> prodCDSfa <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ),
+   data = dat )
> summary( prodCDSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 12 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	0.228813	1.247739	0.1834	0.8544981	
log(qCap)	0.160934	0.081883	1.9654	0.0493668	*
log(qLab)	0.684777	0.146797	4.6648	3.089e-06	***
log(qMat)	0.465871	0.131588	3.5404	0.0003996	***
sigmaSq	1.000040	0.202456	4.9396	7.830e-07	***
gamma	0.896664	0.070952	12.6375	< 2.2e-16	***

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -133.8893
```

```
cross-sectional data
```

```
total number of observations = 140
```

```
mean efficiency: 0.5379937
```

The parameters of the Cobb-Douglas production frontier can be interpreted as before. The estimated production function is monotonically increasing in all inputs. The output elasticity of capital is 0.161, the output elasticity of labor is 0.685, The output elasticity of materials is 0.466, and the elasticity of scale is 1.312.

6.1.3.2 Variance parameters

The estimation algorithm re-parameterizes the variance parameter of the noise term (σ_v^2) and the scale parameter of the inefficiency term (σ_u^2) and instead estimates the parameters $\sigma^2 = \sigma_v^2 + \sigma_u^2$ and $\gamma = \sigma_u^2 / \sigma^2$. The parameter γ lies between zero and one and indicates the importance of the inefficiency term. If γ is zero, the inefficiency term u is irrelevant and the results should be equal to OLS results. In contrast, if γ is one, the noise term v is irrelevant and all deviations from the production frontier are explained by technical inefficiency. As the estimate of γ is 0.897, we can conclude that both statistical noise and inefficiency are important for explaining deviations from the production function but that inefficiency is more important than noise. As σ_u^2 is not equal to the variance of the inefficiency term u , the estimated parameter γ cannot be interpreted as the proportion of the total variance that is due to inefficiency. In fact, the variance of the inefficiency term u is

$$Var(u) = \sigma_u^2 \left[1 - \frac{\frac{\mu}{\sigma_u} \phi\left(\frac{\mu}{\sigma_u}\right)}{\Phi\left(\frac{\mu}{\sigma_u}\right)} - \left(\frac{\phi\left(\frac{\mu}{\sigma_u}\right)}{\Phi\left(\frac{\mu}{\sigma_u}\right)} \right)^2 \right], \quad (6.10)$$

where $\Phi(\cdot)$ indicates the cumulative distribution function and $\phi(\cdot)$ the probability density function of the standard normal distribution. If the inefficiency term u follows a positive halfnormal distribution (i.e. $\mu = 0$), the above equation reduces to

$$Var(u) = \sigma_u^2 \left[1 - (2 \phi(0))^2 \right], \quad (6.11)$$

We can calculate the estimated variances of the inefficiency term u and the noise term v by following commands:

```
> gamma <- unname( coef(prodCDSfa) ["gamma"] )
[1] 0.8966641
```

```
> sigmaSq <- unname( coef(prodCDSfa)["sigmaSq"] )
```

```
[1] 1.00004
```

```
> sigmaSqU <- gamma * sigmaSq
```

```
[1] 0.8966997
```

```
> varU <- sigmaSqU * ( 1 - ( 2 * dnorm(0) )^2 )
```

```
[1] 0.3258429
```

```
> varV <- sigmaSqV <- ( 1 - gamma ) * sigmaSq
```

```
[1] 0.10334
```

Hence, the proportion of the total variance ($Var(-u + v) = Var(u) + Var(v)$)² that is due to inefficiency is estimated to be:

```
> varU / ( varU + varV )
```

```
[1] 0.7592169
```

This indicates that around 75.9% of the total variance is due to inefficiency.

The *frontier* package calculates these additional variance parameters (and some further variance parameters) automatically, if argument `extraPar` of the `summary()` method is set to `TRUE`:

```
> summary( prodCDSfa, extraPar = TRUE )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 12 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.228813	1.247739	0.1834	0.8544981
log(qCap)	0.160934	0.081883	1.9654	0.0493668 *
log(qLab)	0.684777	0.146797	4.6648	3.089e-06 ***
log(qMat)	0.465871	0.131588	3.5404	0.0003996 ***

²This equation relies on the assumption that the inefficiency term u and the noise term v are independent, i.e. their covariance is zero.

```

sigmaSq      1.000040    0.202456    4.9396    7.830e-07 ***
gamma        0.896664    0.070952   12.6375    < 2.2e-16 ***
sigmaSqU     0.896700    0.241715    3.7097    0.0002075 ***
sigmaSqV     0.103340    0.055831    1.8509    0.0641777 .
sigma        1.000020    0.101226    9.8791    < 2.2e-16 ***
sigmaU       0.946942    0.127629    7.4195    1.176e-13 ***
sigmaV       0.321465    0.086838    3.7019    0.0002140 ***
lambdaSq     8.677179    6.644542    1.3059    0.1915829
lambda       2.945705    1.127835    2.6118    0.0090061 **
varU         0.325843             NA             NA             NA
sdU          0.570827             NA             NA             NA
gammaVar     0.759217             NA             NA             NA

```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -133.8893

cross-sectional data

total number of observations = 140

mean efficiency: 0.5379937

The additionally returned parameter are defined as follows: $\text{sigmaSqU} = \sigma_u^2 = \sigma^2 \cdot \gamma$, $\text{sigmaSqV} = \sigma_v^2 = \sigma^2 \cdot (1 - \gamma) = \text{Var}(v)$, $\text{sigma} = \sigma = \sqrt{\sigma^2}$, $\text{sigmaU} = \sigma_u = \sqrt{\sigma_u^2}$, $\text{sigmaV} = \sigma_v = \sqrt{\sigma_v^2}$, $\text{lambdaSq} = \lambda^2 = \sigma_u^2 / \sigma_v^2$, $\text{lambda} = \lambda = \sigma_u / \sigma_v$, $\text{varU} = \text{Var}(u)$, $\text{sdU} = \sqrt{\text{Var}(u)}$, and $\text{gammaVar} = \text{Var}(u) / (\text{Var}(u) + \text{Var}(v))$.

6.1.3.3 Statistical tests for inefficiencies

If there would be no inefficiencies, i.e. $u = 0$ for all observations, coefficient γ would be equal to zero. Hence, one can test the null hypothesis of no inefficiencies by simply testing whether γ is equal to (does not significantly deviate from) zero. However, a t -test of the null hypothesis $\gamma = 0$ (e.g. reported in the output of the `summary` method) is not valid, because γ is bound to the interval $[0, 1]$ and hence, cannot follow a t -distribution. Instead, we can use a likelihood ratio test to check whether adding the inefficiency term u significantly improves the fit of the model. If the `lrtest` method is called just with a single stochastic frontier model, it compares the stochastic frontier model with the corresponding OLS model (i.e. a model with γ equal to zero):

```
> lrtest( prodCDSfa )
```

Likelihood ratio test


```

Model 1: OLS (no inefficiency)
Model 2: Error Components Frontier (ECF)
#Df LogLik Df Chisq Pr(>Chisq)
1 5 -137.61
2 6 -133.89 1 7.4387 0.003192 **
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Under the null hypothesis (no inefficiency, only noise), the test statistic asymptotically follows a mixed χ^2 -distribution (Coelli, 1995).³ The rather small P-value indicates that the data clearly reject the OLS model in favor of the stochastic frontier model, i.e. there is significant technical inefficiency.

6.1.3.4 Obtaining technical efficiency estimates

As neither the noise term v nor the inefficiency term u but only the total error term $\varepsilon = -u + v$ is known, the technical efficiencies $TE = e^{-u}$ are generally unknown. However, given that the parameter estimates (including the parameters σ^2 and γ or σ_v^2 and σ_u^2) and the total error term ε are known, it is possible to determine the expected value of the technical efficiency (see, e.g. Coelli et al., 2005, p. 255):

$$\widehat{TE} = E[e^{-u}] \quad (6.12)$$

These efficiency estimates can be obtained by the `efficiencies` method:

```
> dat$effCD <- efficiencies( prodCDSfa )
```

The following two commands create a scatter plot that visualizes the relationship between the residuals ($y - f(x) = \varepsilon = -u + v$) and the efficiency estimates ($\widehat{TE} = E[e^{-u}]$) and a histogram that visualizes the variation of the efficiency estimates, respectively:

```
> plot( residuals( prodCDSfa ), dat$effCD )
> hist( dat$effCD, 15 )
```

The resulting graphs are shown in figure 6.3. The efficiency estimates are rather low: the firms only produce between 10% and 90% of the maximum possible output quantities.

In order to illustrate how efficiency estimates are obtained, e.g., with the `efficiencies()` method, we manually calculate the approximate efficiency estimate of the first observation:

```
> # choose the observation number
> obsNo <- 1
> # print the automatically obtained efficiency estimate for this observation
> dat$effCD[ obsNo ]
```

³As a standard likelihood ratio test assumes that the test statistic follows a (standard) χ^2 -distribution under the null hypothesis, a test that is conducted by the command `lrtest( prodCD, prodCDSfa )` returns an incorrect P-value.

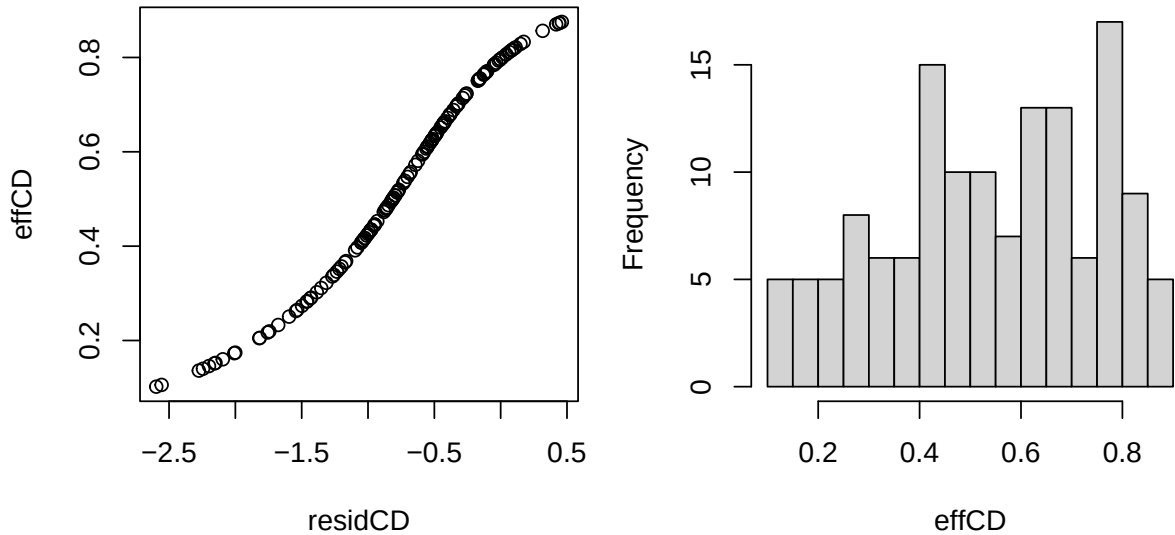


Figure 6.3: Efficiency estimates of Cobb-Douglas production frontier

```
[1] 0.262434
```

```
> # obtain the residual of this observation
> r <- residuals( prodCDSfa )[ obsNo ]
> # create a vector with a range of technical efficiencies
> TE <- c( 1:100 ) / 100
> # calculate values of the inefficiency term u that correspond to the values of TE
> u <- - log( TE )
> # calculate values of the noise term v that correspond to the values of TE
> v <- r + u
> # obtain sigma_U
> su <- coef( summary( prodCDSfa, extraPar = TRUE ) )[ "sigmaU", 1 ]
> # sigma_V
> sv <- coef( summary( prodCDSfa, extraPar = TRUE ) )[ "sigmaV", 1 ]
> # relative probabilities of the values of TE
> prTE <- dnorm( v, sd = sv ) * ( 2 * dnorm( u, sd = su ) )
> # probabilities of the values of TE
> pTE <- prTE / sum( prTE )
> # approximate expected TE
> sum( TE * pTE )
```

```
[1] 0.287908
```

We can also illustrate the probabilities of the technical efficiencies of the first observation in a density plot:

```
> plot( TE, pTE, type = "l" )
```

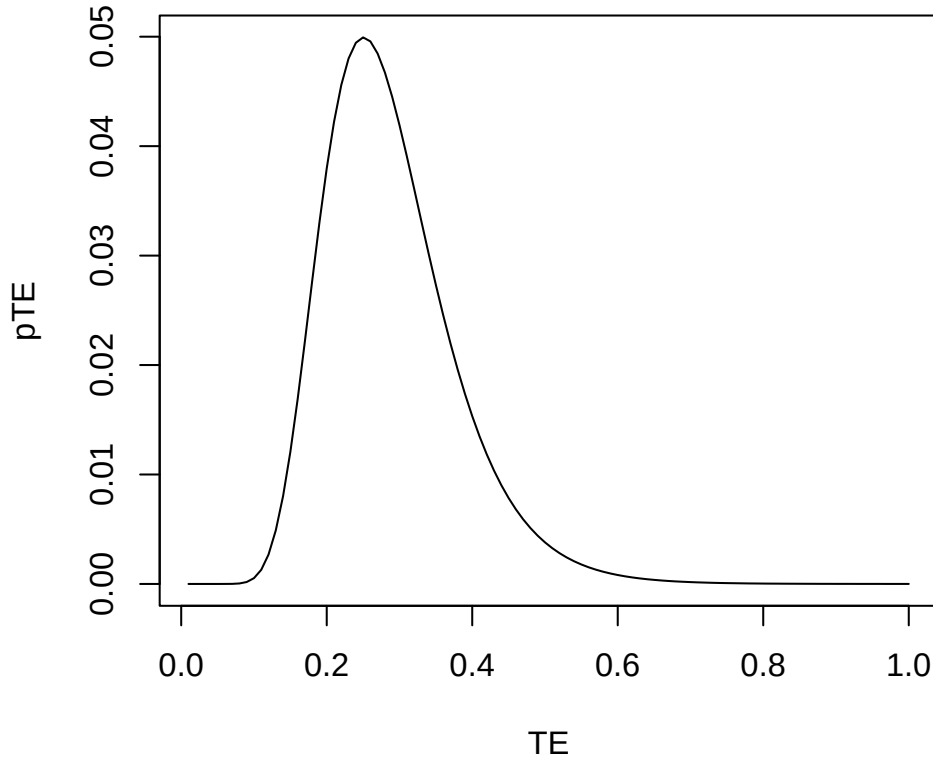


Figure 6.4: Probabilities of the technical efficiencies of the first observation

The resulting graph is shown in figure 6.4.

We explore the correlation between firm size (measured as output quantity as well as aggregate input quantity indicated by a Fisher quantity index of all inputs) and the efficiency estimates:

```
> plot( dat$qOut, dat$effCD, log = "x" )
> plot( dat$X, dat$effCD, log = "x" )
```

The resulting graphs are shown in figure 6.5. As the efficiency directly influences the output quantity, it is not surprising that the efficiency estimates are highly correlated with the output quantity. On the other hand, the efficiency estimates are only slightly correlated with firm size measured as aggregate input quantity. However, the largest firms all have an above-average efficiency estimate.

6.1.3.5 Truncated normal distribution of the inefficiency term

Instead of assuming that the inefficiency term u follows a half-normal distribution, we can assume that it follows a truncated norm distribution:

$$u \sim N^+(\mu, \sigma_u^2) \quad (6.13)$$

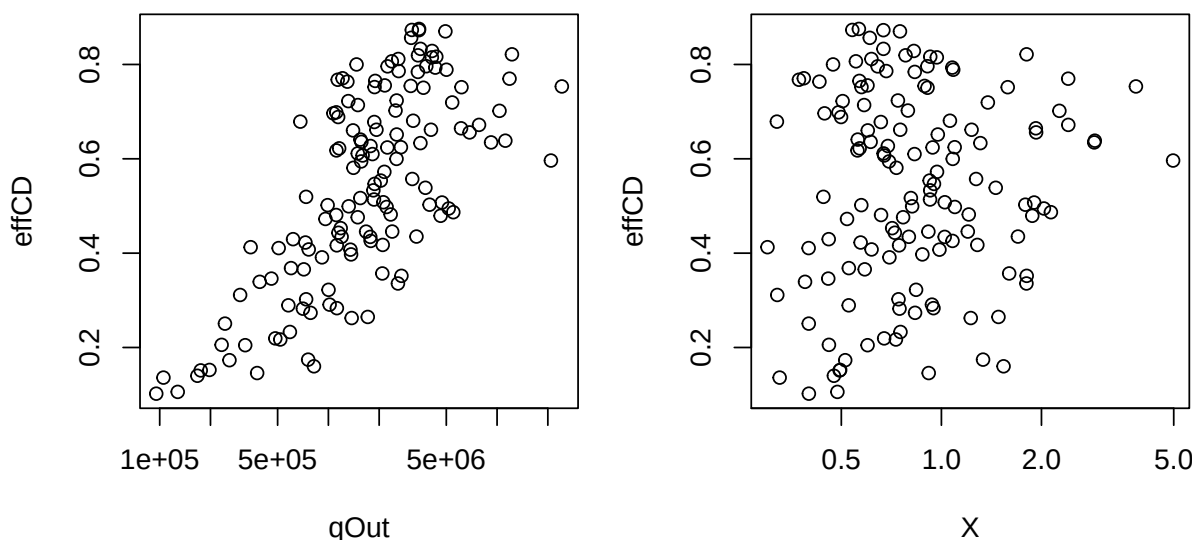


Figure 6.5: Firm size and efficiency estimates of Cobb-Douglas production frontier

If the location parameter μ is equal to zero, the truncated normal distribution is identical to the half-normal distribution.

If argument `truncNorm` of function `sfa()` of the *frontier* package is set to `TRUE`, this function estimates a stochastic frontier model assuming a truncated normal distribution of the inefficiency term:

```
> prodCDSfaTn <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ),
+   data = dat, truncNorm = TRUE )
> summary( prodCDSfaTn, extraPar = TRUE )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 13 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.227323	1.250360	0.1818	0.8557349
log(qCap)	0.160632	0.079397	2.0232	0.0430568 *
log(qLab)	0.688275	0.148699	4.6286	3.681e-06 ***
log(qMat)	0.464421	0.132878	3.4951	0.0004739 ***
sigmaSq	0.934405	0.599621	1.5583	0.1191562
gamma	0.895931	0.069525	12.8865	< 2.2e-16 ***

```

mu          0.130129   1.145962   0.1136   0.9095912
sigmaSqU    0.837163   0.557136   1.5026   0.1329373
sigmaSqV    0.097242   0.077929   1.2478   0.2120920
sigma       0.966646   0.310155   3.1167   0.0018292 **
sigmaU      0.914966   0.304457   3.0052   0.0026537 **
sigmaV      0.311837   0.124951   2.4957   0.0125720 *
lambdaSq    8.609044   6.419490   1.3411   0.1798948
lambda      2.934117   1.093939   2.6822   0.0073149 **
varU        0.331133           NA           NA           NA
sdU         0.575442           NA           NA           NA
gammaVar     0.772998           NA           NA           NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -133.8836

```

```

cross-sectional data
total number of observations = 140

```

```
mean efficiency: 0.5272536
```

The results still indicate statistical significant technical inefficiency:

```
> lrtest( prodCDSfaTn )
```

```
Likelihood ratio test
```

```
Model 1: OLS (no inefficiency)
```

```
Model 2: Error Components Frontier (ECF)
```

```

#Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -137.61
2    7 -133.88  2   7.45    0.0092 **

```

```

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

However, neither the t -test presented in the `summary` output above nor a likelihood ratio test can reject the model with the half-normal distribution of the inefficiency term (i.e., with $\mu = 0$) in favour of the model with truncated normal distribution of the inefficiency term:

```
> lrtest( prodCDSfa, prodCDSfaTn )
```

```
Likelihood ratio test
```

```

Model 1: prodCDSfa
Model 2: prodCDSfaTn
#Df LogLik Df Chisq Pr(>Chisq)
1 6 -133.89
2 7 -133.88 1 0.0113 0.9153

```

Given the high P-values of these two tests, it seems to be reasonable to use the model with the half-normal distribution of the error term.

6.1.4 Translog production frontier

6.1.4.1 Estimation

As the Cobb-Douglas functional form is very restrictive, we additionally estimate a Translog stochastic production frontier:

```

> prodTLSfa <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat )
+   + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qLab )^2 )
+   + I( 0.5 * log( qMat )^2 ) + I( log( qCap ) * log( qLab ) )
+   + I( log( qCap ) * log( qMat ) ) + I( log( qLab ) * log( qMat ) ),
+   data = dat )
> summary( prodTLSfa, extraPar = TRUE )

```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 23 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-8.8110424	19.9181626	-0.4424	0.6582271
log(qCap)	-0.6332521	2.0855273	-0.3036	0.7614012
log(qLab)	4.4511064	4.4552358	0.9991	0.3177593
log(qMat)	-1.3976309	3.8097808	-0.3669	0.7137284
I(0.5 * log(qCap)^2)	0.0053258	0.1866174	0.0285	0.9772324
I(0.5 * log(qLab)^2)	-1.5030433	0.6812813	-2.2062	0.0273700 *
I(0.5 * log(qMat)^2)	-0.5113559	0.3733348	-1.3697	0.1707812
I(log(qCap) * log(qLab))	0.4187529	0.2747251	1.5243	0.1274434
I(log(qCap) * log(qMat))	-0.4371561	0.1902856	-2.2974	0.0215978 *
I(log(qLab) * log(qMat))	0.9800294	0.4216637	2.3242	0.0201150 *

```

sigmaSq          0.9587307  0.1968009  4.8716  1.107e-06 ***
gamma            0.9153387  0.0647478 14.1370 < 2.2e-16 ***
sigmaSqU         0.8775633  0.2328364  3.7690  0.0001639 ***
sigmaSqV         0.0811674  0.0497448  1.6317  0.1027476
sigma            0.9791480  0.1004960  9.7432 < 2.2e-16 ***
sigmaU           0.9367835  0.1242744  7.5380  4.771e-14 ***
sigmaV           0.2848989  0.0873025  3.2634  0.0011010 **
lambdaSq        10.8117751  9.0334818  1.1969  0.2313628
lambda           3.2881264  1.3736518  2.3937  0.0166789 *
varU             0.3188892           NA      NA      NA
sdU              0.5647027           NA      NA      NA
gammaVar         0.7971103           NA      NA      NA

```

```
---
```

```

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -128.0684

```

```
cross-sectional data
```

```
total number of observations = 140
```

```
mean efficiency: 0.5379939
```

6.1.4.2 Statistical test for inefficiencies

A likelihood ratio test confirms that the stochastic frontier model fits the data much better than an average production function estimated by OLS:

```
> lrtest( prodTLSfa )
```

```
Likelihood ratio test
```

```
Model 1: OLS (no inefficiency)
```

```
Model 2: Error Components Frontier (ECF)
```

```

#Df  LogLik Df  Chisq Pr(>Chisq)
1   11 -131.25
2   12 -128.07  1  6.353  0.005859 **

```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.1.4.3 Testing against the Cobb-Douglas functional form

A further likelihood ratio test indicates that it is not really clear whether the Translog stochastic frontier model fits the data significantly better than the Cobb-Douglas stochastic frontier model:

```
> lrtest( prodCDSfa, prodTLSfa )
```

Likelihood ratio test

Model 1: prodCDSfa

Model 2: prodTLSfa

```
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    6 -133.89
2   12 -128.07  6 11.642    0.07045 .
---
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

While the Cobb-Douglas functional form cannot be rejected at the 5% significance level, it is rejected in favor of the Translog functional form at the 10% significance level.

6.1.4.4 Obtaining technical efficiency estimates

The efficiency estimates based on the Translog stochastic production frontier can be obtained (again) by the `efficiencies` method:

```
> dat$effTL <- efficiencies( prodTLSfa )
```

The following commands illustrate their variation, their correlation with the output level, and their correlation with the firm size (measured as input use):

```
> hist( dat$effTL, 15 )
> plot( dat$qOut, dat$effTL, log = "x" )
> plot( dat$X, dat$effTL, log = "x" )
```

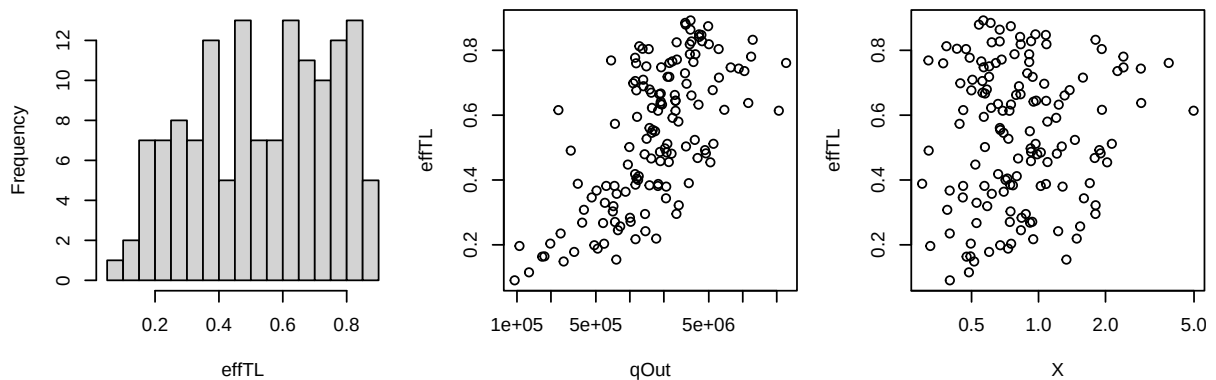


Figure 6.6: Efficiency estimates of Translog production frontier

The resulting graphs are shown in figure 6.6. These efficiency estimates are rather similar to the efficiency estimates based on the Cobb-Douglas stochastic production frontier. This is confirmed by a direct comparison of these efficiency estimates:


```
> compPlot( dat$effCD, dat$effTL )
```

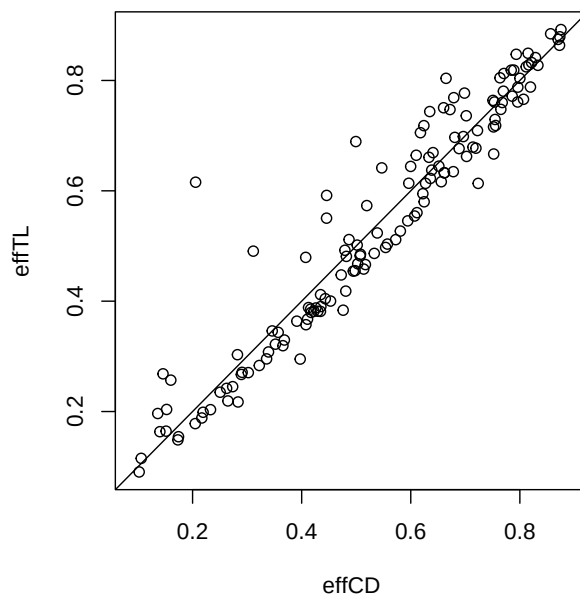


Figure 6.7: Efficiency estimates of Cobb-Douglas and Translog production frontier

The resulting graph is shown in figure 6.7. Most efficiency estimates only slightly differ between the two functional forms but a few efficiency estimates are considerably higher for the Translog functional form. The inflexibility of the Cobb-Douglas functional form probably resulted in an insufficient adaptation of the frontier to some observations, which lead to larger negative residuals and hence, lower efficiency estimates in the Cobb-Douglas model.

6.1.4.5 Truncated normal distribution of the inefficiency term

As explained in section 6.1.3.5, we can estimate a stochastic frontier model based on the assumption that the inefficiency term u follows a truncated normal distribution instead of a half-normal distribution:

```
> prodTLSfaTn <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat )
+   + I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qLab )^2 )
+   + I( 0.5 * log( qMat )^2 ) + I( log( qCap ) * log( qLab ) )
+   + I( log( qCap ) * log( qMat ) ) + I( log( qLab ) * log( qMat ) ),
+   data = dat, truncNorm = TRUE )
> summary( prodTLSfaTn, extraPar = TRUE )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

6 Stochastic Frontier Analysis

Iterative ML estimation terminated after 30 iterations:
log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-15.759268	21.912418	-0.7192	0.472022
log(qCap)	-0.748266	2.157070	-0.3469	0.728674
log(qLab)	5.349481	4.806460	1.1130	0.265718
log(qMat)	-0.979857	3.971830	-0.2467	0.805139
I(0.5 * log(qCap)^2)	-0.050523	0.184108	-0.2744	0.783760
I(0.5 * log(qLab)^2)	-1.612095	0.723666	-2.2277	0.025902 *
I(0.5 * log(qMat)^2)	-0.552201	0.399247	-1.3831	0.166633
I(log(qCap) * log(qLab))	0.471927	0.287449	1.6418	0.100636
I(log(qCap) * log(qMat))	-0.428907	0.199232	-2.1528	0.031334 *
I(log(qLab) * log(qMat))	0.969984	0.448343	2.1635	0.030504 *
sigmaSq	0.644764	0.221585	2.9098	0.003617 **
gamma	0.939671	0.067754	13.8689	< 2.2e-16 ***
mu	0.658349	0.413449	1.5923	0.111310
sigmaSqU	0.605866	0.213788	2.8340	0.004598 **
sigmaSqV	0.038898	0.045373	0.8573	0.391281
sigma	0.802972	0.137978	5.8196	5.900e-09 ***
sigmaU	0.778374	0.137330	5.6679	1.445e-08 ***
sigmaV	0.197225	0.115028	1.7146	0.086420 .
lambdaSq	15.575816	18.615851	0.8367	0.402763
lambda	3.946621	2.358454	1.6734	0.094250 .
varU	0.353965	NA	NA	NA
sdU	0.594950	NA	NA	NA
gammaVar	0.900989	NA	NA	NA

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -127.581

cross-sectional data

total number of observations = 140

mean efficiency: 0.4613383

The fit of the Translog stochastic production frontier with truncated normal inefficiency term is significantly better than the fit of the Cobb-Douglas stochastic production frontier with truncated

normal inefficiency term (at 5% significance level):

```
> lrtest( prodCDSfaTn, prodTLSfaTn )
```

Likelihood ratio test

Model 1: prodCDSfaTn

Model 2: prodTLSfaTn

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	7	-133.88			
2	13	-127.58	6	12.605	0.04975 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The results indicate statistical significant technical inefficiency:

```
> lrtest( prodTLSfaTn )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Error Components Frontier (ECF)

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	11	-131.25			
2	13	-127.58	2	7.3279	0.009803 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

However, neither the t -test presented in the **summary** output above nor a likelihood ratio test can reject the model with the half-normal distribution of the inefficiency term (i.e., with $\mu = 0$) in favour of the model with truncated normal distribution of the inefficiency term (at 10% significance level):

```
> lrtest( prodTLSfa, prodTLSfaTn )
```

Likelihood ratio test

Model 1: prodTLSfa

Model 2: prodTLSfaTn

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	12	-128.07			
2	13	-127.58	1	0.9748	0.3235

6.1.5 Translog production frontier with mean-scaled variables

As argued in section 2.6.14, it is sometimes convenient to estimate a Translog production (frontier) function with mean-scaled variables. The following command estimates a Translog production function with mean-scaled output and input quantities:

```
> prodTLmSfa <- sfa( log( qmOut ) ~ log( qmCap ) + log( qmLab ) + log( qmMat )
+   + I( 0.5 * log( qmCap )^2 ) + I( 0.5 * log( qmLab )^2 )
+   + I( 0.5 * log( qmMat )^2 ) + I( log( qmCap ) * log( qmLab ) )
+   + I( log( qmCap ) * log( qmMat ) ) + I( log( qmLab ) * log( qmMat ) ),
+   data = dat )
> summary( prodTLmSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 18 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	0.6388843	0.1319099	4.8433	1.277e-06	***
log(qmCap)	0.1308827	0.0993823	1.3170	0.187851	
log(qmLab)	0.7065388	0.1554775	4.5443	5.511e-06	***
log(qmMat)	0.4657241	0.1516106	3.0718	0.002127	**
I(0.5 * log(qmCap)^2)	0.0053261	0.1848845	0.0288	0.977018	
I(0.5 * log(qmLab)^2)	-1.5030423	0.6760718	-2.2232	0.026202	*
I(0.5 * log(qmMat)^2)	-0.5113584	0.3752125	-1.3629	0.172930	
I(log(qmCap) * log(qmLab))	0.4187528	0.2684617	1.5598	0.118802	
I(log(qmCap) * log(qmMat))	-0.4371558	0.1889843	-2.3132	0.020712	*
I(log(qmLab) * log(qmMat))	0.9800291	0.4201484	2.3326	0.019670	*
sigmaSq	0.9587291	0.1977464	4.8483	1.245e-06	***
gamma	0.9153382	0.0660990	13.8480	< 2.2e-16	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -128.0684

cross-sectional data

total number of observations = 140

```
mean efficiency: 0.5379943
```

```
> all.equal( coef( prodTLmSfa )[-c(1:4)], coef( prodTLSfa )[-c(1:4)],
+   check.attributes = FALSE )
```

```
[1] "Mean relative difference: 1.143686e-06"
```

```
> all.equal( efficiencies( prodTLmSfa ), efficiencies( prodTLSfa ) )
```

```
[1] "Mean relative difference: 8.581942e-07"
```

While the intercept and the first-order parameters have adjusted to the new units of measurement, the second-order parameters, the variance parameters, and the efficiency estimates remain (nearly) unchanged. From the estimated coefficients of the Translog production frontier with mean-scaled input quantities, we can immediately see that the monotonicity condition is fulfilled at the sample mean, that the output elasticities of capital, labor, and materials are 0.131, 0.707, and 0.466, respectively, at the sample mean, and that the elasticity of scale is $0.131 + 0.707 + 0.466 = 1.303$ at the sample mean.

6.2 Stochastic cost frontiers

6.2.1 Specification

The general specification of a stochastic cost frontier is

$$\ln c = \ln c(w, y) + u + v \quad \text{with} \quad u \geq 0, \quad (6.14)$$

where $u \geq 0$ accounts for cost inefficiency and v accounts for statistical noise. This model can be re-written as:

$$c = c(w, y) e^u e^v \quad (6.15)$$

The cost efficiency according to Shepard is

$$CE = \frac{c}{c(w, y) e^v} = \frac{f(x) e^u e^v}{c(w, y) e^v} = e^u, \quad (6.16)$$

while the cost efficiency according to Farrell is

$$CE = \frac{c(w, y) e^v}{c} = \frac{c(w, y) e^v}{f(x) e^u e^v} = e^{-u}. \quad (6.17)$$

Assuming a normal distribution of the noise term v and a positive half-normal distribution of the inefficiency term u , the distribution of the residuals from a cost function is expected to be right-skewed in the case of cost inefficiencies.

6.2.2 Skewness of residuals from OLS estimations

The following commands visualize the distribution of the residuals of the OLS estimations of the Cobb-Douglas and Translog cost functions with linear homogeneity in input prices imposed:

```
> hist( residuals( costCDHom ) )
> hist( residuals( costTLHom ) )
```

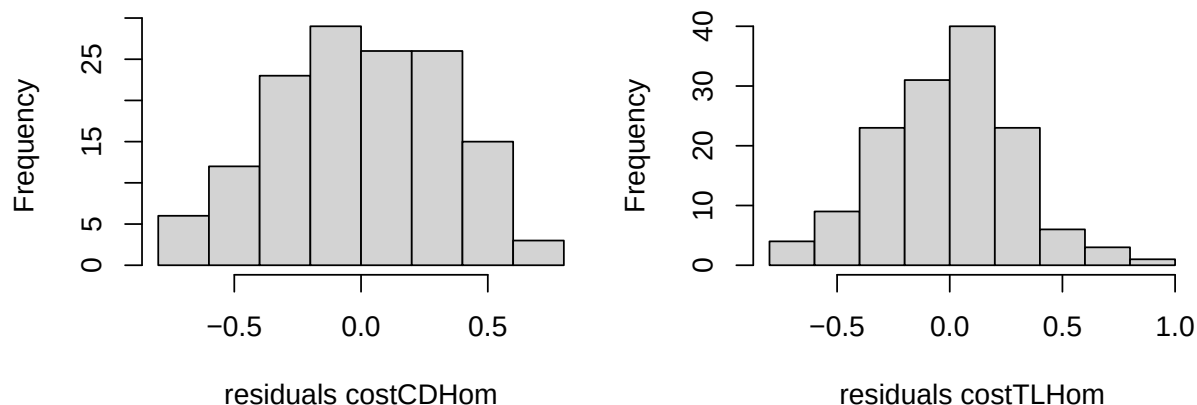


Figure 6.8: Residuals of Cobb-Douglas and Translog cost functions

The resulting graphs are shown in figure 6.8. The distributions of the residuals look approximately symmetric and rather a little left-skewed than right-skewed (although we expected the latter). This visual assessment of the skewness can be confirmed by calculating the skewness using the function `skewness` that is available in the package *moments*:

```
> library( "moments" )
> skewness( residuals( costCDHom ) )
```

```
[1] -0.05788105
```

```
> skewness( residuals( costTLHom ) )
```

```
[1] -0.03709506
```

The residuals of the two cost functions have both a small (in absolute terms) but negative skewness, which means that the residuals are slightly left-skewed, although we expected right-skewed residuals. It could be that the distribution of the unknown true total error term ($u + v$) in the sample is indeed symmetric or slightly left-skewed, e.g. because

- there is no cost inefficiency (but only noise) (the distribution of residuals is “correct”),
- the distribution of the noise term is left-skewed, which neutralizes the right-skewed distribution of the inefficiency term (misspecification of the distribution of the noise term in the SFA model),

- the distribution of the inefficiency term is symmetric or left-skewed (misspecification of the distribution of the inefficiency term in the SFA model),
- the sampling of the observations by coincidence resulted in a symmetric or left-skewed distribution of the true total error term ($u+v$) in this specific sample, although the distribution of the true total error term ($u+v$) in the population is right-skewed, and/or
- the farm managers do not aim at maximizing profit (which implies minimizing costs) but have other objectives.

It could also be that the distribution of the unknown true residuals in the sample is right-skewed, but the OLS estimates are left-skewed, e.g. because

- the parameter estimates are imprecise (but unbiased),
- the estimated functional forms (Cobb-Douglas and Translog) are poor approximations of the unknown true functional form (functional-form misspecification),
- there are further relevant explanatory variables that are not included in the model specification (omitted-variables bias),
- there are measurement errors in the variables, particularly in the explanatory variables (errors-in-variables problem), and/or
- the output quantity or the input prices are not exogenously given (endogeneity bias).

Hence, a left-skewed distribution of the residuals does not necessarily mean that there is no cost inefficiency, but it could also mean that the model is misspecified or that this is just by coincidence.

6.2.3 Estimation of a Cobb-Douglas stochastic cost frontier

The following command estimates a Cobb-Douglas stochastic cost frontier with linear homogeneity in input prices imposed:

```
> costCDHomSfa <- sfa( log( cost / pMat ) ~ log( pCap / pMat ) +
+   log( pLab / pMat ) + log( qOut ), data = dat,
+   ineffDecrease = FALSE )
> summary( costCDHomSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency increases the endogenous variable (as in a cost function)

The dependent variable is logged

Iterative ML estimation terminated after 55 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

Estimate	Std. Error	z value	Pr(> z)
----------	------------	---------	----------

```

(Intercept)    6.74924420 0.74118625  9.1060 < 2.2e-16 ***
log(pCap/pMat) 0.07241390 0.04565845  1.5860    0.1127
log(pLab/pMat) 0.44642025 0.07955197  5.6117 2.004e-08 ***
log(qOut)      0.37415329 0.03009634 12.4319 < 2.2e-16 ***
sigmaSq        0.11117595 0.01442194  7.7088 1.270e-14 ***
gamma          0.00018685 0.06501769  0.0029    0.9977
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -44.87812

```

```

cross-sectional data
total number of observations = 140

```

```
mean efficiency: 0.9963738
```

The parameter γ , which indicates the proportion of the total residual variance that is caused by inefficiency is close to zero and a t -test suggests that it is statistically not significantly different from zero. As the t -test for the parameter γ is not always reliable, we use a likelihood ratio test to verify this result:

```
> lrtest( costCDHomSfa )
```

```
Likelihood ratio test
```

```

Model 1: OLS (no inefficiency)
Model 2: Error Components Frontier (ECF)
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -44.878
2    6 -44.878  1     0    0.4991

```

This test confirms that the fit of the OLS model (which assumes that γ is zero and hence, that there is no inefficiency) is not significantly worse than the fit of the stochastic frontier model. In fact, the cost efficiency estimates are all very close to one. By default, the `efficiencies()` method calculates the efficiency estimates as $E[e^{-u}]$, which means that we obtain estimates of Farrell-type cost efficiencies (6.17). Given that $E[e^u]$ is not equal to $1/E[e^{-u}]$ (as the expectation operator is an additive operator), we cannot obtain estimates of Shepard-type cost efficiencies (6.16) by taking the inverse of the estimates of the Farrell-type cost efficiencies (6.17). However, we can obtain estimates of Shepard-type cost efficiencies (6.16) by setting argument `minusU` of the `efficiencies()` method equal to `FALSE`, which tells the `efficiencies()` method to calculate the efficiency estimates as $E[e^u]$.


```

> dat$costEffCDHomFarrell <- efficiencies( costCDHomSfa )
> dat$costEffCDHomShepard <- efficiencies( costCDHomSfa, minusU = FALSE )

> hist( dat$costEffCDHomFarrell, 15 )
> hist( dat$costEffCDHomShepard, 15 )

```

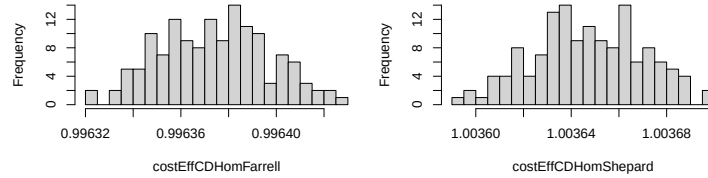


Figure 6.9: Efficiency estimates of Cobb-Douglas cost frontier

The resulting graphs are shown in figure 6.9. While the Farrell-type cost efficiencies are all slightly below one, the Shepard-type cost efficiencies are all slightly above one. Both graphs show that we do not find any relevant cost inefficiencies, although we have found considerable technical inefficiencies.

6.2.4 Short-run cost frontiers

```

> hist( residuals( costCDSRHom ) )

```

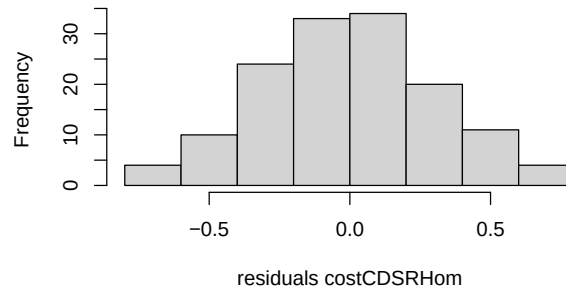


Figure 6.10: Residuals of Cobb-Douglas short-run cost function

The resulting graph is shown in figure 6.10.

```

> costCDSRHomSfa <- sfa( log( vCost / pMat ) ~ log( pLab / pMat ) +
+   log( qCap ) + log( qOut ), data = dat, ineffDecrease = FALSE )
> summary( costCDSRHomSfa )

```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency increases the endogenous variable (as in a cost function)

The dependent variable is logged

iteration failed

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	5.62206	1.00000	5.6221	1.887e-08 ***
log(pLab/pMat)	0.53487	1.00000	0.5349	0.5927
log(qCap)	0.18774	1.00000	0.1877	0.8511
log(qOut)	0.29010	1.00000	0.2901	0.7717
sigmaSq	0.10122	1.00000	0.1012	0.9194
gamma	0.05000	1.00000	0.0500	0.9601

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -36.05615

cross-sectional data

total number of observations = 140

mean efficiency: 0.9456755

```
> costCDSRHomSfa <- sfa( log( vCost / pMat ) ~ log( pLab / pMat ) +
+   log( qCap ) + log( qOut ), data = dat, ineffDecrease = FALSE,
+   startVal = c( coef( costCDSRHom ), 0.1, 0.01 ) )
> summary( costCDSRHomSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency increases the endogenous variable (as in a cost function)

The dependent variable is logged

Iterative ML estimation terminated after 45 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	5.6711615	0.3617424	15.6773	< 2.2e-16 ***
log(pLab/pMat)	0.5348738	0.0637620	8.3886	< 2.2e-16 ***
log(qCap)	0.1877450	0.0367325	5.1111	3.202e-07 ***
log(qOut)	0.2901018	0.0308632	9.3996	< 2.2e-16 ***
sigmaSq	0.0980563	0.0150503	6.5152	7.258e-11 ***
gamma	0.0009243	0.1326111	0.0070	0.9944

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -36.05537
```

```
cross-sectional data
total number of observations = 140
```

```
mean efficiency: 0.9924491
```

```
> dat$costEffCDSRHom <- efficiencies( costCDSRHomSfa )
```

```
> hist( dat$costEffCDSRHom, 15 )
```

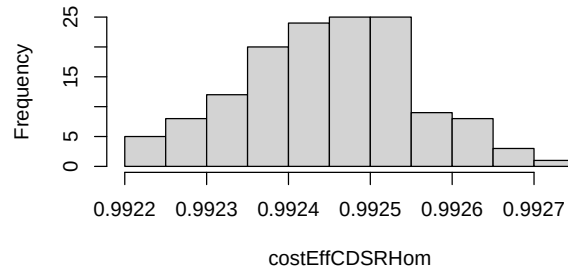


Figure 6.11: Efficiency estimates of Cobb-Douglas short-run cost frontier

The resulting graphs are shown in figure 6.11.

6.2.5 Profit frontiers

$$\ln \pi = \ln \pi(p, w) - u + v \quad \text{with} \quad u \geq 0, \quad (6.18)$$

where $-u \leq 0$ accounts for profit inefficiency and v accounts for statistical noise. This model can be re-written as:

$$\pi = \pi(p, w) e^{-u} e^v \quad (6.19)$$

Profit efficiency according to Farrell:

$$PE = \frac{\pi}{\pi(w, y) e^v} = \frac{\pi(w, y) e^{-u} e^v}{\pi(w, y) e^v} = e^{-u} \quad (6.20)$$

6.3 Analyzing the effects of z variables

In many empirical cases, the output quantity does not only depend on the input quantities but also on some other variables, e.g. the manager's education and experience and in agricultural production also the soil quality and rainfall. If these factors influence the production process, they must be included in applied production analyses in order to avoid an omitted-variables bias. Our data set on French apple producers includes the variable `adv`, which is a dummy variable

and indicates whether the apple producer uses an advisory service. In the following, we will apply different methods to figure out whether the production process differs between users and non-users of an advisory service.

6.3.1 Production functions with z variables

Additional factors that influence the production process (z) can be included as additional explanatory variables in the production function:

$$y = f(x, z). \quad (6.21)$$

This function can be used to analyze how the additional explanatory variables (z) affect the output quantity for given input quantities, i.e. how they affect the productivity.

In case of a Cobb-Douglas functional form, we get following extended production function:

$$\ln y = \alpha_0 + \sum_i \alpha_i \ln x_i + \alpha_z z \quad (6.22)$$

Based on this Cobb-Douglas production function and our data set on French apple producers, we can check whether the apple producers who use an advisory service produce a different output quantity than non-users with the same input quantities, i.e. whether the productivity differs between users and non-users. This extended production function can be estimated by following command:

```
> prodCDAdv <- lm( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ) + adv,
+   data = dat )
> summary( prodCDAdv )
```

Call:

```
lm(formula = log(qOut) ~ log(qCap) + log(qLab) + log(qMat) +
    adv, data = dat)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.7807	-0.3821	0.0022	0.4709	1.3323

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-2.33371	1.29590	-1.801	0.0740 .
log(qCap)	0.15673	0.08581	1.826	0.0700 .
log(qLab)	0.69225	0.15190	4.557	1.15e-05 ***
log(qMat)	0.62814	0.12379	5.074	1.26e-06 ***

```
adv          0.25896    0.10932    2.369    0.0193 *
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.6452 on 135 degrees of freedom
```

```
Multiple R-squared:  0.6105,      Adjusted R-squared:  0.599
```

```
F-statistic:  52.9 on 4 and 135 DF,  p-value: < 2.2e-16
```

The estimation result shows that users of an advisory service produce significantly more than non-users with the same input quantities. Given the Cobb-Douglas production function (6.22), the coefficient of an additional explanatory variable can be interpreted as the marginal effect on the relative change of the output quantity:

$$\alpha_z = \frac{\partial \ln y}{\partial z} = \frac{\partial \ln y}{\partial y} \frac{\partial y}{\partial z} = \frac{\partial y}{\partial z} \frac{1}{y} \quad (6.23)$$

Hence, our estimation result indicates that users of an advisory service produce approximately 25.9% more output than non-users with the same input quantity but the large standard error of this coefficient indicates that this estimate is rather imprecise. Given that the change of a dummy variable from zero to one is not marginal and that the coefficient of the variable `adv` is not close to zero, the above interpretation of this coefficient is a rather poor approximation. In fact, our estimation results suggest that the output quantity of apple producers with advisory service is on average $\exp(\alpha_z) = 1.296$ times as large as (29.6% larger than) the output quantity of apple producers without advisory service given the same input quantities. As users and non-users of an advisory service probably differ in some unobserved variables that affect the productivity (e.g. motivation and effort to increase productivity), the coefficient α_z is not necessarily the causal effect of the advisory service but describes the difference in productivity between users and non-users of the advisory service.

6.3.2 Production frontiers with z variables

A production function that includes additional factors that influence the production process (6.21) can also be estimated as a stochastic production frontier. In this specification, it is assumed that the additional explanatory variables influence the production frontier.

The following command estimates the extended Cobb-Douglas production function (6.22) using the stochastic frontier method:

```
> prodCDAAdvSfa <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ) + adv,
+   data = dat )
> summary( prodCDAAdvSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 14 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-0.247751	1.357917	-0.1824	0.8552301
log(qCap)	0.156906	0.081337	1.9291	0.0537222 .
log(qLab)	0.695977	0.148793	4.6775	2.904e-06 ***
log(qMat)	0.491840	0.139348	3.5296	0.0004162 ***
adv	0.150742	0.111233	1.3552	0.1753583
sigmaSq	0.916031	0.231604	3.9552	7.648e-05 ***
gamma	0.861029	0.114087	7.5471	4.450e-14 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -132.8679

cross-sectional data

total number of observations = 140

mean efficiency: 0.5545099

The estimation result still indicates that users of an advisory service have a higher productivity than non users, but the coefficient is smaller and no longer statistically significant. The result of the *t*-test is confirmed by a likelihood-ratio test:

```
> lrtest( prodCDSfa, prodCDAAdvSfa )
```

Likelihood ratio test

Model 1: prodCDSfa

Model 2: prodCDAAdvSfa

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	6	-133.89			
2	7	-132.87	1	2.0428	0.1529

The model with advisory service as additional explanatory variable indicates that there are significant inefficiencies (at 5% significance level):

```
> lrtest( prodCDAAdvSfa )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Error Components Frontier (ECF)

```
#Df  LogLik Df Chisq Pr(>Chisq)
1    6 -134.76
2    7 -132.87  1  3.78    0.02593 *
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The following commands compute the technical efficiency estimates and compare them to the efficiency estimates obtained from the Cobb-Douglas production frontier without advisory service as an explanatory variable:

```
> dat$effCDAdv <- efficiencies( prodCDAdvSfa )
> compPlot( dat$effCD[ dat$adv == 0 ],
+   dat$effCDAdv[ dat$adv == 0 ] )
> points( dat$effCD[ dat$adv == 1 ],
+   dat$effCDAdv[ dat$adv == 1 ], pch = 20 )
```

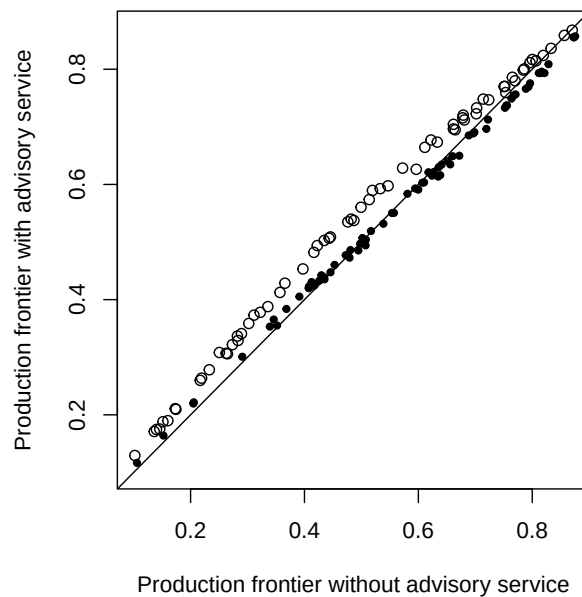


Figure 6.12: Technical efficiency estimates of Cobb-Douglas production frontier with and without advisory service as additional explanatory variable (circles = producers who do not use an advisory service, solid dots = producers who use an advisory service)

The resulting graph is shown in figure 6.12. It appears as if the non-users of an advisory service became somewhat more efficient. This is because the stochastic frontier model that includes

the advisory service as an explanatory variable has in fact two production frontiers: a lower frontier for the non-users of an advisory service and a higher frontier for the users of an advisory service. The coefficient of the dummy variable `adv`, i.e. α_{adv} , can be interpreted as a quick estimate of the difference between the two frontier functions. In our empirical case, the difference is approximately 15.1%. However, a precise calculation indicates that the frontier of the users of the advisory service is $\exp(\alpha_{adv}) = 1.163$ times (16.3% higher than) the frontier of the non-users of advisory service. And the frontier of the non-users of the advisory service is $\exp(-\alpha_{adv}) = 0.86$ times (14% lower than) the frontier of the users of advisory service. As the non-users of an advisory service are compared to a lower frontier now, they appear to be more efficient now. While it is reasonable to have different frontier functions for different soil types, it does not seem to be too reasonable to have different frontier functions for users and non-users of an advisory service, because there is no physical reasons, why users of an advisory service should have a maximum output quantity that is different from the maximum output quantity of non-users.

6.3.3 Efficiency effects production frontiers

As explained above, it does not seem to be too reasonable to have different frontier functions for users and non-users of an advisory service. However, it seems to be reasonable to assume that users of an advisory service have on average different efficiencies than non-users. A model that can account for this has been proposed by Battese and Coelli (1995). In this stochastic frontier model, the efficiency level might be affected by additional explanatory variables: The inefficiency term u follows a positive truncated normal distribution with constant scale parameter σ_u^2 and a location parameter μ that depends on additional explanatory variables:

$$u \sim N^+(\mu, \sigma_u^2) \quad \text{with} \quad \mu = \delta' z, \quad (6.24)$$

where δ is an additional parameter (vector) to be estimated. Function `sfa` can also estimate these “efficiency effects frontiers”. The additional variables that should explain the efficiency level must be specified at the end of the model formula, where a vertical bar separates them from the (regular) input variables:

```
> prodCDSfaAdvInt <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ) |
+   adv, data = dat )
> summary( prodCDSfaAdvInt )
```

Efficiency Effects Frontier (see Battese & Coelli 1995)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 19 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit


```

final maximum likelihood estimates
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.090700    1.235454 -0.0734  0.941476
log(qCap)    0.168623    0.081284  2.0745  0.038034 *
log(qLab)    0.653860    0.146054  4.4768 7.576e-06 ***
log(qMat)    0.513533    0.132236  3.8835  0.000103 ***
Z_(Intercept) -0.016812    1.255298 -0.0134  0.989314
Z_adv       -1.077590    1.053765 -1.0226  0.306492
sigmaSq      1.096521    0.789600  1.3887  0.164922
gamma        0.863095    0.099424  8.6809 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -130.516

```

```

cross-sectional data
total number of observations = 140

```

```
mean efficiency: 0.6004358
```

One can use the `lrtest()` method to test the statistical significance of the entire inefficiency model, i.e. the null hypothesis is $H_0: \gamma = 0$ and $\delta_j = 0 \forall j$:

```
> lrtest( prodCDSfaAdvInt )
```

```
Likelihood ratio test
```

```

Model 1: OLS (no inefficiency)
Model 2: Efficiency Effects Frontier (EEF)
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -137.61
2    8 -130.52  3 14.185    0.001123 **
---

```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The test indicates that the fit of this model is significantly better than the fit of the OLS model (without advisory service as explanatory variable).

The coefficient of the advisory service in the inefficiency model is negative but statistically insignificant. By default, an intercept is added to the inefficiency model but it is completely statistically insignificant. In many econometric estimations of the efficiency effects frontier model, the intercept of the inefficiency model (δ_0) is only weakly identified, because the values of δ_0 can

often be changed with only marginally reducing the log-likelihood value, if the slope parameters of the inefficiency model ($\delta_i, i \neq 0$) and the variance parameters (σ^2 and γ) are adjusted accordingly. This can be checked by taking a look at the correlation matrix of the estimated parameters:

```
> round( cov2cor( vcov( prodCDSfaAdvInt ) ), 2 )
```

	(Intercept)	log(qCap)	log(qLab)	log(qMat)	Z_(Intercept)	Z_adv
(Intercept)	1.00	-0.06	-0.50	-0.18	0.02	0.05
log(qCap)	-0.06	1.00	-0.37	-0.15	-0.15	-0.16
log(qLab)	-0.50	-0.37	1.00	-0.58	0.24	0.27
log(qMat)	-0.18	-0.15	-0.58	1.00	-0.12	-0.19
Z_(Intercept)	0.02	-0.15	0.24	-0.12	1.00	0.90
Z_adv	0.05	-0.16	0.27	-0.19	0.90	1.00
sigmaSq	0.09	0.12	-0.20	0.02	-0.95	-0.86
gamma	0.33	-0.01	0.00	-0.30	-0.59	-0.46
	sigmaSq gamma					
(Intercept)	0.09	0.33				
log(qCap)	0.12	-0.01				
log(qLab)	-0.20	0.00				
log(qMat)	0.02	-0.30				
Z_(Intercept)	-0.95	-0.59				
Z_adv	-0.86	-0.46				
sigmaSq	1.00	0.76				
gamma	0.76	1.00				

The estimate of the intercept of the inefficiency model (δ_0) is very highly correlated with the estimate of the (slope) coefficient of the advisory service in the inefficiency model (δ_1) and the estimate of the parameter σ^2 and it is considerably correlated with the estimate of the parameter γ .

The intercept can be suppressed by adding a “-1” to the specification of the inefficiency model:

```
> prodCDSfaAdv <- sfa( log( qOut ) ~ log( qCap ) + log( qLab ) + log( qMat ) |
+   adv - 1, data = dat )
> summary( prodCDSfaAdv )
```

Efficiency Effects Frontier (see Battese & Coelli 1995)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 14 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-0.090455	1.247496	-0.0725	0.94220
log(qCap)	0.168471	0.077008	2.1877	0.02869 *
log(qLab)	0.654341	0.139669	4.6849	2.800e-06 ***
log(qMat)	0.513291	0.130854	3.9226	8.759e-05 ***
Z_adv	-1.064859	0.545950	-1.9505	0.05112 .
sigmaSq	1.086417	0.255371	4.2543	2.097e-05 ***
gamma	0.862306	0.081468	10.5845	< 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -130.5161

cross-sectional data

total number of observations = 140

mean efficiency: 0.599406

A likelihood ratio test against the corresponding OLS model indicates that the fit of this SFA model is significantly better than the fit of the corresponding OLS model (without advisory service as explanatory variable):

```
> lrtest( prodCDSfaAdv )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Efficiency Effects Frontier (EEF)

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	5	-137.61			
2	7	-130.52	2	14.185	0.0002907 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

A likelihood ratio test confirms the *t*-test that the intercept in the inefficiency model is statistically insignificant:

```
> lrtest( prodCDSfaAdv, prodCDSfaAdvInt )
```

Likelihood ratio test

Model 1: prodCDSfaAdv

Model 2: prodCDSfaAdvInt

```
#Df  LogLik Df  Chisq Pr(>Chisq)
1    7 -130.52
2    8 -130.52  1 2e-04    0.9892
```

The coefficient of the advisory service in the inefficiency model is now significantly negative (at 10% significance level), which means that users of an advisory service have a significantly smaller inefficiency term u , i.e. are significantly more efficient. The size of the coefficients of the inefficiency model (δ) cannot be reasonably interpreted. However, if argument `margEff` of the `efficiencies` method is set to `TRUE`, this method does not only return the efficiency estimates but also the marginal effects of the variables that should explain the efficiency level on the efficiency estimates (see [Olsen and Henningsen, 2011](#)):

```
> dat$effCDAAdv2 <- efficiencies( prodCDSfaAdv, margEff = TRUE )
```

The marginal effects differ between observations and are available in the attribute `margEff`. The following command extracts and visualizes the marginal effects of the variable that indicates the use of an advisory service on the efficiency estimates:

```
> hist( attr( dat$effCDAAdv2, "margEff" ), 20 )
```

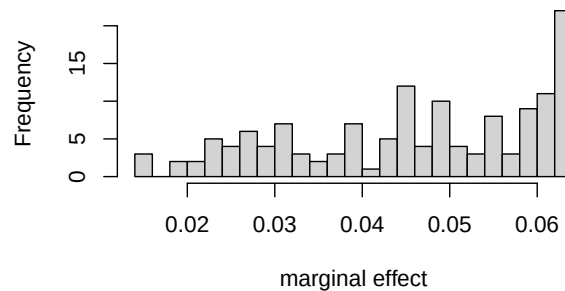


Figure 6.13: Marginal effects of the variable that indicates the use of an advisory service on the efficiency estimates

The resulting graph is shown in figure 6.13. It indicates that apple producers who use an advisory service are between 1.4 and 6.4 percentage points more efficient than apple producers who do not use an advisory service.

6.4 Decomposition of cost efficiency

In this section, we estimate a Cobb-Douglas production function and decompose cost efficiency into (cost) technical efficiency and (cost) allocative efficiency. The following code replicates an example in the textbook of [Coelli et al. \(2005, p. 273, Table 10.2\)](#):

6 Stochastic Frontier Analysis

```
> # estimating a Copbb-Douglas production frontier (ignoring "other inputs")
> riceCdSfaTime <- sfa( log(prod) ~ YEARDUM + log(area) + log(labor) + log(npk),
+   data = riceProdPhil )
> summary( riceCdSfaTime, extraPar = TRUE )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 10 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	0.2705295	0.0373920	7.2350	4.657e-13	***
YEARDUM	0.0148901	0.0067910	2.1926	0.028335	*
log(area)	0.3557513	0.0597575	5.9533	2.629e-09	***
log(labor)	0.3507348	0.0635010	5.5233	3.327e-08	***
log(npk)	0.2565320	0.0352279	7.2821	3.287e-13	***
sigmaSq	0.2418370	0.0262782	9.2030	< 2.2e-16	***
gamma	0.8979776	0.0310801	28.8924	< 2.2e-16	***
sigmaSqU	0.2171642	0.0291291	7.4552	8.970e-14	***
sigmaSqV	0.0246728	0.0060797	4.0582	4.945e-05	***
sigma	0.4917693	0.0267180	18.4059	< 2.2e-16	***
sigmaU	0.4660088	0.0312538	14.9105	< 2.2e-16	***
sigmaV	0.1570757	0.0193529	8.1164	4.802e-16	***
lambdaSq	8.8017709	2.9860105	2.9477	0.003202	**
lambda	2.9667779	0.5032413	5.8953	3.739e-09	***
varU	0.0789132	NA	NA	NA	
sdU	0.2809149	NA	NA	NA	
gammaVar	0.7618134	NA	NA	NA	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -83.76704

cross-sectional data

total number of observations = 344

mean efficiency: 0.7201092

```

> # extracting coefficients
> b0 <- coef( riceCdSfaTime )["(Intercept)"]
> theta <- coef( riceCdSfaTime )["YEARDUM"]
> b1 <- coef( riceCdSfaTime )["log(area)"]
> b2 <- coef( riceCdSfaTime )["log(labor)"]
> b3 <- coef( riceCdSfaTime )["log(npk)"]
> sig2 <- coef( riceCdSfaTime )["sigmaSq"]
> gamma <- coef( riceCdSfaTime )["gamma"]
> print( lambda <- unname( sqrt( gamma ) / sqrt( 1 - gamma ) ) )

[1] 2.966778

> print( sigma <- unname( sqrt( sig2 ) ) )

[1] 0.4917693

> # calculating technical efficiencies manually
> sig2v <- sig2 * ( 1 - gamma )
> sig2u <- sig2 * gamma
> mustar <- - residuals( riceCdSfaTime, asInData = TRUE ) * gamma
> sigstar <- sqrt( sig2u * sig2v / sig2 )
> musig <- mustar/sigstar
> uhati <- mustar + sigstar * dnorm(musig) / pnorm(musig)
> tei <- ( pnorm(musig-sigstar) / pnorm(musig) ) * exp( sigstar^2 / 2 - mustar )
> # calculating technical efficiencies with the frontier package
> tei2 <- efficiencies( riceCdSfaTime, asInData = TRUE )
> all.equal( tei, tei2 )

[1] TRUE

> # calculating cost efficiencies
> h2 <- log( riceProdPhil$sArea / riceProdPhil$sLabor ) - log(b1/b2)
> h3 <- log( riceProdPhil$sArea / riceProdPhil$sNpk ) - log(b1/b3)
> r <- b1 + b2 + b3
> ai <- (1/r) * (b2*h2 + b3*h3) + log(b1 + b2*exp(-h2) + b3*exp(-h3))
> # cost efficiency due to technical inefficiency
> ctei <- exp( -uhati / r )
> # cost efficiency due to allocative inefficiency
> caei = exp( log(r) - ai )
> # cost efficiency
> cei <- ctei * caei
> # collecting all efficiency measures in a data frame

```

```
> eff <- data.frame( tei, ctei, caei, cei )
> # efficiencies of first and last observations
> eff[ c(1:4, 339:344), ]
```

	tei	ctei	caei	cei
1	0.7516838	0.7361695	0.9304091	0.6849388
2	0.7404122	0.7245165	0.9494638	0.6879022
3	0.7848997	0.7707317	0.9343774	0.7201543
4	0.8670764	0.8577511	0.8378218	0.7186425
339	0.9362054	0.9321857	0.7034357	0.6557327
340	0.6608281	0.6431254	0.8376981	0.5387449
341	0.7711476	0.7563807	0.9158451	0.6927275
342	0.9260517	0.9212374	0.7758011	0.7146970
343	0.8261157	0.8141105	0.8493577	0.6914710
344	0.8980248	0.8910094	0.8012788	0.7139469

```
> # some summary statistics of efficiencies
> t(sapply(eff, function(x)
+   c( N=length(x), MEAN=mean(x), ST.DEV=sd(x), VARIANCE=var(x),
+     MINIMUM = min(x), MAXIMUM = max(x), COEF.OF.VAR=sd(x)/mean(x))))
```

	N	MEAN	ST.DEV	VARIANCE	MINIMUM	MAXIMUM	COEF.OF.VAR
tei	344	0.7201092	0.15320508	0.023471796	0.12701877	0.9574515	0.2127526
ctei	344	0.7059377	0.15662013	0.024529864	0.11600023	0.9550248	0.2218611
caei	344	0.8885442	0.07884834	0.006217061	0.60541374	0.9973617	0.0887388
cei	344	0.6221760	0.12735580	0.016219501	0.08423157	0.8335533	0.2046942

7 Data Envelopment Analysis (DEA)

7.1 Preparations

We load the R package “Benchmarking” in order to use it for Data Envelopment Analysis:

```
> library( "Benchmarking" )
```

We create a matrix of input quantities and a vector of output quantities:

```
> xMat <- cbind( dat$qCap, dat$qLab, dat$qMat )
> yVec <- dat$qOut
```

7.2 DEA with input-oriented efficiencies

The following command conducts an input-oriented DEA with VRS:

```
> deaVrsIn <- dea( xMat, yVec )
> hist( eff( deaVrsIn ) )
```

Display the “peers” of the first 14 observations:

```
> peers( deaVrsIn )[ 1:14, ]
```

	peer1	peer2	peer3	peer4
[1,]	44	73	80	135
[2,]	80	100	126	NA
[3,]	44	54	73	100
[4,]	4	NA	NA	NA
[5,]	17	54	81	NA
[6,]	41	73	126	132
[7,]	7	NA	NA	NA
[8,]	44	54	80	83
[9,]	100	126	132	NA
[10,]	38	73	80	135
[11,]	54	81	100	NA
[12,]	44	54	81	100
[13,]	38	73	80	135
[14,]	44	54	81	100

7 Data Envelopment Analysis (DEA)

Display the λ s of the first 14 observations:

```
> lambda( deaVrsIn )[ 1:14, ]
```

	L4	L7	L17	L19	L38	L41	L44	L54	L61	L64
[1,]	0	0	0.00000000	0	0.00000000	0.00000000	0.08707089	0.00000000	0	0
[2,]	0	0	0.00000000	0	0.00000000	0.00000000	0.00000000	0.00000000	0	0
[3,]	0	0	0.00000000	0	0.00000000	0.00000000	0.05466873	0.34157362	0	0
[4,]	1	0	0.00000000	0	0.00000000	0.00000000	0.00000000	0.00000000	0	0
[5,]	0	0	0.07874218	0	0.00000000	0.00000000	0.00000000	0.62716635	0	0
[6,]	0	0	0.00000000	0	0.00000000	0.9520817	0.00000000	0.00000000	0	0
[7,]	0	1	0.00000000	0	0.00000000	0.00000000	0.00000000	0.00000000	0	0
[8,]	0	0	0.00000000	0	0.00000000	0.00000000	0.39228600	0.34818591	0	0
[9,]	0	0	0.00000000	0	0.00000000	0.00000000	0.00000000	0.00000000	0	0
[10,]	0	0	0.00000000	0	0.06541405	0.00000000	0.00000000	0.00000000	0	0
[11,]	0	0	0.00000000	0	0.00000000	0.00000000	0.00000000	0.52820862	0	0
[12,]	0	0	0.00000000	0	0.00000000	0.00000000	0.44076458	0.09749327	0	0
[13,]	0	0	0.00000000	0	0.01725343	0.00000000	0.00000000	0.00000000	0	0
[14,]	0	0	0.00000000	0	0.00000000	0.00000000	0.35937585	0.44329381	0	0

	L73	L74	L78	L80	L81	L83	L100	L103
[1,]	0.243735897	0	0	0.6423537	0.00000000	0.00000000	0.00000000	0
[2,]	0.000000000	0	0	0.5147430	0.00000000	0.00000000	0.3620871	0
[3,]	0.153372277	0	0	0.00000000	0.00000000	0.00000000	0.4503854	0
[4,]	0.000000000	0	0	0.00000000	0.00000000	0.00000000	0.00000000	0
[5,]	0.000000000	0	0	0.00000000	0.29409147	0.00000000	0.00000000	0
[6,]	0.002769034	0	0	0.00000000	0.00000000	0.00000000	0.00000000	0
[7,]	0.000000000	0	0	0.00000000	0.00000000	0.00000000	0.00000000	0
[8,]	0.000000000	0	0	0.2101886	0.00000000	0.04933947	0.00000000	0
[9,]	0.000000000	0	0	0.00000000	0.00000000	0.00000000	0.6917918	0
[10,]	0.068686498	0	0	0.2825911	0.00000000	0.00000000	0.00000000	0
[11,]	0.000000000	0	0	0.00000000	0.25455055	0.00000000	0.2172408	0
[12,]	0.000000000	0	0	0.00000000	0.29388540	0.00000000	0.1678567	0
[13,]	0.383969646	0	0	0.5669254	0.00000000	0.00000000	0.00000000	0
[14,]	0.000000000	0	0	0.00000000	0.04033289	0.00000000	0.1569974	0

	L126	L129	L132	L135	L137
[1,]	0.000000000	0	0.00000000	0.02683954	0
[2,]	0.123169836	0	0.00000000	0.00000000	0
[3,]	0.000000000	0	0.00000000	0.00000000	0
[4,]	0.000000000	0	0.00000000	0.00000000	0
[5,]	0.000000000	0	0.00000000	0.00000000	0
[6,]	0.008468157	0	0.03668108	0.00000000	0

```
[7,] 0.000000000 0 0.00000000 0.00000000 0
[8,] 0.000000000 0 0.00000000 0.00000000 0
[9,] 0.249102366 0 0.05910586 0.00000000 0
[10,] 0.000000000 0 0.00000000 0.58330837 0
[11,] 0.000000000 0 0.00000000 0.00000000 0
[12,] 0.000000000 0 0.00000000 0.00000000 0
[13,] 0.000000000 0 0.00000000 0.03185153 0
[14,] 0.000000000 0 0.00000000 0.00000000 0
```

The following commands display the “slack” of the first 14 observations in an input-oriented DEA with VRS:

```
> deaVrsIn <- dea( xMat, yVec, SLACK = TRUE )
> table( deaVrsIn$slack )
```

```
FALSE  TRUE
    78    62
```

```
> deaVrsIn$sx[ 1:14, ]
```

	sx1	sx2	sx3
[1,]	0	0	0.00000
[2,]	0	0	345.70719
[3,]	0	0	0.00000
[4,]	0	0	0.00000
[5,]	0	0	38.54949
[6,]	0	0	0.00000
[7,]	0	0	0.00000
[8,]	0	0	0.00000
[9,]	0	0	1624.33417
[10,]	0	0	0.00000
[11,]	0	0	12993.07250
[12,]	0	0	0.00000
[13,]	0	0	0.00000
[14,]	0	0	0.00000

```
> deaVrsIn$sy[ 1:14, ]
```

```
[1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

The following command conducts an input-oriented DEA with CRS:

```
> deaCrsIn <- dea( xMat, yVec, RTS = "crs" )
> hist( eff( deaCrsIn ) )
```

We can calculate the scale efficiencies by:

```
> se <- eff( deaCrsIn ) / eff( deaVrsIn )
> hist( se )
```

The following command conducts an input-oriented DEA with DRS

```
> deaDrsIn <- dea( xMat, yVec, RTS = "drs" )
> hist( eff( deaDrsIn ) )
```

And we check if firms are too small or too large. This is the number of observations that produce at the scale below the optimal scale size:

```
> sum( eff( deaVrsIn ) - eff( deaDrsIn ) > 1e-4 )

[1] 117
```

7.3 DEA with output-oriented efficiencies

The following command conducts an output-oriented DEA with VRS:

```
> deaVrsOut <- dea( xMat, yVec, ORIENTATION = "out" )
> hist( efficiencies( deaVrsOut ) )
```

The following command conducts an output-oriented DEA with CRS:

```
> deaCrsOut <- dea( xMat, yVec, RTS = "crs", ORIENTATION = "out" )
> hist( eff( deaCrsOut ) )
```

In case of CRS, input-oriented efficiencies are equivalent to output-oriented efficiencies:

```
> all.equal( eff( deaCrsIn ), 1 / eff( deaCrsOut ) )

[1] TRUE
```

7.4 DEA with “super efficiencies”

The following command obtains “super efficiencies” for an input-oriented DEA with CRS:

```
> sdeaVrsIn <- sdea( xMat, yVec )
> hist( eff( sdeaVrsIn ) )
```

7.5 DEA with graph hyperbolic efficiencies

The following command conducts a DEA with graph hyperbolic efficiencies and VRS:

```
> deaVrsGraph <- dea( xMat, yVec, ORIENTATION = "graph" )
> hist( eff( deaVrsGraph ) )
> plot( eff( deaVrsIn ), eff( deaVrsGraph ) )
> abline(0,1)
```

8 Distance Functions

8.1 Theory

8.1.1 Output distance functions

The Shepard output distance function is defined as:

$$D^o(x, y) = \min\{\lambda > 0 | y/\lambda \in P(x)\} \quad (8.1)$$

or (equivalently) as:

$$D^o(x, y) = \min\{\lambda > 0 | (x, y/\lambda) \in T\}, \quad (8.2)$$

where x is a vector of input quantities, y is a vector of output quantities, $P(x)$ is the production possibility set, and T is the technology set. The Shepard output distance function, $D^o(x, y)$, defined in (8.1) and (8.2) returns the Shepard output-oriented technical efficiencies defined in (5.1) and (5.5). Thus, it returns a value of one for fully efficient sets of inputs and outputs (x, y) , whereas it returns a non-negative value smaller than one for inefficient sets of inputs and outputs (x, y) .¹

8.1.1.1 Properties

It is usually assumed that the Shepard output distance function $D^o(x, y)$ given in (8.1) fulfills the following properties (see, e.g., Färe and Primont, 1995; Coelli et al., 2005):

1. $D^o(x, 0) = 0$ for all non-negative x
2. $D^o(x, y)$ is non-increasing in x , i.e. $\partial D^o(x, y)/\partial x_i \leq 0 \forall i = 1, \dots, N$
3. $D^o(x, y)$ is non-decreasing in y , i.e. $\partial D^o(x, y)/\partial y_i \geq 0 \forall i = 1, \dots, M$
4. $D^o(x, y)$ is linearly homogeneous in y , i.e. $D^o(x, ky) = k D^o(x, y) \forall k > 0$
5. $D^o(x, y)$ is quasiconvex in $x \Rightarrow$ convex isoquants²

¹The Farrell output distance function is defined as the inverse of the Shepard output distance function: $D^{oF}(x, y) = \max\{\lambda > 0 | \lambda y \in P(x)\} = 1/D^o(x, y)$. It returns the Farrell output-oriented technical efficiencies defined in (5.2) and (5.6).

²O'Donnell and Coelli (2005) clarified a typographical error in Färe and Primont (1995, p. 152) who state by mistake that the output distance function is quasiconcave in input quantities x . The quasiconcavity of the production function is equivalent to the quasiconvexity of the output distance function, whereas the 'conversion' between quasiconcavity and quasiconvexity is caused by the opposite 'direction' regarding the input quantities

6. $D^o(x, y)$ is convex in $y \Rightarrow$ concave transformation curves (i.e., joint production is advantageous compared to specialized production for a given set of input quantities)
7. $D^o(x, y) \leq 1$ indicates that y belongs to the production possibility set of x , i.e. $y \in P(x)$, while $D^o(x, y) > 1$ indicates that y does not belong to the production possibility set of x .
8. $D^o(x, y) = 1$ indicates that y is at the boundary of the production possibility set $P(x)$, i.e. y lies on the transformation curve (given x) and x lies on the isoquant (given y).

We will illustrate these properties using a very simple output distance function with two inputs $x = (x_1, x_2)'$ and two outputs $y = (y_1, y_2)'$:³

$$D^o(x, y) = \sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4} \quad (8.3)$$

In the following, we illustrate that this simple output distance function fulfills the above-mentioned properties 1–4:

- no output

$$D^o(x, 0) = \sqrt{0^2 + 0^2} x_1^{-0.4} x_2^{-0.4} = 0 \quad (8.4)$$

- monotonically non-increasing in x

$$\frac{\partial D^o(x, y)}{\partial x_i} = \frac{-0.4 \sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4}}{x_i} \leq 0; i \in \{1, 2\} \quad (8.5)$$

- monotonically non-decreasing in y

$$\frac{\partial D^o(x, y)}{\partial y_i} = \frac{x_1^{-0.4} x_2^{-0.4} y_i}{\sqrt{y_1^2 + y_2^2}} \geq 0; i \in \{1, 2\} \quad (8.6)$$

- linear homogeneous in y

$$D^o(x, ky) = \sqrt{(ky_1)^2 + (ky_2)^2} x_1^{-0.4} x_2^{-0.4} \quad (8.7)$$

$$= \sqrt{k^2 y_1^2 + k^2 y_2^2} x_1^{-0.4} x_2^{-0.4} \quad (8.8)$$

$$= \sqrt{k^2 (y_1^2 + y_2^2)} x_1^{-0.4} x_2^{-0.4} \quad (8.9)$$

$$= \sqrt{k^2} \sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4} \quad (8.10)$$

$$= k \sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4} \quad (8.11)$$

$$= k D^o(x, y) \quad (8.12)$$

(i.e., the production function is increasing in input quantities, while the output distance function is decreasing in input quantities).

³This output distance function is a very much simplified version of the multiple-output “ray frontier production function” suggested by Löthgren (1997, 2000).

- quasiconvexity in x

A sufficient condition for quasiconvexity is that all leading principal minors of the bordered Hessian matrix are strictly negative (see section 1.5.6). The first derivatives of the output distance function (8.3) with respect to the input quantities are given in (8.5). The remaining elements of the bordered Hessian matrix, i.e. the second derivatives with respect to the input quantities, are:

$$\frac{\partial^2 D^o(x, y)}{\partial x_i \partial x_j} = \frac{0.4(\Delta_{ij} + 0.4)\sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4}}{x_i x_j}; i \in \{1, 2\} \quad (8.13)$$

$$\Delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (8.14)$$

The first leading principal minor of the bordered Hessian B is:

$$|B_1| = - \left(\frac{\partial D^o(x, y)}{\partial x_1} \right)^2 \quad (8.15)$$

$$= - \left(-0.4\sqrt{y_1^2 + y_2^2} x_1^{-1.4} x_2^{-0.4} \right)^2 \quad (8.16)$$

$$= -0.4^2 \left(y_1^2 + y_2^2 \right) x_1^{-2.8} x_2^{-0.8} \quad (8.17)$$

$$< 0 \quad \forall x_1, x_2 > 0; y_1, y_2 \geq 0; y_1 + y_2 > 0 \quad (8.18)$$

The determinant of the (3×3) bordered Hessian is:

$$|B| = 2 \frac{\partial D^o(x, y)}{\partial x_1} \frac{\partial D^o(x, y)}{\partial x_2} \frac{\partial^2 D^o(x, y)}{\partial x_1 \partial x_2} \quad (8.19)$$

$$\begin{aligned} & - \left(\frac{\partial D^o(x, y)}{\partial x_1} \right)^2 \frac{\partial^2 D^o(x, y)}{\partial x_2^2} - \left(\frac{\partial D^o(x, y)}{\partial x_2} \right)^2 \frac{\partial^2 D^o(x, y)}{\partial x_1^2} \\ & = 2 \left(-0.4\sqrt{y_1^2 + y_2^2} x_1^{-1.4} x_2^{-0.4} \right) \left(-0.4\sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-1.4} \right) \\ & \quad \left(0.4^2 \sqrt{y_1^2 + y_2^2} x_1^{-1.4} x_2^{-1.4} \right) \end{aligned} \quad (8.20)$$

$$\begin{aligned} & - \left(-0.4\sqrt{y_1^2 + y_2^2} x_1^{-1.4} x_2^{-0.4} \right)^2 \left(0.4 \cdot 1.4 \sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-2.4} \right) \\ & - \left(-0.4\sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-1.4} \right)^2 \left(0.4 \cdot 1.4 \sqrt{y_1^2 + y_2^2} x_1^{-2.4} x_2^{-0.4} \right) \\ & = \left(0.4\sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4} \right)^3 \end{aligned} \quad (8.21)$$

$$\begin{aligned} & \left[2 \left(-x_1^{-1} \right) \left(-x_2^{-1} \right) \left(0.4 x_1^{-1} x_2^{-1} \right) - \left(-x_1^{-1} \right)^2 \left(1.4 x_2^{-2} \right) - \left(-x_2^{-1} \right)^2 \left(1.4 x_1^{-2} \right) \right] \\ & = \left(0.4\sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4} \right)^3 \left[0.8 x_1^{-2} x_2^{-2} - 1.4 x_1^{-2} x_2^{-2} - 1.4 x_1^{-2} x_2^{-2} \right] \end{aligned} \quad (8.22)$$

$$= -2 \left(0.4 \sqrt{y_1^2 + y_2^2} x_1^{-0.4} x_2^{-0.4} \right)^3 x_1^{-2} x_2^{-2} \quad (8.23)$$

$$= -2 \cdot 0.4^3 \left(\sqrt{y_1^2 + y_2^2} \right)^3 x_1^{-3.2} x_2^{-3.2} \quad (8.24)$$

$$< 0 \quad \forall x_1, x_2 > 0; y_1, y_2 \geq 0; y_1 + y_2 > 0 \quad (8.25)$$

If both of the input quantities are strictly positive and at least one of the output quantities is strictly positive, the first leading principal minor and the determinant of the bordered Hessian matrix are both strictly negative, which means that both the necessary condition and the sufficient condition for quasiconvexity are fulfilled.

Figure 8.1 illustrates the relationship between the two input quantities x_1 and x_2 and the output distance measure $D^o(x, y)$ holding the output quantities y_1 and y_2 constant so that $\sqrt{y_1^2 + y_2^2} = 1$, e.g. $y_1 = y_2 = \sqrt{0.5}$ (please note that the origin, i.e. $x_1 = x_2 = 0$, is in the back). The grey part of the surface indicates distance measures that are larger than one ($D^o(x, y) > 1$), i.e. combinations of the input quantities x_1 and x_2 that are too small to produce the output quantities y_1 and y_2 . The dark green line is an isoquant that indicates all combinations of the input quantities x_1 and x_2 that a fully efficient firm (i.e. $D^o(x, y) = 1$) needs to produce the output quantities y_1 and y_2 . Given that the output distance function is quasiconvex in x , the lower contour sets, i.e. the sets of input combinations that give an output distance measure ($D^o(x, y)$) smaller than a certain value (e.g. 1), are convex sets (see section 1.5.6). As the origin ($x_1 = x_2 = 0$) is not inside the lower contour sets, the quasiconvexity of the output distance function in x results in isoquants that are convex to the origin. The light green part of the surface indicates distance measures ($D^o(x, y)$) that are slightly smaller than one, indicating small technical inefficiency. The red part of the surface indicates distance measures ($D^o(x, y)$) that are considerably smaller than one, indicating considerable technical inefficiency.

- convexity in y

Convexity requires that the Hessian matrix is positive semidefinite (see section 1.5.5). The elements of the Hessian matrix, i.e. the second derivatives with respect to the output quantities, are:

$$\frac{\partial^2 D^o(x, y)}{\partial y_i \partial y_j} = \frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \left(\Delta_{ij} - \frac{y_i y_j}{y_1^2 + y_2^2} \right); i, j \in \{1, 2\} \quad (8.26)$$

$$\Delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (8.27)$$

Equation (8.26) shows that all diagonal elements of the Hessian matrix (i.e. $i = j$) are non-negative, because $1 - y_i^2/(y_1^2 + y_2^2)$ is non-negative for $i \in \{1, 2\}$.

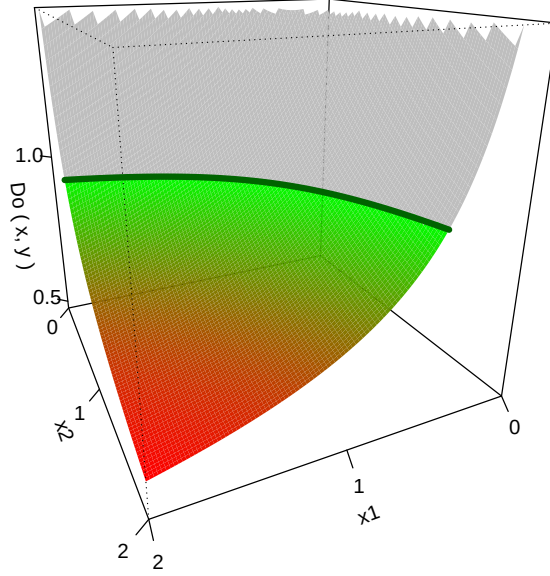


Figure 8.1: Output distance function for different input quantities

The determinant of the (2×2) Hessian matrix is:

$$|H| = \frac{\partial^2 D^i(x, y)}{\partial x_1^2} \cdot \frac{\partial^2 D^i(x, y)}{\partial x_2^2} - \left(\frac{\partial^2 D^i(x, y)}{\partial x_1 \partial x_2} \right)^2 \quad (8.28)$$

$$\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \left(1 - \frac{y_1^2}{y_1^2 + y_2^2} \right) \frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \left(1 - \frac{y_2^2}{y_1^2 + y_2^2} \right) - \left(\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \frac{y_1 y_2}{y_1^2 + y_2^2} \right)^2 \quad (8.29)$$

$$= \left(\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \right)^2 \left(1 - \frac{y_1^2}{y_1^2 + y_2^2} \right) \left(1 - \frac{y_2^2}{y_1^2 + y_2^2} \right) - \left(\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \right)^2 \left(\frac{y_1 y_2}{y_1^2 + y_2^2} \right)^2 \quad (8.30)$$

$$= \left(\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \right)^2 \left[\left(1 - \frac{y_1^2}{y_1^2 + y_2^2} \right) \left(1 - \frac{y_2^2}{y_1^2 + y_2^2} \right) - \left(\frac{y_1 y_2}{y_1^2 + y_2^2} \right)^2 \right] \quad (8.31)$$

$$= \left(\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \right)^2 \left[\frac{y_2^2}{y_1^2 + y_2^2} \frac{y_1^2}{y_1^2 + y_2^2} - \frac{y_1^2 y_2^2}{(y_1^2 + y_2^2)^2} \right] \quad (8.32)$$

$$= \left(\frac{x_1^{-0.4} x_2^{-0.4}}{\sqrt{y_1^2 + y_2^2}} \right)^2 \left[\frac{y_2^2 y_1^2}{(y_1^2 + y_2^2)^2} - \frac{y_1^2 y_2^2}{(y_1^2 + y_2^2)^2} \right] = 0 \geq 0 \quad (8.33)$$

As all three principal minors (the two diagonal elements and the determinant) of the Hessian matrix are non-negative, we can conclude that the Hessian matrix is positive semidefinite and, thus, the output distance function 8.3 is convex in y .

It is indeed not surprising that the determinant of the Hessian matrix is exactly zero, because the output distance function is linearly homogeneous in y and the determinants of linearly homogeneous functions are always zero.

Figure 8.2 illustrates the relationship between the two output quantities y_1 and y_2 and the output distance measure $D^o(x, y)$ holding the input quantities x_1 and x_2 constant so that $x_1^{0.4}x_2^{0.4} = 1$, e.g. $x_1 = x_2 = 1$ (please note that the origin, i.e. $y_1 = y_2 = 0$, is in the front). The grey part of the surface indicates distance measures that are larger than one ($D^o(x, y) > 1$), i.e. combinations of the output quantities y_1 and y_2 that cannot be produced by the input quantities x_1 and x_2 . The dark green line is a transformation curve that indicates all combinations of the output quantities y_1 and y_2 that a fully efficient firm (i.e. $D^o(x, y) = 1$) can produce from the input quantities x_1 and x_2 . Given that the output distance function is convex in y , it is also quasiconvex in y so that the lower contour sets, i.e. the sets of output combinations that give an output distance measure ($D^o(x, y)$) smaller than a certain value (e.g. 1), are convex sets (see section 1.5.6). As the origin ($y_1 = y_2 = 0$) is inside the lower contour sets, the convexity of the output distance function in y results in transformation curves that are concave to the origin. The light green part of the surface indicates distance measures ($D^o(x, y)$) that are slightly smaller than one, indicating small technical inefficiency. The red part of the surface indicates distance measures ($D^o(x, y)$) that are considerably smaller than one, indicating considerable technical inefficiency.

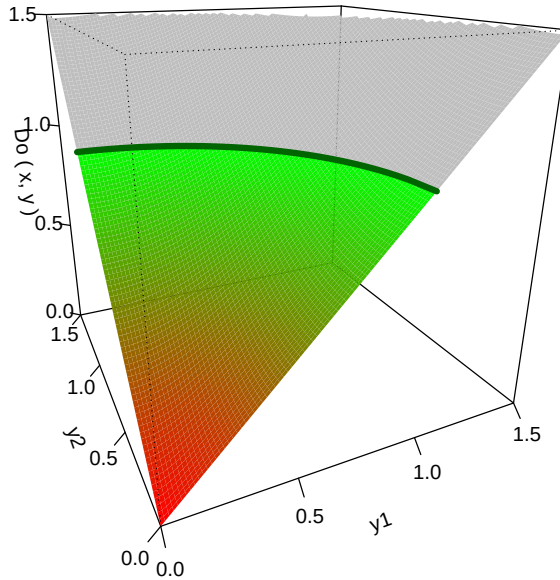


Figure 8.2: Output distance function for different output quantities

The following code generates figures 8.1 and 8.2:

```
> nVal <- 100
> colRedGreen <- colorRampPalette( c( "red", "green" ) )( 100 )
```

```

> x1 <- seq( 0.2, 2, length.out = nVal )
> x2 <- seq( 0.2, 2, length.out = nVal )
> dxMat <- outer( x1, x2, function( a, b ) a^(-0.4 ) * b^(-0.4 ) )
> dxFacet <- ( dxMat[-1, -1] + dxMat[-1, -nVal] + dxMat[-nVal, -1] +
+           dxMat[-nVal, -nVal] ) / 4
> dxInt <- cut( dxFacet,
+   seq( min( dxMat ), 1, length.out = length( colRedGreen ) ) )
> dxCol <- colRedGreen[ dxInt ]
> dxCol[ dxFacet > 1 ] <- "grey"
> dxMat[ dxMat > dxMat[1,nVal] ] <- NA

> ppx <- persp( x1, x2, dxMat, xlim = c( 0, max( x1 ) ), ylim = c( 0, max( x2 ) ),
+   zlim = c( min( dxMat, na.rm = TRUE ) - 0.1, max( dxMat, na.rm = TRUE ) ),
+   zlab = "Do ( x, y )", ticktype = "detailed", nticks = 3,
+   theta = 155, phi = 20, col = dxCol, border = NA )
> xa <- seq( 1/max(x2), max(x1), length.out = nVal )
> lines( trans3d( xa, 1/xa, 1, ppx ), col = "darkgreen", lwd = 5 )

> y1 <- seq( 0, 1.5, length.out = nVal )
> y2 <- seq( 0, 1.5, length.out = nVal )
> dyMat <- outer( y1, y2, function( a, b ) sqrt( a^2 + b^2 ) )
> dyFacet <- ( dyMat[-1, -1] + dyMat[-1, -nVal] + dyMat[-nVal, -1] +
+           dyMat[-nVal, -nVal] ) / 4
> dyInt <- cut( dyFacet,
+   seq( min( dyMat ), 1, length.out = length( colRedGreen ) ) )
> dyCol <- colRedGreen[ dyInt ]
> dyCol[ dyFacet > 1 ] <- "grey"
> dyMat[ dyMat > dyMat[ 1, nVal ] ] <- NA

> ppy <- persp( y1, y2, dyMat, zlab = "Do ( x, y )",
+   ticktype = "detailed", nticks = 3,
+   theta = -25, phi = 20, col = dyCol, border = NA )
> ya <- seq( 0, 1, length.out = nVal )
> lines( trans3d( ya, sqrt(1-ya^2), 1, ppy ), col = "darkgreen", lwd = 5 )

```

8.1.1.2 Distance elasticities

A distance elasticity of the output distance function $D^o(x, y)$ with respect to an input quantity x_i indicates the percentage change in the output distance measure $D^o(x, y)$ (and, hence, in the firm's efficiency) given a one percent increase in this input quantity. The distance elasticity of the i th

input is defined as:

$$\epsilon_i^I = \frac{\partial D^o(x, y)}{\partial x_i} \frac{x_i}{D^o} = \frac{\partial \ln D^o(x, y)}{\partial \ln x_i} \leq 0 \quad (8.34)$$

The linear homogeneity of the output distance function $D^o(x, y)$ in output quantities y allows an alternative interpretation of the distance elasticities of the inputs. The interpretation above equation (8.34) assumes that the remaining input quantities and all output quantities remain unchanged, while the output distance measure $D^o(x, y)$ and, thus, the technical efficiency adjusts to the change in one input quantity. Alternatively, we can assume the remaining input quantities and the output distance measure $D^o(x, y)$ and, thus, the technical efficiency remain unchanged, while the output quantities adjust to the change in one input quantity. Under this assumption, if the quantity of the i th input increases by one percent, all output quantities can simultaneously increase by $-\epsilon_i^I\%$ if the remaining input quantities and the technical efficiency remain unchanged. This interpretation includes the case, where the technical efficiency remains constant at one so that the change in the output quantities refers to the ‘frontier’, i.e., the maximum obtainable output quantities.

A distance elasticity of the output distance function $D^o(x, y)$ with respect to one output quantity y_i indicates the percentage change in the output distance measure $D^o(x, y)$ (hence, in the firm’s efficiency) given a one percent increase in this output quantity. The distance elasticity of the i th output is defined as:

$$\epsilon_i^O = \frac{\partial D^o(x, y)}{\partial y_i} \frac{y_i}{D^o} = \frac{\partial \ln D^o(x, y)}{\partial \ln y_i} \geq 0 \quad (8.35)$$

Linear homogeneity of the output distance function $D^o(x, y)$ in output quantities implies that the distance elasticities with respect to the output quantities sum to one:

$$\sum_{i=1}^M \epsilon_i^O = 1 \quad (8.36)$$

8.1.1.3 Elasticity of scale

The elasticity of scale of an output distance function $D^o(x, y)$ equals the negative sum of the distance elasticities with respect to the input quantities:

$$\epsilon = - \sum_i \epsilon_i^I \quad (8.37)$$

8.1.1.4 Optimising behaviour and distance elasticities

A necessary condition for revenue maximisation and profit maximisation is that the slope of the transformation curve is equal to the inverse of the negative output price ratio:

$$\frac{\partial y_i}{\partial y_j} = - \frac{p_j}{p_i}, \quad (8.38)$$

where $p_i; i = 1, \dots, M$ is the price of the i th output.

Equation (8.38) can be re-arranged in the following way:

$$-\frac{\partial y_i}{\partial y_j} = \frac{\frac{\partial D^o(x, y)}{\partial y_j}}{\frac{\partial D^o(x, y)}{\partial y_i}} = \frac{p_j}{p_i} \quad (8.39)$$

$$\frac{\frac{\partial D^o(x, y)}{\partial y_j} \frac{y_j}{D^o(x, y)}}{\frac{\partial D^o(x, y)}{\partial y_i} \frac{y_i}{D^o(x, y)}} = \frac{p_j y_i}{p_i y_i} \quad (8.40)$$

$$\frac{\epsilon_j^O}{\epsilon_i^O} = \frac{p_j y_j}{p_i y_i} \quad (8.41)$$

$$p_j y_j = \frac{\epsilon_j^O}{\epsilon_i^O} p_i y_i \quad (8.42)$$

$$\sum_{j=1}^M p_j y_j = \sum_{j=1}^M \frac{\epsilon_j^O}{\epsilon_i^O} p_i y_i \quad (8.43)$$

$$\sum_{j=1}^M p_j y_j = \frac{p_i y_i}{\epsilon_i^O} \sum_{j=1}^M \epsilon_j^O \quad (8.44)$$

$$\sum_{j=1}^M p_j y_j = \frac{p_i y_i}{\epsilon_i^O} \quad (8.45)$$

$$\epsilon_i^O = \frac{p_i y_i}{\sum_{j=1}^M p_j y_j}, \quad (8.46)$$

where we use equation (8.36), which is based on the linear homogeneity of the output distance function in output quantities. Hence, in case of revenue maximisation or profit maximisation, the distance elasticities of the outputs are equal to their revenue shares.

A necessary condition for cost minimisation and profit maximisation is that the slope of the isoquant is equal to the inverse of the negative input price ratio:

$$\frac{\partial x_i}{\partial x_j} = -\frac{x_j}{x_i}, \quad (8.47)$$

where $w_i; i = 1, \dots, N$ is the price of the i th input.

Equation (8.47) can be re-arranged in the following way:

$$-\frac{\partial x_i}{\partial x_j} = \frac{\frac{\partial D^o(x, y)}{\partial x_j}}{\frac{\partial D^o(x, y)}{\partial x_i}} = \frac{w_j}{w_i} \quad (8.48)$$

$$\frac{\frac{\partial D^o(x, y)}{\partial x_j} \frac{x_j}{D^o(x, y)}}{\frac{\partial D^o(x, y)}{\partial x_i} \frac{x_i}{D^o(x, y)}} = \frac{w_j x_j}{w_i x_i} \quad (8.49)$$

$$\frac{\epsilon_j^I}{\epsilon_i^I} = \frac{w_j x_j}{w_i x_i} \quad (8.50)$$

$$w_j x_j = \frac{\epsilon_j^I}{\epsilon_i^I} w_i x_i \quad (8.51)$$

$$\sum_{j=1}^N w_j x_j = \sum_{j=1}^N \frac{\epsilon_j^I}{\epsilon_i^I} w_i x_i \quad (8.52)$$

$$\sum_{j=1}^N w_j x_j = \frac{w_i x_i}{\epsilon_i^I} \sum_{j=1}^N \epsilon_j^I \quad (8.53)$$

$$\sum_{j=1}^N w_j x_j = \frac{w_i x_i}{\epsilon_i^I} (-\epsilon) \quad (8.54)$$

$$\frac{-\epsilon_i^I}{\epsilon} = \frac{w_i x_i}{\sum_{j=1}^N w_j x_j}, \quad (8.55)$$

where ϵ is the elasticity of scale as derived in equation (8.37). Hence, in case of cost minimisation or profit maximisation, the absolute values of the distance elasticities of the inputs divided by the elasticity of scale are equal to their cost shares.

8.1.1.5 Obtaining unobserved output prices

If some (but not all) of the outputs are non-market goods and the assumption of revenue maximisation or profit maximisation is fulfilled, one can use the condition for revenue maximisation or profit maximisation (8.38) to calculate the shadow price of a non-market output by re-arranging equation (8.39) to:

$$p_j = p_i \frac{\frac{\partial D^o(x, y)}{\partial y_j}}{\frac{\partial D^o(x, y)}{\partial y_i}}, \quad (8.56)$$

where p_j is the (non-observed) shadow price of a non-market output j and p_i is the (observed) price of an output i (see, e.g., [Reig-Martinez, Picazo-Tadeo, and Hernandez-Sancho, 2001](#)).

If the prices of the individual outputs are unknown but the total revenue ($\sum_{j=1}^M p_j y_j$) is known and the assumption of revenue maximisation or profit maximisation is fulfilled, one can use the condition for revenue maximisation or profit maximisation (8.38) to calculate the prices of the

individual outputs by re-arranging equation (8.46) to:

$$p_i = \frac{\epsilon_i^O}{y_i} \sum_{j=1}^M p_j y_j = \frac{\partial D^O(x, y)}{\partial y_i} \frac{\sum_{j=1}^M p_j y_j}{D^O(x, y)} \quad \forall i \quad (8.57)$$

(see, e.g., Färe and Primont, 1995; Reig-Martinez, Picazo-Tadeo, and Hernandez-Sancho, 2001).

8.1.2 Input distance functions

The Shepard input distance function is defined as:

$$D^i(x, y) = \max\{\lambda > 0 | x/\lambda \in L(y)\} \quad (8.58)$$

or (equivalently) as:

$$D^i(x, y) = \max\{\lambda > 0 | (x/\lambda, y) \in T\}, \quad (8.59)$$

where x is a vector of input quantities, y is a vector of output quantities, $L(y)$ is the input requirement set, and T is the technology set. The Shepard input distance function, $D^i(x, y)$, defined in (8.58) and (8.59) returns the Shepard input-oriented technical efficiencies defined in (5.3) and (5.7). Thus, it returns a value of one for fully efficient sets of inputs and outputs (x, y) , whereas it returns a value larger than one for inefficient sets of inputs and outputs (x, y) .⁴

8.1.2.1 Properties

It is usually assumed that the Shepard input distance function $D^i(x, y)$ defined in (8.58) fulfills the following properties (see, e.g., Färe and Primont, 1995; Coelli et al., 2005):

1. $D^i(x, y)$ is non-decreasing in x , i.e. $\partial D^i(x, y)/\partial x_i \geq 0 \quad \forall i = 1, \dots, N$
2. $D^i(x, y)$ is non-increasing in y , i.e. $\partial D^i(x, y)/\partial y_i \leq 0 \quad \forall i = 1, \dots, M$
3. $D^i(x, y)$ is linearly homogeneous in x , i.e. $D^i(kx, y) = k D^i(x, y) \quad \forall k > 0$
4. $D^i(x, y)$ is concave in $x \Rightarrow$ convex isoquants
5. $D^i(x, y)$ is quasiconcave in $y \Rightarrow$ concave transformation curves⁵ (i.e., joint production is advantageous compared to specialized production for a given set of input quantities)
6. $D^i(x, y) \geq 1$ indicates that x belongs to the input requirement set of y , i.e. $x \in L(y)$, while $D^i(x, y) < 1$ indicates that x does not belong to the input requirement set of y .

⁴The Farrell input distance function is defined as the inverse of the Shepard input distance function: $D^{iF}(x, y) = \min\{\lambda > 0 | \lambda x \in L(y)\} = 1/D^i(x, y)$. It returns the Farrell input-oriented technical efficiencies defined in (5.4) and (5.8).

⁵O'Donnell and Coelli (2005) clarified a typographical error in Färe and Primont (1995, p. 152) who state by mistake that the input distance function is quasiconvex in output quantities y .

7. $D^i(x, y) = 1$ indicates that x is at the boundary of the input requirement set $L(y)$, i.e. x lies on the isoquant (given y) and y lies on the transformation curve (given x).

We will illustrate these properties using a very simple input distance function with two inputs $x = (x_1, x_2)'$ and two outputs $y = (y_1, y_2)'$:⁶

$$D^i(x, y) = \frac{x_1^{0.5} x_2^{0.5}}{\sqrt{y_1^2 + y_2^2}} \quad (8.60)$$

In the following, we illustrate that this simple input distance function fulfills the above-mentioned properties 1–3:

- monotonically non-decreasing in x

$$\frac{\partial D^i(x, y)}{\partial x_i} = 0.5 \frac{x_1^{0.5} x_2^{0.5}}{x_i \sqrt{y_1^2 + y_2^2}} \geq 0; i \in \{1, 2\} \quad (8.61)$$

- monotonically non-increasing in y

$$\frac{\partial D^i(x, y)}{\partial y_i} = -\frac{x_1^{0.5} x_2^{0.5} y_i}{(y_1^2 + y_2^2)^{1.5}} \leq 0; i \in \{1, 2\} \quad (8.62)$$

- linear homogeneous in x

$$D^i(kx, y) = \frac{(k x_1)^{0.5} (k x_2)^{0.5}}{\sqrt{y_1^2 + y_2^2}} \quad (8.63)$$

$$= \frac{k^{0.5} x_1^{0.5} k^{0.5} x_2^{0.5}}{\sqrt{y_1^2 + y_2^2}} \quad (8.64)$$

$$= \frac{k x_1^{0.5} x_2^{0.5}}{\sqrt{y_1^2 + y_2^2}} \quad (8.65)$$

$$= k D^i(x, y) \quad (8.66)$$

- concave x

Concavity requires that the Hessian matrix is negative semidefinite (see section 1.5.5). The elements of the Hessian matrix, i.e. the second derivatives with respect to the input quantities, are:

$$\frac{\partial^2 D^i(x, y)}{\partial x_i \partial x_j} = 0.5 (0.5 - \Delta_{ij}) \frac{x_1^{0.5} x_2^{0.5}}{x_i x_j \sqrt{y_1^2 + y_2^2}} \quad (8.67)$$

$$\Delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (8.68)$$

⁶The specification of this input distance function is inspired by the multiple-output “ray frontier production function” suggested by Löthgren (1997, 2000).

Equation (8.67) shows that all diagonal elements of the Hessian matrix (i.e. $i = j$) are non-positive, because $(0.5 - \Delta_{ii})$ is negative for $i \in \{1, 2\}$, while all other terms on the right-hand side of (8.67) are always non-negative.

The determinant of the (2×2) Hessian matrix is:

$$|H| = \frac{\partial^2 D^i(x, y)}{\partial x_1^2} \cdot \frac{\partial^2 D^i(x, y)}{\partial x_2^2} - \left(\frac{\partial^2 D^i(x, y)}{\partial x_1 \partial x_2} \right)^2 \quad (8.69)$$

$$= 0.5 (-0.5) \frac{x_1^{0.5} x_2^{0.5}}{x_1^2 \sqrt{y_1^2 + y_2^2}} \cdot 0.5 (-0.5) \frac{x_1^{0.5} x_2^{0.5}}{x_2^2 \sqrt{y_1^2 + y_2^2}} - \left(0.5 \cdot 0.5 \frac{x_1^{0.5} x_2^{0.5}}{x_1 x_2 \sqrt{y_1^2 + y_2^2}} \right)^2 \quad (8.70)$$

$$= \frac{0.5^4}{x_1 x_2 (y_1^2 + y_2^2)} - \frac{0.5^4}{x_1 x_2 (y_1^2 + y_2^2)} = 0 \geq 0 \quad (8.71)$$

As all first-order principal minors (i.e. the two diagonal elements) are non-positive and the second-order principal (i.e. the determinant) is non-negative, we can conclude that the Hessian matrix is negative semidefinite and, thus, the input distance function (8.60) is concave in x . It is indeed not surprising that the determinant of the Hessian matrix is exactly zero, because the input distance function is linearly homogeneous in x and the determinants of linearly homogeneous functions are always zero.

Figure 8.3 illustrates the relationship between the two input quantities x_1 and x_2 and the input distance measure $D^i(x, y)$ holding the output quantities y_1 and y_2 constant so that $\sqrt{y_1^2 + y_2^2} = 1$, e.g. $y_1 = y_2 = \sqrt{0.5}$ (please note that the origin, i.e. $x_1 = x_2 = 0$, is in the front). The grey part of the surface indicates distance measures that are smaller than one ($D^i(x, y) < 1$), i.e. combinations of the input quantities x_1 and x_2 that are too small to produce the output quantities y_1 and y_2 . The dark green line is an isoquant that indicates all combinations of the input quantities x_1 and x_2 that a fully efficient firm (i.e. $D^i(x, y) = 1$) needs to produce the output quantities y_1 and y_2 . Given that the input distance function is concave in x , it is also quasiconcave in x so that the upper contour sets, i.e. the sets of input combinations that give an input distance measure ($D^i(x, y)$) larger than a certain value (e.g. 1), are convex sets (see section 1.5.6). As the origin ($x_1 = x_2 = 0$) is not inside the upper contour sets (only on its border if $D^i(x, y) = 0$), the concavity of the input distance function in x results in isoquants that are convex to the origin. The light green part of the surface indicates distance measures ($D^i(x, y)$) that are slightly larger than one, indicating small technical inefficiency. The red part of the surface indicates distance measures ($D^i(x, y)$) that are considerably larger than one, indicating considerable technical inefficiency.

- quasiconcavity in y

A sufficient condition for quasiconcavity of a function with two arguments is that the first leading principal minor of the bordered Hessian matrix is strictly negative, while its second

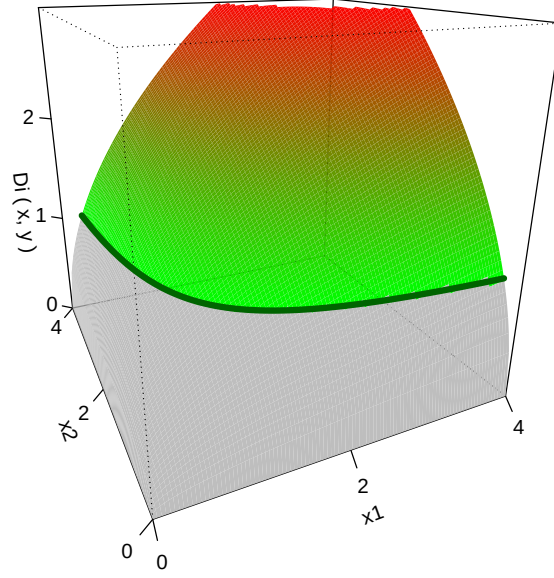


Figure 8.3: Input distance function for different input quantities

leading principle minor is strictly positive (see section 1.5.6). The first derivatives of the input distance function (8.60) with respect to the output quantities are given in (8.62). The remaining elements of the bordered Hessian matrix, i.e. the second derivatives with respect to the output quantities, are:

$$\frac{\partial^2 D^i(x, y)}{\partial y_i \partial y_j} = \frac{x_1^{0.5} x_2^{0.5}}{(y_1^2 + y_2^2)^{1.5}} \left(\frac{3 y_i y_j}{y_1^2 + y_2^2} - \Delta_{ij} \right); i \in \{1, 2\} \quad (8.72)$$

$$\Delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (8.73)$$

The first leading principal minor of the bordered Hessian B is:

$$|B_1| = - \left(\frac{\partial D^o(x, y)}{\partial x_1} \right)^2 \quad (8.74)$$

$$= - \left(- \frac{x_1^{0.5} x_2^{0.5} y_1}{(y_1^2 + y_2^2)^{1.5}} \right)^2 \quad (8.75)$$

$$= - \frac{x_1 x_2 y_1^2}{(y_1^2 + y_2^2)^3} \quad (8.76)$$

$$< 0 \quad \forall x_1, x_2 > 0; y_1 > 0; y_2 \geq 0 \quad (8.77)$$

The determinant of the (3×3) bordered Hessian is:

$$|B| = 2 \frac{\partial D^i(x, y)}{\partial y_1} \frac{\partial D^i(x, y)}{\partial y_2} \frac{\partial^2 D^i(x, y)}{\partial y_1 \partial y_2} \quad (8.78)$$

$$\begin{aligned}
& - \left(\frac{\partial D^i(x, y)}{\partial y_1} \right)^2 \frac{\partial^2 D^i(x, y)}{\partial y_2^2} - \left(\frac{\partial D^i(x, y)}{\partial y_2} \right)^2 \frac{\partial^2 D^i(x, y)}{\partial y_1^2} \\
& = 2 \left(- \frac{x_1^{0.5} x_2^{0.5} y_1}{(y_1^2 + y_2^2)^{1.5}} \right) \left(- \frac{x_1^{0.5} x_2^{0.5} y_2}{(y_1^2 + y_2^2)^{1.5}} \right) \left(\frac{3 x_1^{0.5} x_2^{0.5} y_1 y_2}{(y_1^2 + y_2^2)^{2.5}} \right) \quad (8.79)
\end{aligned}$$

$$\begin{aligned}
& - \left(- \frac{x_1^{0.5} x_2^{0.5} y_1}{(y_1^2 + y_2^2)^{1.5}} \right)^2 \left(\frac{x_1^{0.5} x_2^{0.5}}{(y_1^2 + y_2^2)^{1.5}} \left(\frac{3 y_2^2}{y_1^2 + y_2^2} - 1 \right) \right) \\
& - \left(- \frac{x_1^{0.5} x_2^{0.5} y_2}{(y_1^2 + y_2^2)^{1.5}} \right)^2 \left(\frac{x_1^{0.5} x_2^{0.5}}{(y_1^2 + y_2^2)^{1.5}} \left(\frac{3 y_1^2}{y_1^2 + y_2^2} - 1 \right) \right) \\
& = \left(\frac{x_1^{0.5} x_2^{0.5}}{(y_1^2 + y_2^2)^{1.5}} \right)^3 \left[\frac{6 y_1^2 y_2^2}{y_1^2 + y_2^2} - y_1^2 \left(\frac{3 y_2^2}{y_1^2 + y_2^2} - 1 \right) - y_2^2 \left(\frac{3 y_1^2}{y_1^2 + y_2^2} - 1 \right) \right] \quad (8.80)
\end{aligned}$$

$$= \left(\frac{x_1^{1.5} x_2^{1.5}}{(y_1^2 + y_2^2)^{4.5}} \right) [y_1^2 + y_2^2] \quad (8.81)$$

$$= \frac{x_1^{1.5} x_2^{1.5}}{(y_1^2 + y_2^2)^{3.5}} \quad (8.82)$$

$$> 0 \quad \forall x_1, x_2 > 0; y_1, y_2 \geq 0; y_1 + y_2 > 0 \quad (8.83)$$

If both of the input quantities are strictly positive and at least one of the output quantities is strictly positive, the first leading principal minor of the bordered Hessian matrix is non-positive and its determinant is non-negative, which means that the necessary conditions for quasiconvexity are fulfilled. If the quantities of both of the inputs and of the first output are strictly positive and the quantity of the second output is non-negative, the first leading principal minor of the bordered Hessian matrix is strictly negative and its determinant is strictly positive, which means that also the sufficient conditions for quasiconvexity are fulfilled. As the order of the outputs is arbitrary, we can rearrange the order of the outputs so that the quantity of the first input is strictly positive as long as at least one output quantity is strictly positive. Hence, if at least one of the output quantities is strictly positive, not only the necessary conditions but also the sufficient conditions for quasiconvexity are fulfilled.

Figure 8.4 illustrates the relationship between the two output quantities y_1 and y_2 and the input distance measure $D^i(x, y)$ holding the input quantities x_1 and x_2 constant so that $x_1^{0.5} x_2^{0.5} = 1$, e.g. $x_1 = x_2 = 1$ (please note that the origin, i.e. $y_1 = y_2 = 0$, is in the back). The grey part of the surface indicates distance measures that are smaller than one ($D^i(x, y) < 1$), i.e. combinations of the output quantities y_1 and y_2 that cannot be produced by the input quantities x_1 and x_2 . The dark green line is a transformation curve that indicates all combinations of the output quantities y_1 and y_2 that a fully efficient firm (i.e. $D^i(x, y) = 1$) can produce from the input quantities x_1 and x_2 . Given that the input distance function is quasiconcave in y , the upper contour sets, i.e. the sets of output combinations that give an input distance measure ($D^i(x, y)$) larger than a certain value

(e.g. 1), are convex sets (see section 1.5.6). As the origin ($y_1 = y_2 = 0$) is inside the upper contour sets, the quasiconcavity of the input distance function in y results in transformation curves that are concave to the origin. The light green part of the surface indicates distance measures ($D^i(x, y)$) that are slightly larger than one, indicating small technical inefficiency. The red part of the surface indicates distance measures ($D^i(x, y)$) that are considerably larger than one, indicating considerable technical inefficiency.

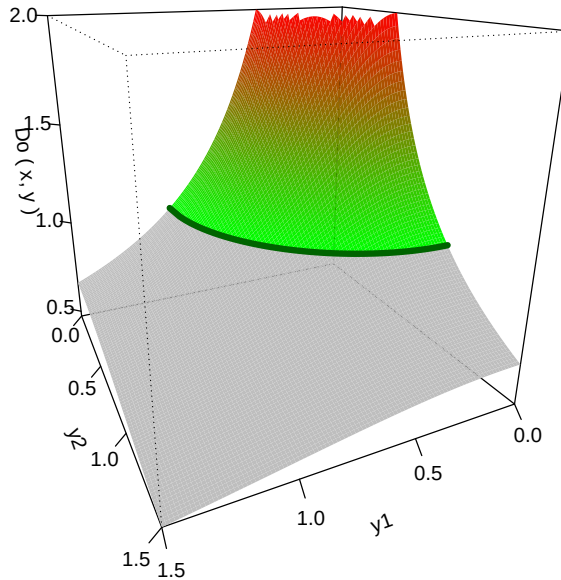


Figure 8.4: Input distance function for different output quantities

The following code generates figures 8.3 and 8.4:

```
> nVal <- 100
> colRedGreen <- colorRampPalette( c( "green", "red" ) )( 100 )
> x1 <- seq( 0, 4, length.out = nVal )
> x2 <- seq( 0, 4, length.out = nVal )
> dxMat <- outer( x1, x2, function( a, b ) a^( 0.5 ) * b^( 0.5 ) )
> dxFacet <- ( dxMat[-1, -1] + dxMat[-1, -nVal] + dxMat[-nVal, -1] +
+             dxMat[-nVal, -nVal] ) / 4
> dxInt <- cut( dxFacet,
+             seq( 1, 3, length.out = length( colRedGreen ) ) )
> dxCol <- colRedGreen[ dxInt ]
> dxCol[ dxFacet < 1 ] <- "grey"
> dxMat[ dxMat > 3 ] <- NA

> ppx <- persp( x1, x2, dxMat, xlim = c( 0, max( x1 ) ), ylim = c( 0, max( x2 ) ),
+             zlim = c( min( dxMat, na.rm = TRUE ), max( dxMat, na.rm = TRUE ) ),
```

```

+   zlab = "Di ( x, y )", ticktype = "detailed", nticks = 3,
+   theta = -25, phi = 20, col = dxCol, border = NA )
> xa <- seq( 1/max(x2), max(x1), length.out = nVal )
> lines( trans3d( xa, 1/xa, 1, ppx ), col = "darkgreen", lwd = 5 )

> y1 <- seq( 0, 1.5, length.out = nVal )
> y2 <- seq( 0, 1.5, length.out = nVal )
> dyMat <- outer( y1, y2, function( a, b ) 1 / sqrt( a^2 + b^2 ) )
> dyFacet <- ( dyMat[-1, -1] + dyMat[-1, -nVal] + dyMat[-nVal, -1] +
+   dyMat[-nVal, -nVal] ) / 4
> dyInt <- cut( dyFacet,
+   seq( 1, 2, length.out = length( colRedGreen ) ) )
> dyCol <- colRedGreen[ dyInt ]
> dyCol[ dyFacet < 1 ] <- "grey"
> # dyMat[ dyMat > dyMat[ 1, nVal ] ] <- NA
> dyMat[ dyMat > 2 ] <- NA

> ppy <- persp( y1, y2, dyMat, zlab = "Do ( x, y )",
+   ticktype = "detailed", nticks = 3,
+   theta = 155, phi = 20, col = dyCol, border = NA )
> ya <- seq( 0, 1, length.out = nVal )
> lines( trans3d( ya, sqrt(1-ya^2), 1, ppy ), col = "darkgreen", lwd = 5 )

```

8.1.2.2 Distance elasticities

A distance elasticity of the input distance function $D^i(x, y)$ with respect to an input quantity x_i indicates the percentage change in the input distance measure $D^i(x, y)$ (and, hence, in the firm's inefficiency) given a one percent increase in this input quantity. The distance elasticity of the i th input quantity is defined as:

$$\epsilon_i^I = \frac{\partial D^i(x, y)}{\partial x_i} \frac{x_i}{D^i} = \frac{\partial \ln D^i(x, y)}{\partial \ln x_i} \geq 0 \quad (8.84)$$

Linear homogeneity of the input distance function $D^i(x, y)$ in input quantities implies that its distance elasticities with respect to the input quantities sum to one:

$$\sum_{i=1}^N \epsilon_i^I = 1 \quad (8.85)$$

A distance elasticity of the input distance function $D^i(x, y)$ with respect to an output quantity y_i indicates the percentage change in the input distance measure $D^i(x, y)$ (and, hence, in the firm's inefficiency) given a one percent increase in this output quantity. The distance elasticity

of the i th output quantity is defined as:

$$\epsilon_i^O = \frac{\partial D^i(x, y)}{\partial y_i} \frac{y_i}{D^i} = \frac{\partial \ln D^i(x, y)}{\partial \ln y_i} \leq 0 \quad (8.86)$$

The linear homogeneity of the input distance function $D^i(x, y)$ in input quantities x allows an alternative interpretation of the distance elasticities of the outputs. The interpretation above equation (8.86) assumes that the remaining output quantities and all input quantities remain unchanged, while the input distance measure $D^i(x, y)$ and, thus, the technical inefficiency adjusts to the change in one output quantity. Alternatively, we can assume the remaining output quantities and the input distance measure $D^i(x, y)$ and, thus, the technical inefficiency remain unchanged, while the input quantities adjust to the change in one output quantity. Under this assumption, if the quantity of the i th output increases by one percent, all input quantities must simultaneously increase by $-\epsilon_i^O\%$ if the remaining output quantities and the technical efficiency should remain unchanged. This interpretation includes the case, where the technical (in)efficiency remains constant at one so that the change in the input quantities refers to the ‘frontier’, i.e., the minimum required input quantities.

8.1.2.3 Elasticity of scale

The elasticity of scale of an input distance function $D^i(x, y)$ equals the *inverse* of the negative sum of distance elasticities with respect to the output quantities:

$$\epsilon = \left(- \sum_{i=1}^M \epsilon_i^O \right)^{-1} \quad (8.87)$$

8.2 Cobb-Douglas output distance function

8.2.1 Specification

The general form of the Cobb-Douglas output distance function is

$$D^o(x, y) = A \left(\prod_{i=1}^N x_i^{\alpha_i} \right) \left(\prod_{i=1}^M y_i^{\beta_i} \right), \quad (8.88)$$

which can be linearized to:

$$\ln D^o(x, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^M \beta_i \ln y_i \quad (8.89)$$

with $\alpha_0 = \ln A$.

8.2.2 Estimation

The Cobb-Douglas output distance function can be expressed (and then estimated) as traditional stochastic frontier model.

In the following, we will derive, under which conditions the Cobb-Douglas output distance function defined in (8.88) and (8.89) is linear homogeneous in output quantities (y):

$$kD^o(x, y) = D^o(x, ky) \quad (8.90)$$

$$\ln(kD^o(x, y)) = \ln D^o(x, ky) \quad (8.91)$$

$$\ln(kD^o(x, y)) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^M \beta_i \ln(k y_i) \quad (8.92)$$

$$\ln k + \ln D^o(x, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^M \beta_i \ln k + \sum_{i=1}^M \beta_i \ln y_i \quad (8.93)$$

$$\ln k + \ln D^o(x, y) = \ln D^o(x, y) + \ln k \sum_{i=1}^M \beta_i \quad (8.94)$$

$$\ln k = \ln k \sum_{i=1}^M \beta_i \quad (8.95)$$

$$1 = \sum_{i=1}^M \beta_i \quad (8.96)$$

Hence, the Cobb-Douglas output distance function is linear homogeneous in output quantities if the coefficients of the output quantities (β_i) sum up to one. We can impose linear homogeneity in output quantities by re-arranging equation (8.96) to:

$$\beta_M = 1 - \sum_{i=1}^{M-1} \beta_i \quad (8.97)$$

and by substituting the right-hand side of (8.97) for β_M in (8.89):

$$\ln D^o(x, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^{M-1} \beta_i \ln(y_i) + \left(1 - \sum_{i=1}^{M-1} \beta_i\right) \ln(y_M) \quad (8.98)$$

$$\ln D^o(x, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^{M-1} \beta_i (\ln y_i - \ln y_M) + \ln y_M \quad (8.99)$$

$$-\ln y_M = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^{M-1} \beta_i \ln(y_i/y_M) - \ln D^o(x, y). \quad (8.100)$$

We can assume that $u \equiv -\ln(D^o(x, y)) \geq 0$ follows a half normal or truncated normal distribution (i.e. $u \sim N^+(\mu, \sigma_u^2)$) and add a disturbance term v , that accounts for statistical noise and

follows a normal distribution (i.e. $v \sim N(0, \sigma_v^2)$) so that we get:

$$-\ln(y_M) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^{M-1} \beta_i \ln(y_i/y_M) + u + v. \quad (8.101)$$

This specification is equivalent to the specification of stochastic frontier models so that we can use the stochastic frontier methods introduced in Chapter 6 to estimate this output distance function.

In the following, we will estimate a Cobb-Douglas output distance function with our data set of French apple producers. This data set distinguishes between two outputs: the quantity of apples produced (`qApples`) and the quantity of the other outputs (`qOtherOut`). The following command estimates the Cobb-Douglas output distance function using the command `sfa()` for stochastic frontier analysis:

```
> odFCd <- sfa( -log(qApples) ~ log( qCap ) + log( qLab ) + log( qMat ) +
+             log( qOtherOut/qApples), data = dat, ineffDecrease = FALSE )
> summary( odFCd )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency increases the endogenous variable (as in a cost function)

The dependent variable is logged

Iterative ML estimation terminated after 11 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	10.776633	1.393814	7.7318	1.061e-14 ***
log(qCap)	-0.087286	0.086890	-1.0046	0.315106
log(qLab)	-0.501954	0.157395	-3.1891	0.001427 **
log(qMat)	-0.451892	0.116505	-3.8787	0.000105 ***
log(qOtherOut/qApples)	0.621158	0.044476	13.9663	< 2.2e-16 ***
sigmaSq	1.306200	0.219877	5.9406	2.840e-09 ***
gamma	0.923605	0.039930	23.1307	< 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -147.8225

cross-sectional data

total number of observations = 140

mean efficiency: 0.5069105

8.2.3 Distance elasticities

The distance elasticities of a Cobb-Douglas output distance function with respect to input quantities (x), ϵ^I , can be calculated as:

$$\epsilon_i^I = \frac{\partial D^o(x, y)}{\partial x_i} \frac{x_i}{D^o} = \frac{\partial \ln D^o(x, y)}{\partial \ln x_i} = \alpha_i \quad (8.102)$$

The distance elasticities of a Cobb-Douglas output distance function with respect to output quantities (y), ϵ^O , can be calculated as:

$$\epsilon_i^O = \frac{\partial D^o(x, y)}{\partial y_i} \frac{y_i}{D^o} = \frac{\partial \ln D^o(x, y)}{\partial \ln y_i} = \beta_i \quad (8.103)$$

Since the distance elasticities of Cobb-Douglas output distance functions are equal to some of the estimated parameters, they can be directly obtained from the estimation output. The distance elasticity with respect to the capital input ($\epsilon_{\text{qCap}}^I = -0.087$) indicates that a 1% increase in the capital input results (*ceteris paribus*) in a 0.09% decrease of the distance measure (i.e. efficiency decreases and inefficiency increases). The distance elasticity with respect to the labor input ($\epsilon_{\text{qLab}}^I = -0.502$) indicates that a 1% increase in the labor input results (*ceteris paribus*) in a 0.5% decrease of the distance measure (i.e. efficiency decreases and inefficiency increases). The distance elasticity with respect to the material input ($\epsilon_{\text{qMat}}^I = -0.452$) indicates that a 1% increase in the materials input results (*ceteris paribus*) in a 0.45% decrease of the distance measure (i.e. efficiency decreases and inefficiency increases).

According to the alternative interpretation of the distance elasticities of the inputs (see section 8.1.1.2), the distance elasticity with respect to the capital input ($\epsilon_{\text{qCap}}^I = -0.087$) indicates that a 1% increase in the capital input increases the maximum feasible quantities of both outputs (simultaneously) by 0.09%. Similarly, the distance elasticity with respect to the labor input ($\epsilon_{\text{qLab}}^I = -0.502$) indicates that a 1% increase in the labor input increases the maximum feasible quantities of both outputs (simultaneously) by 0.5%, while the distance elasticity with respect to the material input ($\epsilon_{\text{qMat}}^I = -0.452$) indicates that a 1% increase in the materials input increases the maximum feasible quantities of both outputs (simultaneously) by 0.45%.

The distance elasticity with respect to the other output ($\epsilon_{\text{qOtherOut}}^O = 0.621$) indicates that a 1% increase in other outputs results (*ceteris paribus*) in a 0.62% increase of the distance measure (i.e. efficiency increases and inefficiency decreases). Based on the homogeneity property (8.97), we can calculate the distance elasticity of the apple quantity $\epsilon_{\text{qApples}}^O = 1 - \epsilon_{\text{qOtherOut}}^O = 0.379$, which indicates that a 1% increase in the output of apples results (*ceteris paribus*) in a 0.38% increase of the distance measure (i.e. efficiency increases and inefficiency decreases).

8.2.4 Elasticity of scale

The elasticity of scale for an output distance function $D^o(x, y)$ equals the negative sum of distance elasticities with respect to the input quantities:

$$\epsilon = - \sum_i \epsilon_i^I \quad (8.104)$$

For our Cobb-Douglas output distance function, the elasticity of scale can be obtained by using following command:

```
> -sum(coef(odfCD)[c("log(qCap)", "log(qLab)", "log(qMat)")]])
[1] 1.041133
```

The estimated elasticity of scale $\epsilon = 1.04$ indicates slightly increasing returns to scale.

8.2.5 Properties

8.2.5.1 Non-increasing in input quantities

$D^o(x, y)$ is non-increasing in x if its partial derivatives with respect to the input quantities are non-positive:

$$\frac{\partial D^o(x, y)}{\partial x_i} = \frac{\partial \ln D^o(x, y)}{\partial \ln x_i} \frac{D^o(x, y)}{x_i} = \alpha_i \frac{D^o(x, y)}{x_i} \leq 0 \quad (8.105)$$

Given that $D^o(x, y)$ and x_i are always non-negative, a Cobb-Douglas output distance function $D^o(x, y)$ is non-increasing in x if all coefficients α_i are non-positive.

We can see from the `summary()` output of our estimated Cobb-Douglas output distance function that all coefficients related to input quantities are negative. This means that monotonicity in input quantities is fulfilled.

8.2.5.2 Non-decreasing in output quantities

$D^o(x, y)$ is non-decreasing in y if its partial derivatives with respect to the output quantities are non-negative:

$$\frac{\partial D^o(x, y)}{\partial y_i} = \frac{\partial \ln D^o(x, y)}{\partial \ln y_i} \frac{D^o(x, y)}{y_i} = \beta_i \frac{D^o(x, y)}{y_i} \geq 0 \quad (8.106)$$

Given that $D^o(x, y)$ and y_i are always non-negative, a Cobb-Douglas output distance function $D^o(x, y)$ is non-decreasing in y if all coefficients β_i are non-negative.

For our Cobb-Douglas output distance function, the coefficient of the (normalized) quantity of “other outputs” is positive (0.621). The coefficient of the apple quantity can be recovered from the homogeneity condition ($1 - 0.621 = 0.379$). Since the coefficients of both output quantities are positive, we can conclude that monotonicity in output quantities is fulfilled.

8.2.5.3 Quasiconvex in input quantities

In the following, we check if our estimated Cobb-Douglas output distance function is quasiconvex in input quantities (x). A sufficient condition for quasiconvexity of $D^o(x, y)$ in x is that $|B_i| < 0; i = 1, \dots, N$, where $|B_i|$ is the i th leading principal minor of the bordered Hessian matrix B with

$$B = \begin{bmatrix} 0 & f_1 & f_2 & \dots & f_N \\ f_1 & f_{11} & f_{12} & \dots & f_{1N} \\ f_2 & f_{12} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_N & f_{1N} & f_{2N} & \dots & f_{NN} \end{bmatrix}, \quad (8.107)$$

where $f_i = \partial D^o(x, y) / \partial x_i$ are the first-order partial derivatives and f_{ij} are the second-order partial derivatives of the Cobb-Douglas output distance function $D^o(x, y)$ with respect to the input quantities x :

$$f_{ij} = \frac{\partial^2 D^o(x, y)}{\partial x_i \partial x_j} = \frac{\partial \left(\frac{\partial D^o(x, y)}{\partial x_i} \right)}{\partial x_j} = \frac{\partial \left(\alpha_i \frac{D^o(x, y)}{x_i} \right)}{\partial x_j} \quad (8.108)$$

$$= \frac{\alpha_i}{x_i} \frac{\partial D^o(x, y)}{\partial x_i} - \delta_{ij} \alpha_i \frac{D^o(x, y)}{x_i^2} \quad (8.109)$$

$$= \frac{\alpha_i}{x_i} \alpha_j \frac{D^o(x, y)}{x_j} - \delta_{ij} \alpha_i \frac{D^o(x, y)}{x_i^2} \quad (8.110)$$

$$= \alpha_i (\alpha_j - \delta_{ij}) \frac{D^o(x, y)}{x_i x_j}, \quad (8.111)$$

where δ_{ij} is (again) Kronecker's delta (2.95).

As all elements of the bordered Hessian matrix (both the first derivatives and the second derivatives) contain the output distance measure $D^o(x, y)$ as a multiplicative element, we can factor it out of the bordered Hessian matrix. As the output distance measure $D^o(x, y)$ is non-negative, it does not affect the signs of the principal minors so that we can ignore it in our further calculations.

For a sake of simplicity of the code, we create short-cuts for the coefficients.

```
> oCap <- coef(odfCD) ["log(qCap)"]
> oLab <- coef(odfCD) ["log(qLab)"]
> oMat <- coef(odfCD) ["log(qMat)"]
```

The following code computes the first-order and second-order partial derivatives of the Cobb-Douglas output distance function with respect to the input quantities (with the output distance measures being factored out of the bordered Hessian matrix or assuming that the firms are fully efficient, i.e. $D^o(x, y) = 1$):

```
> dat$fCap <- oCap * 1 / dat$qCap
> dat$fLab <- oLab * 1 / dat$qLab
```

```

> dat$fMat <- oMat * 1 / dat$qMat
> dat$fCapCap <- oCap * ( oCap - 1 ) * 1 / dat$qCap^2
> dat$fLabLab <- oLab * ( oLab - 1 ) * 1 / dat$qLab^2
> dat$fMatMat <- oMat * ( oMat - 1 ) * 1 / dat$qMat^2
> dat$fCapLab <- oCap * oLab * 1 / ( dat$qCap * dat$qLab )
> dat$fCapMat <- oCap * oMat * 1 / ( dat$qCap * dat$qMat )
> dat$fLabMat <- oLab * oMat * 1 / ( dat$qLab * dat$qMat )

```

The following code generates a three-dimensional array of the bordered Hessian matrices stacked for all observation:

```

> bhmArray <- array( 0, c( 4, 4, nrow( dat ) ) )
> bhmArray[ 1, 2, ] <- bhmArray[ 2, 1, ] <- dat$fCap
> bhmArray[ 1, 3, ] <- bhmArray[ 3, 1, ] <- dat$fLab
> bhmArray[ 1, 4, ] <- bhmArray[ 4, 1, ] <- dat$fMat
> bhmArray[ 2, 2, ] <- dat$fCapCap
> bhmArray[ 3, 3, ] <- dat$fLabLab
> bhmArray[ 4, 4, ] <- dat$fMatMat
> bhmArray[ 2, 3, ] <- bhmArray[ 3, 2, ] <- dat$fCapLab
> bhmArray[ 2, 4, ] <- bhmArray[ 4, 2, ] <- dat$fCapMat
> bhmArray[ 3, 4, ] <- bhmArray[ 4, 3, ] <- dat$fLabMat
> print( bhmArray[ , , 1 ] )

```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.000000e+00	-1.038501e-06	-1.394062e-06	-1.325718e-05
[2,]	-1.038501e-06	1.343418e-11	1.447736e-12	1.376760e-11
[3,]	-1.394062e-06	1.447736e-12	5.815100e-12	1.848134e-11
[4,]	-1.325718e-05	1.376760e-11	1.848134e-11	5.646795e-10

The following code checks if our estimated Cobb-Douglas output distance function is quasi-convex at the first observation:

```

> det( bhmArray[ 1:2, 1:2, 1 ] )

[1] -1.078485e-12

> det( bhmArray[ 1:3, 1:3, 1 ] )

[1] -2.818774e-23

> det( bhmArray[ , , 1 ] )

[1] -1.937053e-32

```

As all three leading principal minors of the bordered Hessian matrix are negative, we can conclude that our estimated Cobb-Douglas output distance function is quasiconvex in input quantities at the first observation.

The following code checks the conditions for quasiconvexity in inputs at all observations:

```
> dat$quasiConvexIn0dfCD <- apply( bhmArray[ 1:2, 1:2, ], 3, det ) < 0 &
+   apply( bhmArray[ 1:3, 1:3, ], 3, det ) < 0 &
+   apply( bhmArray, 3, det ) < 0
> table( dat$quasiConvexIn0dfCD )
```

TRUE

140

Our estimated Cobb-Douglas output distance function is quasiconvex in input quantities (x) at all observations in our data set.

8.2.5.4 Convex in output quantities

In the following, we check if our estimated Cobb-Douglas output distance function is convex in output quantities (y). An output distance function $D^o(x, y)$ is convex in y if and only if its Hessian matrix H with respect to the output quantities is positive semidefinite, which is the case if all its principal minors (not only the leading principal minors) are non-negative. The Hessian matrix with respect to the output quantities is

$$H = \begin{bmatrix} f_{11} & f_{12} & \dots & f_{1M} \\ f_{12} & f_{22} & \dots & f_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ f_{1M} & f_{2M} & \dots & f_{MM} \end{bmatrix}, \quad (8.112)$$

where f_{ij} are the second derivatives of the output distance function with respect to the output quantities:

$$f_{ij} = \frac{\partial^2 D^o(x, y)}{\partial y_i \partial y_j} = \frac{\partial \left(\frac{\partial D^o(x, y)}{\partial y_i} \right)}{\partial y_j} = \frac{\partial \left(\beta_i \frac{D^o(x, y)}{y_i} \right)}{\partial y_j} \quad (8.113)$$

$$= \frac{\beta_i}{y_i} \frac{\partial D^o(x, y)}{\partial y_i} - \delta_{ij} \beta_i \frac{D^o(x, y)}{y_i^2} \quad (8.114)$$

$$= \frac{\beta_i}{y_i} \beta_j \frac{D^o(x, y)}{y_j} - \delta_{ij} \beta_i \frac{D^o(x, y)}{y_i^2} \quad (8.115)$$

$$= \beta_i (\beta_j - \delta_{ij}) \frac{D^o(x, y)}{y_i y_j} \quad (8.116)$$

As all elements of the Hessian matrix (all second derivatives) contain the output distance measure $D^o(x, y)$ as a multiplicative element, we can factor it out of the Hessian matrix. As the

output distance measure $D^o(x, y)$ is non-negative, it does not affect the signs of the principal minors so that we can ignore it in our further calculations.

In order to facilitate the following computations, we create a short-cut for the coefficient of the (normalized) quantity of “other outputs”:

```
> oOther <- coef(odfCD)["log(qOtherOut/qApples)"]
```

Applying the homogeneity condition (8.97), we can calculate the coefficient of the apple quantity:

```
> oApples <- 1 - oOther
```

The following code computes the second derivatives of the Cobb-Douglas output distance function with respect to the output quantities (with the output distance measures being factored out of the Hessian matrix or assuming that the firms are fully efficient, i.e. $D^o(x, y) = 1$):

```
> dat$fApplesApples <- oApples * ( oApples - 1 ) * 1 / dat$qApples^2
> dat$fApplesOther <- oApples * oOther * 1 / (dat$qApples * dat$qOtherOut)
> dat$fOtherOther <- oOther * ( oOther - 1 ) * 1 / dat$qOtherOut^2
```

The following code generates a three-dimensional array of the Hessian matrices stacked for all observations:

```
> hessianArray <- array( NA, c( 2, 2, nrow( dat ) ) )
> hessianArray[ 1, 1, ] <- dat$fApplesApples
> hessianArray[ 2, 2, ] <- dat$fOtherOther
> hessianArray[ 1, 2, ] <- hessianArray[ 2, 1, ] <- dat$fApplesOther
> print( hessianArray[ , , 1 ] )
```

```
      [,1]      [,2]
[1,] -0.1214799  0.1730083
[2,]  0.1730083 -0.2463935
```

The following code checks whether the Hessian matrix is positive semidefinite at the first observation:

```
> diag( hessianArray[ , , 1 ] )

[1] -0.1214799 -0.2463935

> det( hessianArray[ , , 1 ] )

[1] 0
```

As the two diagonal elements (first-order principal minors) of the Hessian matrix are strictly negative, we can conclude that the Hessian matrix is not positive semidefinite and, thus, our

estimated Cobb-Douglas output distance function is not convex in output quantities at the first observation.

It is not surprising that the determinant of the Hessian matrix with respect to the output quantities is exactly zero or very close to zero, because the output distance function is linearly homogeneous in output quantities, which implies that the determinant of the Hessian must be zero.

The following code checks the semidefiniteness of the Hessian matrix at each individual observation:

```
> dat$convexOutOdfCD <- apply( hessianArray, 3, semidefiniteness,
+   positive = TRUE )
> table( dat$convexOutOdfCD )
```

```
FALSE
```

```
140
```

This shows that our estimated Cobb-Douglas output distance function is convex in output quantities y not at a single observation.

Indeed, Cobb-Douglas output distance functions cannot be convex in output quantities, because this functional form implies transformation curves that are convex to the origin.

8.2.6 Efficiency estimates

If there were no inefficiencies, the output distance measures $D^o(x, y)$ would be equal to one (which would imply that $u = -\ln D^o = -\ln 1 = 0$) and all observations would be on the frontier. We can use a likelihood ratio test to investigate if there is significant inefficiency.

```
> lrtest( odfCD )
```

```
Likelihood ratio test
```

```
Model 1: OLS (no inefficiency)
```

```
Model 2: Error Components Frontier (ECF)
```

```
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    6 -156.41
2    7 -147.82  1 17.179  1.701e-05 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The very small p -value confirms that there is significant inefficiency.

The efficiency estimates can be obtained by the `efficiencies` method:

```
> dat$effOdfCD <- efficiencies( odfCD )
> summary( dat$effOdfCD )
```

```

efficiency
Min.      :0.02638
1st Qu.   :0.33639
Median    :0.54408
Mean      :0.50691
3rd Qu.   :0.67876
Max.      :0.89680

```

A median Shepard output-oriented efficiency estimate of 0.544 indicates that this apple producer produces 45.6% less of each output quantity than would be possible to produce with its input quantities.

8.3 Translog output distance function

8.3.1 Specification

The general form of the Translog output distance function is:

$$\begin{aligned}
\ln D^o(x, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln x_j \\
& + \sum_{i=1}^M \beta_i \ln y_i + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln y_i \ln y_j \\
& + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln x_i \ln y_j
\end{aligned} \tag{8.117}$$

with $\alpha_{ij} = \alpha_{ji} \forall i, j = 1, \dots, N$ and $\beta_{ij} = \beta_{ji} \forall i, j = 1, \dots, M$.

8.3.2 Estimation

The Translog output distance function can be expressed (and then estimated) as traditional stochastic frontier model. This is facilitated by imposing linear homogeneity in the output quantities (y).

In the following, we will derive, under which conditions the Translog output distance function defined in (8.117) is linear homogeneous in output quantities (y):

$$k D^o(x, y) = D^o(x, k y) \tag{8.118}$$

$$\ln(k D^o(x, y)) = \ln D^o(x, k y) \tag{8.119}$$

$$\begin{aligned}
\ln(k D^o(x, y)) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln x_j \\
& + \sum_{i=1}^M \beta_i \ln(k y_i) + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln(k y_i) \ln(k y_j)
\end{aligned} \tag{8.120}$$

$$\begin{aligned}
& + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln x_i \ln(k y_j) \\
\ln k + \ln D^o(x, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln x_j \\
& + \sum_{i=1}^M \beta_i \ln(k) + \sum_{i=1}^M \beta_i \ln y_i \\
& + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln(k) \ln(k) + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln(k) \ln(y_j) \\
& + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln(y_i) \ln(k) + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln(y_i) \ln(y_j) \\
& + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln x_i \ln k + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln x_i \ln y_j
\end{aligned} \tag{8.121}$$

$$\begin{aligned}
\ln k + \ln D^o(x, y) = & \ln D^o(x, y) + \ln(k) \sum_{i=1}^M \beta_i + \ln(k) \ln(k) \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \\
& + \ln(k) \frac{1}{2} \sum_{j=1}^M \ln(y_j) \sum_{i=1}^M \beta_{ij} + \ln(k) \frac{1}{2} \sum_{i=1}^M \ln(y_i) \sum_{j=1}^M \beta_{ij} \\
& + \ln k \sum_{i=1}^N \ln x_i \sum_{j=1}^M \zeta_{ij}
\end{aligned} \tag{8.122}$$

$$\begin{aligned}
\ln k = & \ln(k) \sum_{i=1}^M \beta_i + \ln(k) \ln(k) \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \\
& + \ln(k) \frac{1}{2} \sum_{j=1}^M \ln(y_j) \sum_{i=1}^M \beta_{ij} + \ln(k) \frac{1}{2} \sum_{i=1}^M \ln(y_i) \sum_{j=1}^M \beta_{ij} \\
& + \ln k \sum_{i=1}^N \ln x_i \sum_{j=1}^M \zeta_{ij}
\end{aligned} \tag{8.123}$$

$$\begin{aligned}
1 = & \sum_{i=1}^M \beta_i + \ln(k) \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \\
& + \frac{1}{2} \sum_{j=1}^M \ln(y_j) \sum_{i=1}^M \beta_{ij} + \frac{1}{2} \sum_{i=1}^M \ln(y_i) \sum_{j=1}^M \beta_{ij} \\
& + \sum_{i=1}^N \ln x_i \sum_{j=1}^M \zeta_{ij}
\end{aligned} \tag{8.124}$$

This condition is only fulfilled for all values of k , y_i , and x_i , if the parameters fulfill the following conditions:

$$\sum_{i=1}^M \beta_i = 1 \tag{8.125}$$

$$\sum_{i=1}^M \beta_{ij} = 0 \quad \forall j \xleftrightarrow{\beta_{ij}=\beta_{ji}} \sum_{j=1}^M \beta_{ij} = 0 \quad \forall i \quad (8.126)$$

$$\sum_{j=1}^M \zeta_{ij} = 0 \quad \forall i \quad (8.127)$$

In order to impose linear homogeneity in output quantities, we can rearrange these restrictions to get:

$$\beta_M = 1 - \sum_{i=1}^{M-1} \beta_i \quad (8.128)$$

$$\beta_{Mj} = - \sum_{i=1}^{M-1} \beta_{ij} \quad \forall j = 1, \dots, M \quad (8.129)$$

$$\beta_{iM} = - \sum_{j=1}^{M-1} \beta_{ij} \quad \forall i = 1, \dots, M \quad (8.130)$$

$$\zeta_{iM} = - \sum_{j=1}^{M-1} \zeta_{ij} \quad \forall i = 1, \dots, N \quad (8.131)$$

By substituting the right-hand sides of equations (8.128) to (8.131) for β_M , β_{Mj} , β_{iM} , and ζ_{iM} in equation (8.117) and re-arranging this equation, we get:

$$\begin{aligned} -\ln y_M = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln x_j \\ & + \sum_{i=1}^{M-1} \beta_i \ln \frac{y_i}{y_M} + \frac{1}{2} \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} \beta_{ij} \ln \frac{y_i}{y_M} \ln \frac{y_j}{y_M} \\ & + \sum_{i=1}^N \sum_{j=1}^{M-1} \zeta_{ij} \ln x_i \ln \frac{y_j}{y_M} - \ln D^o(x, y). \end{aligned} \quad (8.132)$$

We can assume that $u \equiv -\ln(D^o(x, y)) \geq 0$ follows a half-normal or truncated normal distribution (i.e. $u \sim N^+(\mu, \sigma_u^2)$) and add a disturbance term v that accounts for statistical noise and follows a normal distribution (i.e. $v \sim N(0, \sigma_v^2)$) so that we get:

$$-\ln y_M = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln x_j \quad (8.133)$$

$$\begin{aligned} & + \sum_{i=1}^{M-1} \beta_i \ln \frac{y_i}{y_M} + \frac{1}{2} \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} \beta_{ij} \ln \frac{y_i}{y_M} \ln \frac{y_j}{y_M} \\ & + \sum_{i=1}^N \sum_{j=1}^{M-1} \zeta_{ij} \ln x_i \ln \frac{y_j}{y_M} + u + v. \end{aligned} \quad (8.134)$$

This specification is equivalent to the specification of stochastic frontier models so that we

can use the stochastic frontier methods introduced in Chapter 6 to estimate this output distance function. In the following, we use command `sfa()` to estimate a Translog output distance function for our data set of French apple producers:

```
> odftL <- sfa( -log(qApples) ~ log( qCap ) + log( qLab ) + log( qMat ) +
+             I( 0.5 * log( qCap )^2 ) + I( 0.5 * log( qLab )^2 ) +
+             I( 0.5 * log( qMat )^2 ) + I( log( qCap ) * log( qLab ) ) +
+             I( log( qCap ) * log( qMat ) ) + I( log( qLab ) * log( qMat ) ) +
+             log( qOtherOut/qApples) + I( 0.5 * log( qOtherOut/qApples )^2 ) +
+             I( log( qCap ) * log( qOtherOut/qApples ) ) +
+             I( log( qLab ) * log( qOtherOut/qApples ) ) +
+             I( log( qMat ) * log( qOtherOut/qApples ) ),
+ data = dat, ineffDecrease = FALSE)
> summary( odftL )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency increases the endogenous variable (as in a cost function)

The dependent variable is logged

Iterative ML estimation terminated after 28 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	38.882431	21.847091	1.7798	0.0751163
log(qCap)	0.901531	2.138098	0.4217	0.6732797
log(qLab)	-5.046845	4.402014	-1.1465	0.2515943
log(qMat)	-1.509307	3.738472	-0.4037	0.6864165
I(0.5 * log(qCap)^2)	0.068372	0.186448	0.3667	0.7138388
I(0.5 * log(qLab)^2)	1.185172	0.654824	1.8099	0.0703096
I(0.5 * log(qMat)^2)	0.204052	0.368668	0.5535	0.5799313
I(log(qCap) * log(qLab))	-0.475477	0.275217	-1.7276	0.0840518
I(log(qCap) * log(qMat))	0.401302	0.194155	2.0669	0.0387422
I(log(qLab) * log(qMat))	-0.458295	0.409950	-1.1179	0.2635971
log(qOtherOut/qApples)	-0.076972	0.784813	-0.0981	0.9218712
I(0.5 * log(qOtherOut/qApples)^2)	0.130865	0.018159	7.2065	5.740e-13
I(log(qCap) * log(qOtherOut/qApples))	-0.022233	0.052472	-0.4237	0.6717711
I(log(qLab) * log(qOtherOut/qApples))	0.028802	0.081368	0.3540	0.7233620
I(log(qMat) * log(qOtherOut/qApples))	0.057849	0.071400	0.8102	0.4178180
sigmaSq	0.677662	0.175928	3.8519	0.0001172
gamma	0.820876	0.135616	6.0530	1.422e-09

```

(Intercept) .
log(qCap)
log(qLab)
log(qMat)
I(0.5 * log(qCap)^2)
I(0.5 * log(qLab)^2) .
I(0.5 * log(qMat)^2)
I(log(qCap) * log(qLab)) .
I(log(qCap) * log(qMat)) *
I(log(qLab) * log(qMat))
log(qOtherOut/qApples)
I(0.5 * log(qOtherOut/qApples)^2) ***
I(log(qCap) * log(qOtherOut/qApples))
I(log(qLab) * log(qOtherOut/qApples))
I(log(qMat) * log(qOtherOut/qApples))
sigmaSq ***
gamma ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -116.8208

```

```

cross-sectional data
total number of observations = 140

```

```

mean efficiency: 0.6032496

```

We can use a likelihood ratio test to compare the Translog output distance function with the corresponding Cobb-Douglas output distance function:

```
> lrtest( odfCD, odfTL )
```

```
Likelihood ratio test
```

```
Model 1: odfCD
```

```
Model 2: odfTL
```

```

#Df  LogLik Df  Chisq Pr(>Chisq)
1    7 -147.82
2   17 -116.82 10 62.003  1.511e-09 ***

```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The likelihood ratio test clearly rejects the Cobb-Douglas output distance function in favor of the Translog output distance function.

8.3.3 Distance elasticities

The distance elasticities of a Translog output distance function with respect to input quantities (x), ϵ^I , can be obtained by:

$$\epsilon_i^I = \frac{\partial D^o(x, y)}{\partial x_i} \frac{x_i}{D^o} = \frac{\partial \ln D^o(x, y)}{\partial \ln x_i} = \alpha_i + \sum_{j=1}^N \alpha_{ij} \ln x_j + \sum_{j=1}^M \zeta_{ij} \ln y_j \quad (8.135)$$

The distance elasticities of a Translog output distance function with respect to output quantities (y), ϵ^O , can be obtained by:

$$\epsilon_i^O = \frac{\partial D^o(x, y)}{\partial y_i} \frac{y_i}{D^o} = \frac{\partial \ln D^o(x, y)}{\partial \ln y_i} = \beta_i + \sum_{j=1}^M \beta_{ij} \ln y_j + \sum_{j=1}^N \zeta_{ji} \ln x_j \quad (8.136)$$

In order to facilitate the calculation of the distance elasticities, we create short-cuts for the estimated coefficients:

```
> aCap <- coef( odfTL )["log(qCap)"]
> aLab <- coef( odfTL )["log(qLab)"]
> aMat <- coef( odfTL )["log(qMat)"]
> aCapCap <- coef( odfTL )["I(0.5 * log(qCap)^2)"]
> aCapLab <- aLabCap <- coef( odfTL )["I(log(qCap) * log(qLab))"]
> aCapMat <- aMatCap <- coef( odfTL )["I(log(qCap) * log(qMat))"]
> aLabLab <- coef( odfTL )["I(0.5 * log(qLab)^2)"]
> aLabMat <- aMatLab <- coef( odfTL )["I(log(qLab) * log(qMat))"]
> aMatMat <- coef( odfTL )["I(0.5 * log(qMat)^2)"]
> bOther <- coef( odfTL )["log(qOtherOut/qApples)"]
> bApples <- 1 - bOther
> bOtherOther <- coef( odfTL )["I(0.5 * log(qOtherOut/qApples)^2)"]
> bOtherApples <- bApplesOther <- - bOtherOther
> bApplesApples <- - bApplesOther
> zCapOther <- coef( odfTL )["I(log(qCap) * log(qOtherOut/qApples))"]
> zCapApples <- - zCapOther
> zLabOther <- coef( odfTL )["I(log(qLab) * log(qOtherOut/qApples))"]
> zLabApples <- - zLabOther
> zMatOther <- coef( odfTL )["I(log(qMat) * log(qOtherOut/qApples))"]
> zMatApples <- - zMatOther
```

The following R code computes the distance elasticities of the inputs and of the outputs:

```

> dat$eCapOdfTL <- with( dat, aCap +
+   aCapCap * log( qCap ) + aCapLab * log( qLab ) + aCapMat * log( qMat ) +
+   zCapApples * log( qApples ) + zCapOther * log( qOtherOut ) )
> dat$eLabOdfTL <- with( dat, aLab +
+   aLabCap * log( qCap ) + aLabLab * log( qLab ) + aLabMat * log( qMat ) +
+   zLabApples * log( qApples ) + zLabOther * log( qOtherOut ) )
> dat$eMatOdfTL <- with( dat, aMat +
+   aMatCap * log( qCap ) + aMatLab * log( qLab ) + aMatMat * log( qMat ) +
+   zMatApples * log( qApples ) + zMatOther * log( qOtherOut ) )
> dat$eApplesOdfTL <- with( dat, bApples +
+   bApplesApples * log( qApples ) + bApplesOther * log( qOtherOut ) +
+   zCapApples * log( qCap ) + zLabApples * log( qLab ) +
+   zMatApples * log( qMat ) )
> dat$eOtherOdfTL <- with( dat, bOther +
+   bOtherApples * log( qApples ) + bOtherOther * log( qOtherOut ) +
+   zCapOther * log( qCap ) + zLabOther * log( qLab ) +
+   zMatOther * log( qMat ) )
> summary( dat[ ,
+   c( "eCapOdfTL", "eLabOdfTL", "eMatOdfTL", "eApplesOdfTL", "eOtherOdfTL" ) ] )

```

eCapOdfTL	eLabOdfTL	eMatOdfTL	eApplesOdfTL
Min. : -0.75715	Min. : -1.5312	Min. : -1.6399	Min. : -0.5368
1st Qu.: -0.25962	1st Qu.: -0.8143	1st Qu.: -0.7657	1st Qu.: 0.2643
Median : -0.13894	Median : -0.4487	Median : -0.5881	Median : 0.3948
Mean : -0.11768	Mean : -0.4782	Mean : -0.5788	Mean : 0.3935
3rd Qu.: 0.01858	3rd Qu.: -0.1502	3rd Qu.: -0.3657	3rd Qu.: 0.5289
Max. : 0.38125	Max. : 0.8224	Max. : 0.3553	Max. : 1.0616

eOtherOdfTL
Min. : -0.06165
1st Qu.: 0.47106
Median : 0.60518
Mean : 0.60655
3rd Qu.: 0.73566
Max. : 1.53677

We can check whether the distance elasticities with respect to the output quantities always sum up to one:

```
> range( dat$eApplesOdfTL + dat$eOtherOdfTL )
```

```
[1] 1 1
```

8.3.4 Elasticity of scale

We can use equation (8.37) in section 8.1.1.3 to calculate the elasticity of scale for our estimated Translog output distance function:

```
> dat$eScaleOdfTL <- -rowSums( dat[, c( "eCapOdfTL", "eLabOdfTL", "eMatOdfTL" ) ] )
> summary( dat$eScaleOdfTL )
```

```
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.6621  1.0412  1.1951  1.1747  1.3053  1.7523
```

On average, the estimated elasticity of scale is $\epsilon = 1.175$, which indicates increasing returns to scale.

The following code plots the elasticities of scale against the two measures of firm size:

```
> plot( dat$qOut, dat$eScaleOdfTL, log = "x" )
> abline( 1, 0 )
> plot( dat$X, dat$eScaleOdfTL, log = "x" )
> abline( 1, 0 )
```

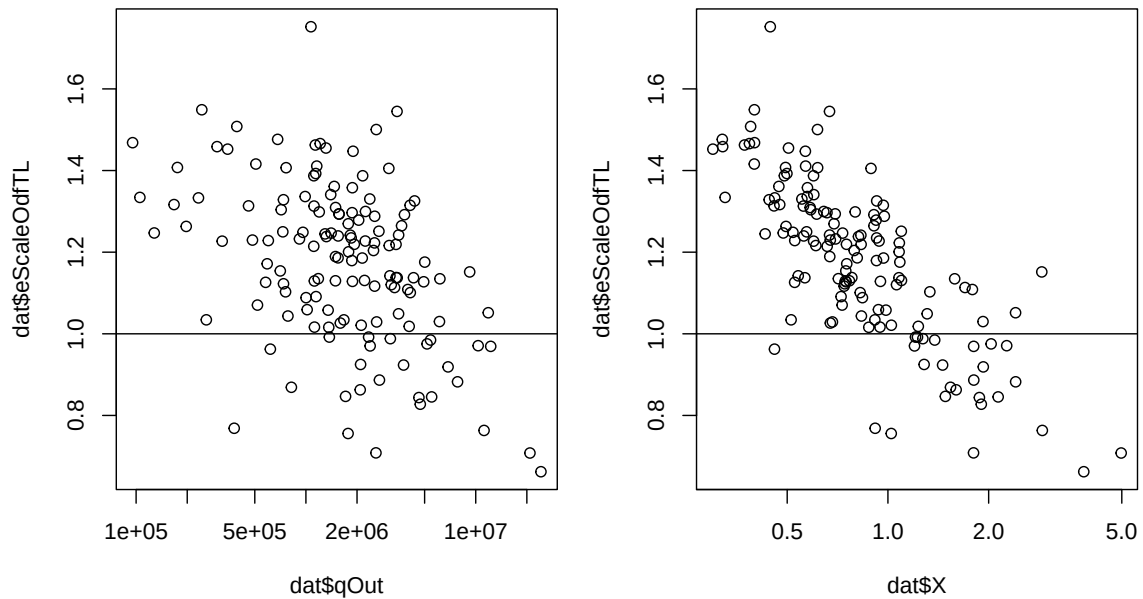


Figure 8.5: Estimated elasticities of scale for different farm sizes

The resulting graphs are shown in figure 8.5. According to these estimates of the elasticity of scale, farms with an aggregate output quantity of less than 1,000,000 units and with an aggregate input quantity of less than the sample mean generally have increasing returns to scale so that increasing their size would increase their (total factor) productivity. In contrast, farms with an aggregate output quantity of more than 5,000,000 units and with an aggregate input quantity of

more than 1.5 times the sample mean generally have decreasing returns to scale so that increasing their size would decrease their (total factor) productivity.

8.3.5 Properties

8.3.5.1 Non-increasing in input quantities

In order to investigate the monotonicity of Translog output distance function in input quantities x , we calculate the partial derivatives ($f_i; i = 1, \dots, N$) of the Translog output distance function with respect to the input quantities (x_i):

$$f_i = \frac{\partial D^o(x, y)}{\partial x_i} = \frac{\partial \ln D^o(x, y)}{\partial \ln x_i} \frac{D^o(x, y)}{x_i} = \epsilon_i^I \frac{D^o(x, y)}{x_i} \quad \forall i. \quad (8.137)$$

As the output distance measure $D^o(x, y)$ and the input quantities x_i are always non-negative, we can check the monotonicity in input quantities by checking the sign of the distance elasticities with respect to the input quantities:

```
> table( dat$eCapOdfTL <= 0 )

FALSE  TRUE
   36   104

> table( dat$eLabOdfTL <= 0 )

FALSE  TRUE
   20   120

> table( dat$eMatOdfTL <= 0 )

FALSE  TRUE
    5   135

> dat$monoInOdfTL <- dat$eCapOdfTL <= 0 & dat$eLabOdfTL <= 0 & dat$eMatOdfTL <= 0
> table( dat$monoInOdfTL )

FALSE  TRUE
   57    83
```

8.3.5.2 Non-decreasing in output quantities

In a similar way, we check the monotonicity of the Translog output distance function in output quantities y by calculating the partial derivatives ($h_i; i = 1, \dots, M$) of the Translog output distance function with respect to the output quantities (y_i):

$$h_i = \frac{\partial D^o(x, y)}{\partial y_i} = \frac{\partial \ln D^o(x, y)}{\partial \ln y_i} \frac{D^o(x, y)}{y_i} = \epsilon_i^O \frac{D^o(x, y)}{y_i} \quad \forall i. \quad (8.138)$$

As the output distance measure $D^o(x, y)$ and the output quantities y_i are always non-negative, we can check the monotonicity in output quantities by checking the sign of the distance elasticities with respect to the output quantities:

```
> table( dat$eApplesOdfTL >= 0 )
```

```
FALSE  TRUE
      7   133
```

```
> table( dat$eOtherOdfTL >= 0 )
```

```
FALSE  TRUE
      2   138
```

```
> dat$monoOutOdfTL <- dat$eApplesOdfTL >= 0 & dat$eOtherOdfTL >= 0
```

```
> table( dat$monoOutOdfTL )
```

```
FALSE  TRUE
      9   131
```

Combining the monotonicity conditions for inputs and outputs, we can investigate how many observations fulfill all monotonicity conditions:

```
> dat$monoOdfTL <- dat$monoInOdfTL & dat$monoOutOdfTL
```

```
> table( dat$monoOdfTL )
```

```
FALSE  TRUE
     63    77
```

8.3.5.3 Quasiconvex in input quantities

In the following, we will check if our estimated Translog output distance function is quasiconvex in input quantities. A sufficient condition for quasiconvexity of $D^o(x, y)$ in x is that $|B_i| < 0$; $i = 1, \dots, N$, where $|B_i|$ is the i th leading principal minor of the bordered Hessian matrix B with

$$B = \begin{bmatrix} 0 & f_1 & f_2 & \dots & f_N \\ f_1 & f_{11} & f_{12} & \dots & f_{1N} \\ f_2 & f_{12} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_N & f_{1N} & f_{2N} & \dots & f_{NN} \end{bmatrix}, \quad (8.139)$$

where $f_i = \partial D^o(x, y) / \partial x_i$ are the first-order partial derivatives and f_{ij} are the second-order partial derivatives of the Translog output distance function $D^o(x, y)$ with respect to the input quantities x .

We use equation (8.137) and the following R code to calculate the (first-order) partial derivatives of the Translog output distance function with respect to the input quantities x at the “frontier” (i.e. $D^o(x, y) = 1$) for each observation:

```
> dat$fCap0dfTL <- with( dat, eCap0dfTL / qCap )
> dat$fLab0dfTL <- with( dat, eLab0dfTL / qLab )
> dat$fMat0dfTL <- with( dat, eMat0dfTL / qMat )
> summary( dat[, c( "fCap0dfTL", "fLab0dfTL", "fMat0dfTL" ) ] )
```

fCap0dfTL	fLab0dfTL	fMat0dfTL
Min. : -8.168e-05	Min. : -1.941e-05	Min. : -2.370e-04
1st Qu.: -4.015e-06	1st Qu.: -4.461e-06	1st Qu.: -4.396e-05
Median : -1.759e-06	Median : -2.165e-06	Median : -2.649e-05
Mean : -4.040e-06	Mean : -3.099e-06	Mean : -3.557e-05
3rd Qu.: 2.070e-07	3rd Qu.: -5.804e-07	3rd Qu.: -1.203e-05
Max. : 8.224e-06	Max. : 4.679e-06	Max. : 7.724e-06

We can calculate the second-order derivatives with respect to input quantities with following formula:

$$f_{ij} = \frac{\partial^2 D^o(x, y)}{\partial x_i \partial x_j} = \frac{\partial f_i}{\partial x_j} = \frac{\partial \left(\frac{\partial \ln D^o(x, y)}{\partial \ln x_i} \frac{D^o(x, y)}{x_i} \right)}{\partial x_j} \quad (8.140)$$

$$= \frac{\partial \left(\left(\alpha_i + \sum_{k=1}^N \alpha_{ik} \ln x_k + \sum_{k=1}^M \zeta_{ik} \ln y_k \right) \frac{D^o(x, y)}{x_i} \right)}{\partial x_j} \quad (8.141)$$

$$= \frac{\alpha_{ij}}{x_i} \frac{D^o(x, y)}{x_i} + \frac{\partial \ln D^o(x, y)}{\partial \ln x_i} \frac{\partial \ln D^o(x, y)}{\partial \ln x_j} \frac{D^o(x, y)}{x_i x_j} \quad (8.142)$$

$$- \delta_{ij} \frac{\partial \ln D^o(x, y)}{\partial \ln x_j} \frac{D^o(x, y)}{x_i x_j} \quad (8.143)$$

$$= \alpha_{ij} \frac{D^o(x, y)}{x_i x_j} + \epsilon_i^I \epsilon_j^I \frac{D^o(x, y)}{x_i x_j} \quad (8.144)$$

$$- \delta_{ij} \epsilon_j^I \frac{D^o(x, y)}{x_i x_j} \quad (8.145)$$

$$= \left(\alpha_{ij} + \epsilon_i^I \epsilon_j^I - \delta_{ij} \epsilon_j^I \right) \frac{D^o(x, y)}{x_i x_j} \quad (8.146)$$

We use the following R code to compute the second-order derivatives of the Translog output distance function with respect to input quantities x for each observation:

```
> dat$fCapCap0dfTL <- with( dat,
+   ( aCapCap + eCap0dfTL^2 - eCap0dfTL ) / qCap^2 )
> dat$fLabLab0dfTL <- with( dat,
+   ( aLabLab + eLab0dfTL^2 - eLab0dfTL ) / qLab^2 )
> dat$fMatMat0dfTL <- with( dat,
```

```

+      ( aMatMat + eMatOdfTL^2 - eMatOdfTL ) / qMat^2 )
> dat$fCapLabOdfTL <- fLabCapOdfTL <- with( dat,
+      ( aCapLab + eCapOdfTL * eLabOdfTL ) / ( qCap * qLab ) )
> dat$fCapMatOdfTL <- fMatCapOdfTL <- with( dat,
+      ( aCapMat + eCapOdfTL * eMatOdfTL ) / ( qCap * qMat ) )
> dat$fLabMatOdfTL <- fMatLabOdfTL <- with( dat,
+      ( aLabMat + eLabOdfTL * eMatOdfTL ) / ( qLab * qMat ) )
> summary( dat[ , c( "fCapCapOdfTL", "fLabLabOdfTL", "fMatMatOdfTL",
+      "fCapLabOdfTL", "fCapMatOdfTL", "fLabMatOdfTL" ) ] )

```

fCapCapOdfTL	fLabLabOdfTL	fMatMatOdfTL
Min. : -1.467e-10	Min. : 7.648e-13	Min. : -1.182e-11
1st Qu.: 2.215e-12	1st Qu.: 2.479e-11	1st Qu.: 6.604e-10
Median : 3.855e-11	Median : 5.406e-11	Median : 2.264e-09
Mean : 5.273e-10	Mean : 9.229e-11	Mean : 6.558e-09
3rd Qu.: 1.061e-10	3rd Qu.: 1.046e-10	3rd Qu.: 5.759e-09
Max. : 2.840e-08	Max. : 8.129e-10	Max. : 1.022e-07

fCapLabOdfTL	fCapMatOdfTL	fLabMatOdfTL
Min. : -1.326e-09	Min. : 7.482e-12	Min. : -1.397e-09
1st Qu.: -6.957e-11	1st Qu.: 1.129e-10	1st Qu.: -1.158e-10
Median : -3.427e-11	Median : 3.527e-10	Median : -4.765e-11
Mean : -7.517e-11	Mean : 1.075e-09	Mean : -1.093e-10
3rd Qu.: -1.810e-11	3rd Qu.: 6.961e-10	3rd Qu.: -1.871e-11
Max. : -8.438e-13	Max. : 2.356e-08	Max. : 3.267e-10

The following code generates a three-dimensional array of the bordered Hessian matrices stacked for all observation:

```

> bhmArray <- array( 0, c( 4, 4, nrow( dat ) ) )
> bhmArray[ 1, 2, ] <- bhmArray[ 2, 1, ] <- dat$fCapOdfTL
> bhmArray[ 1, 3, ] <- bhmArray[ 3, 1, ] <- dat$fLabOdfTL
> bhmArray[ 1, 4, ] <- bhmArray[ 4, 1, ] <- dat$fMatOdfTL
> bhmArray[ 2, 2, ] <- dat$fCapCapOdfTL
> bhmArray[ 3, 3, ] <- dat$fLabLabOdfTL
> bhmArray[ 4, 4, ] <- dat$fMatMatOdfTL
> bhmArray[ 2, 3, ] <- bhmArray[ 3, 2, ] <- dat$fCapLabOdfTL
> bhmArray[ 2, 4, ] <- bhmArray[ 4, 2, ] <- dat$fCapMatOdfTL
> bhmArray[ 3, 4, ] <- bhmArray[ 4, 3, ] <- dat$fLabMatOdfTL
> print( bhmArray[ , , 1 ] )

```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.000000e+00	-2.502648e-06	-1.901585e-07	-2.092192e-05

```
[2,] -2.502648e-06  4.571703e-11 -1.523524e-11  1.924310e-10
[3,] -1.901585e-07 -1.523524e-11  9.705798e-12 -3.336201e-11
[4,] -2.092192e-05  1.924310e-10 -3.336201e-11  1.227134e-09
```

The following code checks if our estimated Translog output distance function is quasiconvex at the first observation:

```
> det( bhmArray[ 1:2, 1:2, 1 ] )
```

```
[1] -6.263247e-12
```

```
> det( bhmArray[ 1:3, 1:3, 1 ] )
```

```
[1] -7.694385e-23
```

```
> det( bhmArray[ , , 1 ] )
```

```
[1] -1.907569e-32
```

As all three leading principal minors of the bordered Hessian matrix are negative, we can conclude that our estimated Translog output distance function is quasiconvex in input quantities at the first observation.

The following code checks the conditions for quasiconvexity in inputs at all observations:

```
> dat$quasiConvexInOdfTL <- apply( bhmArray[ 1:2, 1:2, ], 3, det ) < 0 &
+   apply( bhmArray[ 1:3, 1:3, ], 3, det ) < 0 &
+   apply( bhmArray, 3, det ) < 0
> table( dat$quasiConvexInOdfTL )
```

```
FALSE  TRUE
   87    53
```

```
> table( dat$quasiConvexInOdfTL[ dat$monoOdfTL ] )
```

```
FALSE  TRUE
   42    35
```

Our estimated Translog output distance function is not quasiconvex in input quantities (x) at 87 out of 140 observations in our data set.

8.3.5.4 Convex in output quantities

In the following, we will check if our estimated Translog output distance function is convex in output quantities (y). An output distance function $D^o(x, y)$ is convex in y if and only if its Hessian matrix H with respect to the output quantities is positive semidefinite, which is the case if all its principal minors (not only the leading principal minors) are non-negative. The Hessian matrix with respect to the output quantities is

$$H = \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1M} \\ h_{12} & h_{22} & \dots & h_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ h_{1M} & h_{2M} & \dots & h_{MM} \end{bmatrix}, \quad (8.147)$$

where h_{ij} are the second-order derivatives of the output distance function with respect to the output quantities:

$$h_{ij} = \frac{\partial^2 D^o(x, y)}{\partial y_i \partial y_j} = \frac{\partial h_i}{\partial y_j} = \frac{\partial \left(\frac{\partial \ln D^o(x, y)}{\partial \ln y_i} \frac{D^o(x, y)}{y_i} \right)}{\partial y_j} \quad (8.148)$$

$$= \frac{\partial \left(\left(\beta_i + \sum_{k=1}^M \beta_{ik} \ln y_k + \sum_{k=1}^N \zeta_{ki} \ln x_k \right) \frac{D^o(x, y)}{y_i} \right)}{\partial y_j} \quad (8.149)$$

$$= \frac{\beta_{ij}}{y_i} \frac{D^o(x, y)}{y_i} + \frac{\partial \ln D^o(x, y)}{\partial \ln y_i} \frac{\partial \ln D^o(x, y)}{\partial \ln y_j} \frac{D^o(x, y)}{y_i y_j} \quad (8.150)$$

$$- \delta_{ij} \frac{\partial \ln D^o(x, y)}{\partial \ln y_j} \frac{D^o(x, y)}{y_i y_j} \quad (8.151)$$

$$= \beta_{ij} \frac{D^o(x, y)}{y_i y_j} + \epsilon_j^O \epsilon_i^O \frac{D^o(x, y)}{y_i y_j} \quad (8.152)$$

$$- \delta_{ij} \epsilon_j^O \frac{D^o(x, y)}{y_i y_j} \quad (8.153)$$

$$= \left(\beta_{ij} + \epsilon_j^O \epsilon_i^O - \delta_{ij} \epsilon_i^O \right) \frac{D^o(x, y)}{y_i y_j} \quad (8.154)$$

where δ_{ij} is (again) Kronecker's delta (2.95).

As all elements of the Hessian matrix (all second derivatives) contain the output distance measure $D^o(x, y)$ as a multiplicative element, we can factor it out of the Hessian matrix. As the output distance measure $D^o(x, y)$ is non-negative, it does not affect the signs of the principal minors so that we can ignore it in our further calculations.

The following code computes the second derivatives of the Translog output distance function with respect to the output quantities (with the output distance measures being factored out of the Hessian matrix or assuming that the firms are fully efficient, i.e. $D^o(x, y) = 1$):

```
> dat$hApplesApples <- with( dat,
+   ( bApplesApples + eApples0dfTL^2 - eApples0dfTL ) / qApples^2 )
```

```

> dat$hOtherOther <- with( dat,
+   ( bOtherOther + eOtherOdfTL^2 - eOtherOdfTL ) / qOtherOut^2 )
> dat$hApplesOther <- with( dat,
+   ( bApplesOther + eApplesOdfTL * eOtherOdfTL ) / ( qApples * qOtherOut ) )
> summary( dat[ , c( "hApplesApples", "hOtherOther", "hApplesOther" ) ] )

```

hApplesApples	hOtherOther	hApplesOther
Min. : -25.18	Min. : -11.74372	Min. : -2.405e+03
1st Qu.: -0.19	1st Qu.: -0.17271	1st Qu.: 7.670e-03
Median : -0.02	Median : -0.05618	Median : 3.796e-02
Mean : 19730.88	Mean : 6.24730	Mean : -2.086e+01
3rd Qu.: 0.00	3rd Qu.: -0.00998	3rd Qu.: 1.040e-01
Max. : 2568565.49	Max. : 874.60330	Max. : 1.720e+01

The following code generates a three-dimensional array of the Hessian matrices stacked for all observations:

```

> hessianArray <- array( NA, c( 2, 2, nrow( dat ) ) )
> hessianArray[ 1, 1, ] <- dat$hApplesApples
> hessianArray[ 2, 2, ] <- dat$hOtherOther
> hessianArray[ 1, 2, ] <- hessianArray[ 2, 1, ] <- dat$hApplesOther
> print( hessianArray[ , , 1 ] )

```

```

      [,1]      [,2]
[1,] -0.05665543  0.08068705
[2,]  0.08068705 -0.11491221

```

The following code checks whether the Hessian matrix is positive semidefinite at the first observation:

```

> diag( hessianArray[ , , 1 ] )

[1] -0.05665543 -0.11491221

> det( hessianArray[ , , 1 ] )

[1] 4.479031e-18

```

As the two diagonal elements (first-order principal minors) of the Hessian matrix are strictly negative, we can conclude that the Hessian matrix is not positive semidefinite and, thus, our estimated Translog output distance function is not convex in output quantities at the first observation.

It is not surprising that the determinant of the Hessian matrix with respect to the output quantities is exactly zero or very close to zero, because the output distance function is linearly

homogeneous in output quantities, which implies that the determinant of the Hessian must be zero.

The following code checks the semidefiniteness of the Hessian matrix at each individual observation:

```
> dat$convexOutOdfTL <- apply( hessianArray, 3, isSemidefinite,
+   positive = TRUE )
> table( dat$convexOutOdfTL )
```

```
FALSE  TRUE
  119    21
```

```
> table( dat$convexOutOdfTL[ dat$monoOdfTL ] )
```

```
FALSE  TRUE
   71     6
```

This shows that our estimated Translog output distance function is not convex in output quantities y at 119 out of 140 observations.

8.3.6 Efficiency estimates

If there were no inefficiencies, the output distance measure D^o of all observations would be equal to 1 (which implies $u = -\ln D^o = 0$) and all observations would be on the frontier. We can use a likelihood ratio test to investigate if there is inefficiency:

```
> lrtest( odfTL )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Error Components Frontier (ECF)

```
#Df  LogLik Df  Chisq Pr(>Chisq)
1   16 -117.87
2   17 -116.82  1 2.1065    0.07334 .
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The efficiency estimates can be obtained by the `efficiencies()` method:

```
> dat$effOdfTL <- efficiencies( odfTL )
> summary( dat$effOdfTL )
```

```

efficiency
Min.    :0.1494
1st Qu.:0.4773
Median  :0.6359
Mean    :0.6032
3rd Qu.:0.7348
Max.    :0.8843

```

The following command creates a histogram of the efficiency estimates:

```
> hist( dat$effOdfTL, 20 )
```

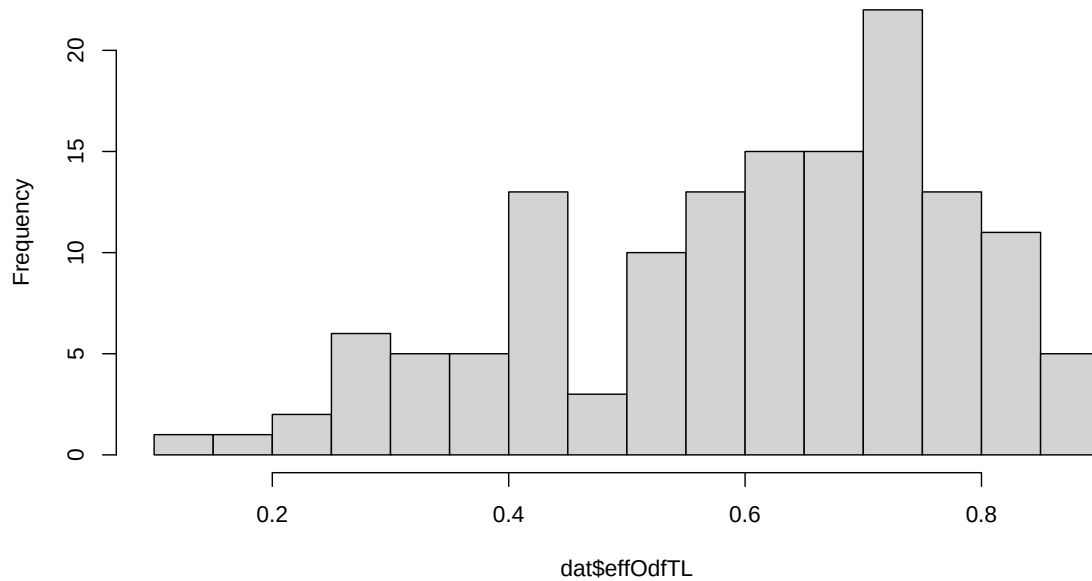


Figure 8.6: Efficiency estimates of the Translog output distance function

The resulting graph is shown in figure 8.6.

8.4 Cobb-Douglas input distance function

8.4.1 Specification

The general form of the Cobb-Douglas input distance function is:

$$D^i(x, y) = A \left(\prod_{i=1}^N x_i^{\alpha_i} \right) \left(\prod_{i=1}^M y_i^{\beta_i} \right), \quad (8.155)$$

which can be linearized to:

$$\ln D^i(x, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^M \beta_i \ln y_i \quad (8.156)$$

with $\alpha_0 = \ln A$.

8.4.2 Estimation

The Cobb-Douglas input distance function can be expressed (and then estimated) as traditional stochastic frontier model.

In the following, we will derive, under which conditions the Cobb-Douglas input distance function defined in (8.155) and (8.156) is linear homogeneous in input quantities (x):

$$kD^i(x, y) = D^i(kx, y) \quad (8.157)$$

$$\ln(kD^i(x, y)) = \ln D^i(kx, y) \quad (8.158)$$

$$\ln(kD^i(x, y)) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln k \ln x_i + \sum_{i=1}^M \beta_i \ln y_i \quad (8.159)$$

$$\ln k + \ln D^i(x, y) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln k + \sum_{i=1}^N \alpha_i \ln x_i + \sum_{i=1}^M \beta_i \ln y_i \quad (8.160)$$

$$\ln k + \ln D^i(x, y) = \ln D^i(x, y) + \ln k \sum_{i=1}^N \alpha_i \quad (8.161)$$

$$\ln k = \ln k \sum_{i=1}^N \alpha_i \quad (8.162)$$

$$1 = \sum_{i=1}^N \alpha_i \quad (8.163)$$

Hence, the Cobb-Douglas input distance function is linear homogeneous in input quantities if the coefficients of the input quantities (α_i) sum up to one. We can impose linear homogeneity in input quantities by re-arranging equation (8.163) to:

$$\alpha_N = 1 - \sum_{i=1}^{N-1} \alpha_i \quad (8.164)$$

and by substituting the right-hand side of (8.164) for α_N in (8.156):

$$\ln D^i(x, y) = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln x_i + (1 - \sum_{i=1}^{N-1} \alpha_i) \ln x_N + \sum_{i=1}^M \beta_i \ln y_i \quad (8.165)$$

$$\ln D^i(x, y) = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i (\ln x_i - \ln x_N) + \ln x_N + \sum_{i=1}^M \beta_i \ln y_i \quad (8.166)$$

$$\ln D^i(x, y) - \ln x_N = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i (\ln x_i/x_N) + \sum_{i=1}^M \beta_i \ln y_i \quad (8.167)$$

$$-\ln x_N = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i (\ln x_i/x_N) + \sum_{i=1}^M \beta_i \ln y_i - \ln D^i(x, y) \quad (8.168)$$

We can assume that $u \equiv \ln(D^i(x, y)) \geq 0$ follows a half normal or truncated normal distribution (i.e. $u \sim N^+(\mu, \sigma_u^2)$) and add a disturbance term v that accounts for statistical noise and follows a normal distribution (i.e. $v \sim N(0, \sigma_v^2)$) so that we get:

$$-\ln(x_N) = \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln(x_i/x_N) + \sum_{i=1}^M \beta_i \ln y_i - u + v \quad (8.169)$$

This specification is equivalent to the specification of stochastic frontier models so that we can use the stochastic frontier methods introduced in Chapter 6 to estimate this input distance function.

In the following, we will estimate a Cobb-Douglas input distance function with our data set of French apple producers. This data set distinguishes between two outputs: the quantity of apples produced (`qApples`) and the quantity of the other outputs (`qOtherOut`). The following command estimates the Cobb-Douglas input distance function using the command `sfa()` for stochastic frontier analysis:

```
> idfCD <- sfa( -log(qMat) ~ log( qCap/qMat ) + log( qLab/qMat ) +
+             log( qApples ) + log ( qOtherOut ),
+             ineffDecrease = TRUE, data = dat )
> summary( idfCD )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 10 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-11.084030	0.135311	-81.9155	< 2.2e-16 ***
log(qCap/qMat)	0.029982	0.040619	0.7381	0.4604
log(qLab/qMat)	0.684137	0.065195	10.4938	< 2.2e-16 ***
log(qApples)	-0.158068	0.022508	-7.0229	2.173e-12 ***
log(qOtherOut)	-0.129456	0.028820	-4.4918	7.061e-06 ***
sigmaSq	0.375209	0.065546	5.7243	1.038e-08 ***
gamma	0.925826	0.044667	20.7271	< 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -60.46615

cross-sectional data

total number of observations = 140

mean efficiency: 0.6594516

8.4.3 Distance elasticities

The distance elasticities of a Cobb-Douglas input distance function $D^i(x, y)$ with respect to input quantities (x) , ϵ^I , can be calculated as:

$$\epsilon_i^I = \frac{\partial D^i(x, y)}{\partial x_i} \frac{x_i}{D^i} = \frac{\partial \ln D^i(x, y)}{\partial \ln x_i} = \alpha_i \quad (8.170)$$

The distance elasticities of a Cobb-Douglas input distance function $D^i(x, y)$ with respect to output quantities (y) , ϵ^O , can be calculated as:

$$\epsilon_i^O = \frac{\partial D^i(x, y)}{\partial y_i} \frac{y_i}{D^i} = \frac{\partial \ln D^i(x, y)}{\partial \ln y_i} = \beta_i \quad (8.171)$$

Since the distance elasticities of Cobb-Douglas input distance functions are equal to some of the estimated parameters, they can be directly obtained from the estimation output.

The distance elasticity with respect to the capital input ($\epsilon_{\text{qCap}}^I = 0.03$) indicates that a 1% increases in the capital input results (*ceteris paribus*) in a 0.03% increase of the distance measure (i.e., inefficiency increases and efficiency decreases). The distance elasticity with respect to the labor input ($\epsilon_{\text{qLab}}^I = 0.684$) indicates that a 1% increases in the labor input results (*ceteris paribus*) in a 0.68% increase of the distance measure (i.e., inefficiency increases and efficiency decreases). The distance elasticity with respect to the material input ($\epsilon_{\text{qMat}}^I = 0.286$) indicates that a 1% increase in the material input results (*ceteris paribus*) in a 0.29% increase of the distance measure (i.e., inefficiency increases and efficiency decreases).

The distance elasticity with respect to the apples output ($\epsilon_{\text{qApples}}^O = -0.158$) indicates that a 1% increase in the apple output results (*ceteris paribus*) in a 0.16% decrease of the distance measure (i.e., inefficiency decreases and efficiency increases). The distance elasticity with respect to the other outputs ($\epsilon_{\text{qOther}}^O = -0.129$) indicates that a 1% increase in the other outputs results (*ceteris paribus*) in a 0.13% decrease of the distance measure (i.e., inefficiency decreases and efficiency increases).

According to the alternative interpretation of the distance elasticities of the outputs (see section 8.1.2.2), the distance elasticity with respect to the apples output ($\epsilon_{\text{qApples}}^O = -0.158$) indicates that a 1% increase in the apple output increases the minimum required quantities of all inputs

(simultaneously) by 0.16%. Similarly, the distance elasticity with respect to the other outputs ($\epsilon_{q0ther}^O = -0.129$) indicates that a 1% increase in the other outputs increases the minimum required quantities of all inputs (simultaneously) by 0.13%.

8.4.4 Elasticity of scale

The elasticity of scale for an input distance function $D^i(x, y)$ equals the *inverse* of the negative sum of the distance elasticities with respect to the output quantities:

$$\epsilon = \left(- \sum_{i=1}^M \epsilon_i^O \right)^{-1} \quad (8.172)$$

For our Cobb-Douglas input distance function, the elasticity of scale can be obtained by using following command:

```
> (-sum(coef(idfCD)[c("log(qApples)", "log(qOtherOut)"))])^(-1)

[1] 3.477971
```

The estimated elasticity of scale $\epsilon = 3.478$ indicates highly increasing returns to scale.

8.4.5 Properties

8.4.5.1 Non-decreasing in input quantities

$D^i(x, y)$ is non-decreasing in x if its partial derivatives with respect to the input quantities are non-negative:

$$\frac{\partial D^i(x, y)}{\partial x_i} = \frac{\partial \ln D^i(x, y)}{\partial \ln x_i} \frac{D^i(x, y)}{x_i} = \alpha_i \frac{D^i(x, y)}{x_i} \geq 0. \quad (8.173)$$

Given that $D^i(x, y)$ and x_i are always non-negative, a Cobb-Douglas input distance function fulfills the monotonicity condition, if all coefficients of the (normalized) input quantities are positive. This is the case for our estimated Cobb-Douglas input distance function, where the coefficient of the quantity of the material input (**qMat**) that was not directly estimated can be obtained from the homogeneity condition, i.e., $1 - 0.03 - 0.684 = 0.286$.

8.4.5.2 Non-increasing in output quantities

$D^i(x, y)$ is non-increasing in y if its partial derivatives with respect to the output quantities are non-positive:

$$\frac{\partial D^i(x, y)}{\partial y_i} = \frac{\partial \ln D^i(x, y)}{\partial \ln y_i} \frac{D^i(x, y)}{y_i} = \beta_i \frac{D^i(x, y)}{y_i} \leq 0. \quad (8.174)$$

Given that $D^i(x, y)$ and y_i are always non-negative, a Cobb-Douglas input distance function is non-increasing in output quantities if the coefficients of all output quantities are non-positive. This is the case for our estimated Cobb-Douglas input distance function so that we can conclude that it is non-increasing in output quantities.

8.4.5.3 Concave in input quantities

In the following, we check if the estimated Cobb-Douglas input distance function is concave in input quantities (x). $D^i(x, y)$ is concave in x if and only if its Hessian matrix H with respect to the input quantities is negative semidefinite. A sufficient condition for negative semidefiniteness of a symmetric matrix is that all its i th-order principal minors (not only its leading principal minors) are non-positive for i being odd and non-negative for i being even for all $i \in \{1, \dots, N\}$ (see section 1.5.5). The Hessian matrix with respect to the input quantities is

$$H = \begin{bmatrix} f_{11} & f_{12} & \dots & f_{1N} \\ f_{12} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ f_{1N} & f_{2N} & \dots & f_{NN} \end{bmatrix}, \quad (8.175)$$

where f_{ij} are the second derivatives of $D^i(x, y)$ with respect to input quantities x :

$$f_{ij} = \frac{\partial^2 D^i(x, y)}{\partial x_i \partial x_j} = \frac{\partial f_i}{\partial x_j} = \frac{\partial \left(\frac{\partial \ln D^i(x, y)}{\partial \ln x_i} \frac{D^i(x, y)}{x_i} \right)}{\partial x_j} \quad (8.176)$$

$$= \frac{\alpha_i}{x_i} \frac{\partial D^i(x, y)}{\partial x_i} - \delta_{ij} \alpha_i \frac{D^i(x, y)}{x_i^2} \quad (8.177)$$

$$= \frac{\alpha_i}{x_i} \alpha_j \frac{D^o(x, y)}{x_j} - \delta_{ij} \alpha_i \frac{D^i(x, y)}{x_i^2} \quad (8.178)$$

$$= \alpha_i (\alpha_j - \delta_{ij}) \frac{D^i(x, y)}{x_i x_j} \quad (8.179)$$

where δ_{ij} is (again) Kronecker's delta (2.95).

As all elements of the Hessian matrix (all second derivatives) contain the input distance measure $D^i(x, y)$ as a multiplicative element, we can factor it out of the Hessian matrix. As the input distance measure $D^i(x, y)$ is non-negative, it does not affect the signs of the principal minors so that we can ignore it in our further calculations.

Again, for the sake of simplicity of the code, we create short-cuts for the coefficients:

```
> iCap <- coef(idfCD) ["log(qCap/qMat)"]
> iLab <- coef(idfCD) ["log(qCap/qMat)"]
> iMat <- 1-(iCap+iLab)
```

The following code computes the second derivatives of our estimated Cobb-Douglas input

distance function with respect to input quantities (x) (with the input distance measures being factored out of the Hessian matrix or assuming that the firms are fully efficient, i.e. $D^i(x, y) = 1$):

```
> dat$fCapCap <- iCap * ( iCap - 1 ) * 1 / dat$qCap^2
> dat$fLabLab <- iLab * ( iLab - 1 ) * 1 / dat$qLab^2
> dat$fMatMat <- iMat * ( iMat - 1 ) * 1 / dat$qMat^2
> dat$fCapLab <- iCap * iLab * 1 / ( dat$qCap * dat$qLab )
> dat$fCapMat <- iCap * iMat * 1 / ( dat$qCap * dat$qMat )
> dat$fLabMat <- iLab * iMat * 1 / ( dat$qLab * dat$qMat )
```

The following code generates a three-dimensional array of the Hessian matrices stacked for all observations:

```
> hessianArray <- array( NA, c( 3, 3, nrow( dat ) ) )
> hessianArray[ 1, 1, ] <- dat$fCapCap
> hessianArray[ 1, 2, ] <- hessianArray[ 2, 1, ] <- dat$fCapLab
> hessianArray[ 1, 3, ] <- hessianArray[ 3, 1, ] <- dat$fCapMat
> hessianArray[ 2, 2, ] <- dat$fLabLab
> hessianArray[ 2, 3, ] <- hessianArray[ 3, 2, ] <- dat$fLabMat
> hessianArray[ 3, 3, ] <- dat$fMatMat
> print( hessianArray[ , , 1 ] )
```

```
      [,1]      [,2]      [,3]
[1,] -4.116775e-12  2.970247e-14  9.837352e-12
[2,]  2.970247e-14 -2.243230e-13  2.296342e-12
[3,]  9.837352e-12  2.296342e-12 -4.851371e-11
```

In the following, we compute all principal minors at the first observation:

```
> diag( hessianArray[ , , 1 ] )

[1] -4.116775e-12 -2.243230e-13 -4.851371e-11

> det( hessianArray[ -3, -3, 1 ] )

[1] 9.226048e-25

> det( hessianArray[ -2, -2, 1 ] )

[1] 1.029465e-22

> det( hessianArray[ -1, -1, 1 ] )

[1] 5.609554e-24
```

```
> det( hessianArray[ , , 1 ] )

[1] -3.936263e-51
```

As all first-order principal minors (diagonal elements) are negative, all second-order principal minors are positive, and the determinant (third-order principal minor) is virtually zero, the conditions for concavity are fulfilled at the first observation.

The following code checks the concavity at all observations:

```
> dat$CDidfConcaveX <- apply( hessianArray, 3, semidefiniteness,
+   positive = FALSE )
> table( dat$CDidfConcaveX )

TRUE
140
```

The estimated Cobb-Douglas input distance function is concave in input quantities at all observations.

8.4.5.4 Quasiconcave in output quantities

In the following, we check if the estimated Cobb-Douglas input distance function $D^i(x, y)$ is quasiconcave in output quantities (y). A sufficient condition for quasiconcavity in y is that $|B_1| < 0$, $|B_2| > 0$, $|B_3| < 0$, $\dots$, $(-1)^N |B_N| > 0$, where $|B_i|$ is the i th principal minor of the bordered Hessian matrix B with

$$B = \begin{bmatrix} 0 & f_1 & f_2 & \dots & f_N \\ f_1 & f_{11} & f_{12} & \dots & f_{1N} \\ f_2 & f_{12} & f_{22} & \dots & f_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_N & f_{1N} & f_{2N} & \dots & f_{NN} \end{bmatrix}, \quad (8.180)$$

where $f_i = \partial D^i(x, y) / \partial y_i$ are the first partial derivatives and f_{ij} are the second partial derivatives of the input distance function with respect to the output quantities y :

$$f_{ij} = \frac{\partial^2 D^i(x, y)}{\partial y_i \partial y_j} = \frac{\partial \left(\frac{\partial D^i(x, y)}{\partial y_i} \right)}{\partial y_j} = \frac{\partial \left(\beta_i \frac{D^i(x, y)}{y_i} \right)}{\partial y_j} \quad (8.181)$$

$$= \frac{\beta_i}{y_i} \frac{\partial D^i(x, y)}{\partial y_j} - \delta_{ij} \beta_i \frac{D^i(x, y)}{y_i^2} \quad (8.182)$$

$$= \frac{\beta_i}{y_i} \beta_j \frac{D^i(x, y)}{y_j} - \delta_{ij} \beta_i \frac{D^i(x, y)}{y_i^2} \quad (8.183)$$

$$= \beta_i (\beta_j - \delta_{ij}) \frac{D^i(x, y)}{y_i y_j}, \quad (8.184)$$

where δ_{ij} is (again) Kronecker's delta (2.95).

As all elements of the bordered Hessian matrix (both the first derivatives and the second derivatives) contain the input distance measure $D^i(x, y)$ as a multiplicative element, we can factor it out of the bordered Hessian matrix. As the output distance measure $D^i(x, y)$ is non-negative, it does not affect the signs of the principal minors so that we can ignore it in our further calculations.

In order to facilitate further computations, we create short-cuts for the coefficients regarding the output quantities y :

```
> iApples <- coef(idfCD)["log(qApples)"]
> iOther <- coef(idfCD)["log(qOtherOut)"]
```

The following code computes the first-order and second-order partial derivatives of the Cobb-Douglas input distance function with respect to the output quantities (with the input distance measures being factored out of the bordered Hessian matrix or assuming that the firms are fully efficient, i.e. $D^i(x, y) = 1$):

```
> dat$fApples <- iApples * 1 / dat$qApples
> dat$fOther <- iOther * 1 / dat$qOtherOut
> dat$fApplesApples <- iApples * ( iApples - 1 ) * 1 / dat$qApples^2
> dat$fApplesOther <- iApples * iOther * 1 / (dat$qApples * dat$qOtherOut)
> dat$fOtherOther <- iOther * ( iOther - 1 ) * 1 / dat$qOtherOut^2
```

The following code creates a three-dimensional array with the bordered Hessian matrices at all observations stacked upon each other:

```
> bhmArray <- array( 0, c( 3, 3, nrow( dat ) ) )
> bhmArray[ 1, 2, ] <- bhmArray[ 2, 1, ] <- dat$fApples
> bhmArray[ 1, 3, ] <- bhmArray[ 3, 1, ] <- dat$fOther
> bhmArray[ 2, 2, ] <- dat$fApplesApples
> bhmArray[ 2, 3, ] <- bhmArray[ 3, 2, ] <- dat$fApplesOther
> bhmArray[ 3, 3, ] <- dat$fOtherOther
> print( bhmArray[ , , 1 ] )
```

```
      [,1]      [,2]      [,3]
[1,] 0.0000000 -0.11357085 -0.13246637
[2,] -0.1135708 0.09449814 0.01504432
[3,] -0.1324664 0.01504432 0.15309443
```

We check quasiconcavity in output quantities at the first observation by calculating the first and second leading principal minor:

```
> det( bhmArray[ 1:2, 1:2, 1 ] )
```

```
[1] -0.01289834
```

```
> det( bhmArray[ , , 1 ] )
```

```
[1] -0.003180192
```

As the determinant (second leading principal minor) of the bordered Hessian matrix is negative, we can conclude that the estimated Cobb-Douglas input distance function is not quasiconcave in output quantities at the first observation.

The following code checks this at all observations:

```
> dat$CDidfQuasiConcaveY <- apply( bhmArray[1:2,1:2, ], 3, det ) < 0 &
+   apply( bhmArray, 3, det ) > 0
> table( dat$CDidfQuasiConcaveY )
```

```
FALSE
```

```
140
```

This estimated Cobb-Douglas input distance function is quasiconcave in output quantities (y) not at a single observation in our data set.

Indeed, Cobb-Douglas input distance functions cannot be quasiconcave in output quantities, because this functional form implies transformation curves that are convex to the origin.

8.4.6 Efficiency estimates

If there were no inefficiencies the input distance D^i would be equal to 1 (what would imply that $\ln D^i(1) = 0$, i.e., $u = 0$) and all observations would be on the frontier. We can use likelihood ratio test to investigate if there is inefficiency

```
> lrtest( idfCD )
```

```
Likelihood ratio test
```

```
Model 1: OLS (no inefficiency)
```

```
Model 2: Error Components Frontier (ECF)
```

```
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    6 -66.739
2    7 -60.466  1 12.546  0.0001986 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The small p -value indicates that there is statistically significant inefficiency.

The efficiency estimates can be obtained by the `efficiencies` method. By default, the `efficiencies` method calculates the efficiency estimates as $\widehat{TE} = E[e^{-u}]$. If we set argument `minusU` of the `efficiencies` method to `FALSE`, it calculates the efficiency estimates as $\widehat{TE} = E[e^u]$.⁷ Hence, in order to obtain Shepard input-oriented efficiency estimates from an input distance function, we need to set argument `minusU` of the `efficiencies` method to `FALSE`:

```
> dat$effIdfCD <- efficiencies( idfCD, minusU = FALSE )
> summary( dat$effIdfCD )
```

```
efficiency
Min.      :1.069
1st Qu.   :1.244
Median    :1.493
Mean      :1.718
3rd Qu.   :1.887
Max.      :5.207
```

A median Shepard input-oriented efficiency estimate of 1.493 indicates that this apple producer uses 49.3% more of each input quantity than would be necessary to produce its output quantities.

8.5 Translog input distance function

8.5.1 Specification

The general form of the Translog input distance function is:

$$\begin{aligned} \ln D^i(x, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln x_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln x_j \\ & + \sum_{i=1}^M \beta_i \ln y_i + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln y_i \ln y_j \\ & + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln x_i \ln y_j, \end{aligned} \quad (8.185)$$

with $\alpha_{ij} = \alpha_{ji} \forall i, j = 1, \dots, N$ and $\beta_{ij} = \beta_{ji} \forall i, j = 1, \dots, M$.

8.5.2 Estimation

The Translog input distance function can be expressed (and then estimated) as traditional stochastic frontier model. This is facilitated by imposing linear homogeneity in the input quantities (x).

⁷Please note that $E[e^u]$ is not equal to $(E[e^{-u}])^{-1}$.

In the following, we will derive, under which conditions the Translog input distance function defined in (8.185) is linear homogeneous in input quantities (x):

$$k D^i(x, y) = D^i(k x, y) \quad (8.186)$$

$$\ln(k D^i(x, y)) = \ln D^i(k x, y) \quad (8.187)$$

$$\ln(k D^i(x, y)) = \alpha_0 + \sum_{i=1}^N \alpha_i \ln(k x_i) + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln(k x_i) \ln(k x_j) \quad (8.188)$$

$$\begin{aligned} & + \sum_{i=1}^M \beta_i \ln y_i + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln y_i \ln y_j \\ & + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln(k x_i) \ln y_j \\ \ln k + \ln D^i(x, y) & = \alpha_0 + \sum_{i=1}^N \alpha_i \ln k + \sum_{i=1}^N \alpha_i \ln x_i \end{aligned} \quad (8.189)$$

$$\begin{aligned} & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln k \ln k + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^M \alpha_{ij} \ln k \ln x_j \\ & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln x_i \ln k + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^M \alpha_{ij} \ln x_i \ln x_j \end{aligned} \quad (8.190)$$

$$\begin{aligned} & + \sum_{i=1}^M \beta_i \ln y_i + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \alpha_{ij} \ln y_i \ln y_j \\ & + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln k \ln y_j + \sum_{i=1}^N \sum_{j=1}^M \zeta_{ij} \ln x_i \ln y_j \\ \ln k + \ln D^i(x, y) & = \ln D^i(x, y) + \ln k \sum_{i=1}^N \alpha_i + \ln k \ln k \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \end{aligned} \quad (8.191)$$

$$\begin{aligned} & + \ln k \frac{1}{2} \sum_{j=1}^N \ln x_j \sum_{i=1}^N \alpha_{ij} + \ln k \frac{1}{2} \sum_{i=1}^N \ln x_i \sum_{j=1}^N \alpha_{ij} \\ & + \ln k \sum_{j=1}^M \ln y_j \sum_{i=1}^N \zeta_{ij} \\ \ln k & = \ln k \sum_{i=1}^N \alpha_i + \ln k \ln k \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \end{aligned} \quad (8.192)$$

$$\begin{aligned} & + \ln k \frac{1}{2} \sum_{j=1}^N \ln x_j \sum_{i=1}^N \alpha_{ij} + \ln k \frac{1}{2} \sum_{i=1}^N \ln x_i \sum_{j=1}^M \alpha_{ij} \\ & + \ln k \sum_{j=1}^M \ln y_j \sum_{i=1}^N \zeta_{ij} \end{aligned}$$

$$\begin{aligned}
 1 = & \sum_{i=1}^N \alpha_i + \ln k \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \\
 & + \frac{1}{2} \sum_{j=1}^N \ln x_j \sum_{i=1}^N \alpha_{ij} + \frac{1}{2} \sum_{i=1}^N \ln x_i \sum_{j=1}^N \alpha_{ij} \\
 & + \sum_{j=1}^M \ln y_j \sum_{i=1}^N \zeta_{ij}
 \end{aligned} \tag{8.193}$$

This condition is only fulfilled for all values of k , x_i , and y_i if the parameters fulfill the following conditions:

$$\sum_{i=1}^N \alpha_i = 1 \tag{8.194}$$

$$\sum_{i=1}^N \alpha_{ij} = 0 \quad \forall j \xleftrightarrow{\alpha_{ij}=\alpha_{ji}} \sum_{j=1}^N \alpha_{ij} = 0 \quad \forall i \tag{8.195}$$

$$\sum_{i=1}^N \zeta_{ij} = 0 \quad \forall j \tag{8.196}$$

In order to impose linear homogeneity in input quantities, we can rearrange these restrictions to get:

$$\alpha_N = 1 - \sum_{i=1}^{N-1} \alpha_i \tag{8.197}$$

$$\alpha_{Nj} = - \sum_{i=1}^{N-1} \alpha_{ij} \quad \forall j \tag{8.198}$$

$$\alpha_{iN} = - \sum_{j=1}^{N-1} \alpha_{ij} \quad \forall i \tag{8.199}$$

$$\zeta_{Nj} = - \sum_{i=1}^{N-1} \zeta_{ij} \quad \forall j \tag{8.200}$$

By substituting the right-hand sides of equations (8.197) to (8.200) for α_N , α_{Nj} , α_{iN} , and ζ_{Nj} in equation (8.185) and re-arranging this equation, we get:

$$\begin{aligned}
 -\ln x_N = & \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln \frac{x_i}{x_N} + \frac{1}{2} \sum_{i=1}^{N-1} \sum_{j=1}^{N-1} \alpha_{ij} \ln \frac{x_i}{x_N} \ln \frac{x_j}{x_N} \\
 & + \sum_{i=1}^M \beta_i \ln y_i + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln y_i \ln y_j \\
 & + \sum_{i=1}^{N-1} \sum_{j=1}^M \zeta_{ij} \ln \frac{x_i}{x_N} \ln y_j - \ln D^i(x, y).
 \end{aligned} \tag{8.201}$$

We can assume that $u \equiv \ln(D^i(x, y)) \geq 0$ follows a half-normal or truncated normal distribution (i.e. $u \sim N^+(\mu, \sigma_u^2)$) and add a disturbance term v that accounts for statistical noise and follows a normal distribution (i.e. $v \sim N(0, \sigma_v^2)$) so that we get:

$$\begin{aligned}
 -\ln x_N = & \alpha_0 + \sum_{i=1}^{N-1} \alpha_i \ln \frac{x_i}{x_N} + \frac{1}{2} \sum_{i=1}^{N-1} \sum_{j=1}^{N-1} \alpha_{ij} \ln \frac{x_i}{x_N} \ln \frac{x_j}{x_N} \\
 & + \sum_{i=1}^M \beta_i \ln y_i + \frac{1}{2} \sum_{i=1}^M \sum_{j=1}^M \beta_{ij} \ln y_i \ln y_j \\
 & + \sum_{i=1}^{N-1} \sum_{j=1}^M \zeta_{ij} \ln \frac{x_i}{x_N} \ln y_j - u + v.
 \end{aligned} \tag{8.202}$$

This specification is equivalent to the specification of stochastic frontier models so that we can use the stochastic frontier methods introduced in Chapter 6 to estimate this input distance function. In the following, we use command `sfa()` to estimate a Translog output distance function for our data set of French apple producers:

```

> idfTL <- sfa( -log(qMat) ~ log( qCap/qMat ) + log( qLab/qMat )
+           + I( 0.5 * log( qCap/qMat )^2 ) + I( 0.5 * log( qLab/qMat )^2 )
+           + I( log( qCap/qMat ) * log( qLab/qMat ) )
+           + log( qApples) + log ( qOtherOut )
+           + I( 0.5 * log( qApples )^2 ) + I( 0.5 * log( qOtherOut )^2 )
+           + I( log( qApples) * log ( qOtherOut ) )
+           + I( log( qCap/qMat ) * log( qApples ) )
+           + I( log( qCap/qMat ) * log( qOtherOut ) )
+           + I( log( qLab/qMat ) * log( qApples ) )
+           + I( log( qLab/qMat ) * log( qOtherOut ) ),
+   data = dat, ineffDecrease = TRUE )
> summary( idfTL )

```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 22 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-10.6399124	0.3968280	-26.8124	< 2.2e-16
log(qCap/qMat)	-0.4212393	0.2066687	-2.0382	0.04153

8 Distance Functions

log(qLab/qMat)	0.5768280	0.4168776	1.3837	0.16645
I(0.5 * log(qCap/qMat)^2)	0.1343118	0.0840011	1.5989	0.10984
I(0.5 * log(qLab/qMat)^2)	-0.0586589	0.2218504	-0.2644	0.79147
I(log(qCap/qMat) * log(qLab/qMat))	0.1436734	0.1014410	1.4163	0.15668
log(qApples)	-0.3003764	0.0709196	-4.2355	2.281e-05
log(qOtherOut)	-0.2566521	0.1475028	-1.7400	0.08186
I(0.5 * log(qApples)^2)	-0.0842968	0.0110531	-7.6266	2.411e-14
I(0.5 * log(qOtherOut)^2)	-0.0204688	0.0269612	-0.7592	0.44774
I(log(qApples) * log(qOtherOut))	0.0127001	0.0181271	0.7006	0.48354
I(log(qCap/qMat) * log(qApples))	0.0535355	0.0296150	1.8077	0.07065
I(log(qCap/qMat) * log(qOtherOut))	0.0081670	0.0461674	0.1769	0.85959
I(log(qLab/qMat) * log(qApples))	0.0038709	0.0267088	0.1449	0.88477
I(log(qLab/qMat) * log(qOtherOut))	0.0406196	0.0701908	0.5787	0.56279
sigmaSq	0.2280426	0.0438838	5.1965	2.031e-07
gamma	0.9048670	0.0599052	15.1050	< 2.2e-16

(Intercept)	***
log(qCap/qMat)	*
log(qLab/qMat)	
I(0.5 * log(qCap/qMat)^2)	
I(0.5 * log(qLab/qMat)^2)	
I(log(qCap/qMat) * log(qLab/qMat))	
log(qApples)	***
log(qOtherOut)	.
I(0.5 * log(qApples)^2)	***
I(0.5 * log(qOtherOut)^2)	
I(log(qApples) * log(qOtherOut))	
I(log(qCap/qMat) * log(qApples))	.
I(log(qCap/qMat) * log(qOtherOut))	
I(log(qLab/qMat) * log(qApples))	
I(log(qLab/qMat) * log(qOtherOut))	
sigmaSq	***
gamma	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -29.39859

cross-sectional data

total number of observations = 140

```
mean efficiency: 0.7180246
```

We can use a likelihood ratio test to compare the Translog input distance function with the corresponding Cobb-Douglas input distance function:

```
> lrtest( idfCD, idfTL )
```

```
Likelihood ratio test
```

```
Model 1: idfCD
```

```
Model 2: idfTL
```

```
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    7 -60.466
2   17 -29.399 10 62.135  1.426e-09 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The likelihood ratio test clearly rejects the Cobb-Douglas input distance function in favor of the Translog input distance function.

8.5.3 Distance elasticities

The distance elasticities of a Translog input distance function with respect to the input quantities (x) , ϵ^I , can be calculated as:

$$\epsilon_i^I = \frac{\partial D^i(x, y)}{\partial x_i} \frac{x_i}{D^i} = \frac{\partial \ln D^i(x, y)}{\partial \ln x_i} = \alpha_i + \sum_{j=1}^N \alpha_{ij} \ln x_j + \sum_{j=1}^M \zeta_{ij} \ln y_j \quad (8.203)$$

The distance elasticities of a Translog input distance function with respect to the output quantities (y) , ϵ^O , can be calculated as:

$$\epsilon_i^O = \frac{\partial D^i(x, y)}{\partial y_i} \frac{y_i}{D^i} = \frac{\partial \ln D^i(x, y)}{\partial \ln y_i} = \beta_i + \sum_{j=1}^M \beta_{ij} \ln y_j + \sum_{j=1}^N \zeta_{ji} \ln x_j \quad (8.204)$$

In order to facilitate the calculation of the distance elasticities, we create short-cuts for the estimated coefficients:

```
> aCap <- coef( idfTL )["log(qCap/qMat)"]
> aLab <- coef( idfTL )["log(qLab/qMat)"]
> aMat <- 1 - ( aCap + aLab )
> aCapCap <- coef( idfTL )["I(0.5 * log(qCap/qMat)^2)"]
> aCapLab <- aLabCap <- coef( idfTL )["I(log(qCap/qMat) * log(qLab/qMat))"]
```

```

> aCapMat <- aMatCap <- - ( aCapCap + aCapLab )
> aLabLab <- coef( idfTL )["I(0.5 * log(qLab/qMat)^2)"]
> aLabMat <- aMatLab <- - ( aLabCap + aLabLab )
> aMatMat <- - ( aMatCap + aMatLab )
> bApples <- coef( idfTL )["log(qApples)"]
> bOther <- coef( idfTL )["log(qOtherOut)"]
> bApplesApples <- coef( idfTL )["I(0.5 * log(qApples)^2)"]
> bOtherOther <- coef( idfTL )["I(0.5 * log(qOtherOut)^2)"]
> bOtherApples <- bApplesOther <- coef( idfTL )["I(log(qApples) * log(qOtherOut))"]
> zCapOther <- coef( idfTL )["I(log(qCap/qMat) * log(qOtherOut))"]
> zCapApples <- coef( idfTL )["I(log(qCap/qMat) * log(qApples))"]
> zLabOther <- coef( idfTL )["I(log(qLab/qMat) * log(qOtherOut))"]
> zLabApples <- coef( idfTL )["I(log(qLab/qMat) * log(qApples))"]
> zMatOther <- - ( zCapOther + zLabOther )
> zMatApples <- - ( zCapApples + zLabApples )

```

The following R code computes the distance elasticities of the inputs and of the outputs:

```

> dat$eCapTLidf <- with( dat, aCap +
+   aCapCap * log( qCap ) + aCapLab * log( qLab ) + aCapMat * log( qMat ) +
+   zCapApples * log( qApples ) + zCapOther * log( qOtherOut ) )
> dat$eLabTLidf <- with( dat, aLab +
+   aLabLab * log( qLab ) + aLabCap * log( qCap ) + aLabMat * log( qMat ) +
+   zLabApples * log( qApples ) + zLabOther * log( qOtherOut ) )
> dat$eMatTLidf <- with( dat, aMat +
+   aMatMat * log( qMat ) + aMatCap * log( qCap ) + aMatLab * log( qLab ) +
+   zMatApples * log( qApples ) + zMatOther * log( qOtherOut ) )
> dat$eApplesTLidf <- with( dat, bApples +
+   bApplesApples * log( qApples ) + bApplesOther * log( qOtherOut ) +
+   zCapApples * log( qCap ) + zLabApples * log( qLab ) +
+   zMatApples * log( qMat ) )
> dat$eOtherTLidf <- with( dat, bOther +
+   bOtherApples * log( qApples ) + bOtherOther * log( qOtherOut ) +
+   zCapOther * log( qCap ) + zLabOther * log( qLab ) +
+   zMatOther * log( qMat ) )
> summary( dat[ ,
+   c( "eCapTLidf", "eLabTLidf", "eMatTLidf", "eApplesTLidf", "eOtherTLidf" ) ] )

```

eCapTLidf	eLabTLidf	eMatTLidf	eApplesTLidf
Min. : -0.59401	Min. : 0.2841	Min. : -0.1160	Min. : -0.5525
1st Qu.: -0.05709	1st Qu.: 0.5514	1st Qu.: 0.1759	1st Qu.: -0.3402

```

Median : 0.06433   Median :0.6240   Median : 0.3072   Median :-0.2571
Mean   : 0.04577   Mean   :0.6124   Mean   : 0.3418   Mean   :-0.2309
3rd Qu.: 0.16061   3rd Qu.:0.6818   3rd Qu.: 0.5105   3rd Qu.: -0.1431
Max.    : 0.35954   Max.    :0.8099   Max.    : 1.0109   Max.    : 0.3607
  eOtherTLidf
Min.     :-0.29949
1st Qu.  :-0.18255
Median   :-0.15620
Mean     :-0.15859
3rd Qu.  :-0.13553
Max.     :-0.07973

```

We can check whether the distance elasticities with respect to the input quantities always sum up to one:

```

> range( dat$eCapTLidf + dat$eLabTLidf + dat$eMatTLidf )

[1] 1 1

```

8.5.4 Elasticity of scale

We can use equation (8.87) in section 8.1.2.3 to calculate the elasticity of scale for our estimated Translog input distance function:

```

> dat$eScaleTLidf <- ( - ( dat$eApplesTLidf + dat$eOtherTLidf ) )^(-1)
> summary( dat$eScaleTLidf )

      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-53.173   2.028    2.368    3.242   3.224   69.218

```

8.5.5 Efficiency estimates

If there were no inefficiencies, the input distance measure D^i of all observations would be equal to 1 (which implies $u = \ln D^i = 0$) and all observations would be on the frontier. We can use a likelihood ratio test to investigate whether there is inefficiency:

```

> lrtest( idfTL )

```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Error Components Frontier (ECF)

```

#Df  LogLik Df  Chisq Pr(>Chisq)

```



```

1  16 -31.980
2  17 -29.399  1 5.1635    0.01153 *
---
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The test rejects the hypothesis that all firms are fully technically efficient at 5% significance level.

The Shepard input-oriented efficiency estimates can be obtained by the `efficiencies` method:

```

> dat$effIdfTL <- efficiencies( idfTL, minusU = FALSE )
> summary( dat$effIdfTL )
```

```

  efficiency
Min.      :1.059
1st Qu.   :1.190
Median    :1.348
Mean      :1.498
3rd Qu.   :1.719
Max.      :2.937
```

8.5.6 Imposing monotonicity

The monotonicity conditions can be written as:

$$\frac{\partial \ln D^i(x, y)}{\partial \ln x_i} = \alpha_i + \sum_{j=1}^N \alpha_{ij} \ln x_j + \sum_{j=1}^M \zeta_{ij} \ln y_j \geq 0 \quad \forall i = 1, \dots, N \quad (8.205)$$

$$\frac{\partial \ln D^i(x, y)}{\partial \ln y_i} = \beta_i + \sum_{j=1}^M \beta_{ij} \ln y_j + \sum_{j=1}^N \zeta_{ji} \ln x_j \leq 0 \quad \forall i = 1, \dots, M \quad (8.206)$$

Replacing the non-estimated coefficients by the right-hand sides of equations (8.197) to (8.200) and multiplying the monotonicity restrictions regarding the outputs by -1 , we get:

$$\frac{\partial \ln D^i(x, y)}{\partial \ln x_i} = \alpha_i + \sum_{j=1}^{N-1} \alpha_{ij} \ln x_j + \alpha_{iN} \ln x_N + \sum_{j=1}^M \zeta_{ij} \ln y_j \quad (8.207)$$

$$= \alpha_i + \sum_{j=1}^{N-1} \alpha_{ij} \ln x_j + \left(- \sum_{j=1}^{N-1} \alpha_{ij} \right) \ln x_N + \sum_{j=1}^M \zeta_{ij} \ln y_j \quad (8.208)$$

$$= \alpha_i + \sum_{j=1}^{N-1} \alpha_{ij} \ln \frac{x_j}{x_N} + \sum_{j=1}^M \zeta_{ij} \ln y_j \geq 0 \quad \forall i = 1, \dots, N \quad (8.209)$$

$$\frac{\partial \ln D^i(x, y)}{\partial \ln x_N} = \alpha_N + \sum_{j=1}^{N-1} \alpha_{Nj} \ln \frac{x_j}{x_N} + \sum_{j=1}^M \zeta_{Nj} \ln y_j \quad (8.210)$$

$$= 1 - \sum_{i=1}^{N-1} \alpha_i + \sum_{j=1}^{N-1} \left(- \sum_{i=1}^{N-1} \alpha_{ij} \right) \ln \frac{x_j}{x_N} + \sum_{j=1}^M \left(- \sum_{i=1}^{N-1} \zeta_{ij} \right) \ln y_j \quad (8.211)$$

$$= 1 - \sum_{i=1}^{N-1} \alpha_i - \sum_{i=1}^{N-1} \sum_{j=1}^{N-1} \alpha_{ij} \ln \frac{x_j}{x_N} - \sum_{i=1}^{N-1} \sum_{j=1}^M \zeta_{ij} \ln y_j \geq 0 \quad (8.212)$$

$$\Leftrightarrow - \sum_{i=1}^{N-1} \alpha_i - \sum_{i=1}^{N-1} \sum_{j=1}^{N-1} \alpha_{ij} \ln \frac{x_j}{x_N} - \sum_{i=1}^{N-1} \sum_{j=1}^M \zeta_{ij} \ln y_j \geq -1 \quad (8.213)$$

$$-\frac{\partial \ln D^i(x, y)}{\partial \ln y_i} = -\beta_i - \sum_{j=1}^M \beta_{ij} \ln y_j - \sum_{j=1}^{N-1} \zeta_{ji} \ln x_j - \zeta_{Ni} \ln x_N \quad (8.214)$$

$$= -\beta_i - \sum_{j=1}^M \beta_{ij} \ln y_j - \sum_{j=1}^{N-1} \zeta_{ji} \ln x_j - \left(- \sum_{j=1}^{N-1} \zeta_{ji} \right) \ln x_N \quad (8.215)$$

$$= -\beta_i - \sum_{j=1}^M \beta_{ij} \ln y_j - \sum_{j=1}^{N-1} \zeta_{ji} \ln \frac{x_j}{x_N} \geq 0 \quad (8.216)$$

We can write these monotonicity restrictions using matrix notation as:

$$R \theta \geq r, \quad (8.217)$$

where

$$R = [R_\alpha, R_\beta, R_\zeta] \text{ with} \quad (8.218)$$

$$R_\alpha = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & \ln \frac{x_1}{x_N} & \ln \frac{x_2}{x_N} & \dots & \ln \frac{x_{N-1}}{x_N} & 0 & \dots & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 & \ln \frac{x_1}{x_N} & \dots & 0 & \ln \frac{x_2}{x_N} & \dots & \ln \frac{x_{N-1}}{x_N} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 & 0 & \dots & \ln \frac{x_1}{x_N} & 0 & \dots & \frac{x_2}{x_N} & \dots & \frac{x_{N-1}}{x_N} \\ 0 & -1 & -1 & \dots & -1 & -\ln \frac{x_1}{x_N} & -\left(\ln \frac{x_1}{x_N} + \ln \frac{x_2}{x_N}\right) & \dots & -\left(\ln \frac{x_1}{x_N} + \ln \frac{x_{N-1}}{x_N}\right) & -\ln \frac{x_2}{x_N} & \dots & -\left(\frac{x_2}{x_N} + \frac{x_{N-1}}{x_N}\right) & \dots & -\frac{x_{N-1}}{x_N} \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \end{bmatrix} \quad (8.219)$$

$$R_\beta = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 & \dots & 0 \\ -1 & 0 & \dots & 0 & -\ln y_1 & -\ln y_2 & \dots & -\ln y_M & 0 & \dots & 0 & \dots & 0 \\ 0 & -1 & \dots & 0 & 0 & -\ln y_1 & \dots & 0 & -\ln y_2 & \dots & -\ln y_M & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 & 0 & 0 & \dots & -\ln y_1 & 0 & \dots & -\ln y_2 & \dots & -\ln y_M \end{bmatrix} \quad (8.220)$$

$$R_{\zeta} = \begin{bmatrix} \ln y_1 & \ln y_2 & \dots & \ln y_M & 0 & 0 & \dots & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & \ln y_1 & \ln y_2 & \dots & \ln y_M & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \dots & \ln y_1 & \ln y_2 & \dots & \ln y_M \\ -\ln y_1 & -\ln y_2 & \dots & -\ln y_M & -\ln y_1 & -\ln y_2 & \dots & -\ln y_M & \dots & -\ln y_1 & -\ln y_2 & \dots & -\ln y_M \\ -\ln \frac{x_1}{x_N} & 0 & \dots & 0 & -\ln \frac{x_2}{x_N} & 0 & \dots & 0 & \dots & -\ln \frac{x_{N-1}}{x_N} & 0 & \dots & 0 \\ 0 & -\ln \frac{x_1}{x_N} & \dots & 0 & 0 & -\ln \frac{x_2}{x_N} & \dots & 0 & \dots & 0 & -\ln \frac{x_{N-1}}{x_N} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -\ln \frac{x_1}{x_N} & 0 & 0 & \dots & -\ln \frac{x_2}{x_N} & \dots & 0 & 0 & \dots & -\ln \frac{x_{N-1}}{x_N} \end{bmatrix} \quad (8.221)$$

is the restriction matrix for the monotonicity restrictions $(\partial \ln D^i(x, y)/\partial \ln x_1, \partial \ln D^i(x, y)/\partial \ln x_2, \dots, \partial \ln D^i(x, y)/\partial \ln x_{N-1}, \partial \ln D^i(x, y)/\partial \ln x_N, \partial \ln D^i(x, y)/\partial \ln y_1, \partial \ln D^i(x, y)/\partial \ln y_2, \dots, \partial \ln D^i(x, y)/\partial \ln y_M)$,

$$\theta = (\alpha', \beta', \zeta')' \text{ with} \quad (8.222)$$

$$\alpha = (\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_{N-1}, \alpha_{11}, \alpha_{12}, \dots, \alpha_{1,N-1}, \alpha_{22}, \dots, \alpha_{2,N-1}, \dots, \alpha_{N-1,N-1})' \quad (8.223)$$

$$\beta = (\beta_1, \beta_2, \dots, \beta_M, \beta_{11}, \beta_{12}, \dots, \beta_{1M}, \beta_{22}, \dots, \beta_{2M}, \dots, \beta_{MM})' \quad (8.224)$$

$$\zeta = (\zeta_{11}, \zeta_{12}, \dots, \zeta_{1M}, \zeta_{21}, \zeta_{22}, \dots, \zeta_{2M}, \dots, \zeta_{N-1,1}, \zeta_{N-1,2}, \dots, \zeta_{N-1,M})' \quad (8.225)$$

is the vector of estimated coefficients, and $r = (0, 0, \dots, 0, -1, 0, 0, \dots, 0)$ is the vector of right-hand-side values.

If one has N inputs and M outputs and one wants to impose monotonicity at n observations, the restriction matrix R must have $n \cdot (N + M)$ rows and $1 + (N - 1)(1 + N/2) + M(3 + M)/2 + (N - 1)M$ columns, where the number of columns is equal to the number of estimated coefficients. As our data set has 140 observations and we have 3 inputs and 2 outputs, our restriction matrix must have $140 \cdot (3 + 2) = 700$ rows and $1 + 2 \cdot 2.5 + 2 \cdot 5/2 + 2 \cdot 2 = 15$ columns,

The following code generates the restriction matrix and the restriction vector:

```
> RMat <- matrix( 0, nrow = 5 * nrow( dat ), ncol = 15 )
> colnames( RMat ) <- names( coef( idfTL )[1:15] )
> rVec <- rep( 0, 5 * nrow( dat ) )
> # regarding capital input
> rowsCap <- 1:nrow(dat)
> RMat[ rowsCap, "log(qCap/qMat)" ] <- 1
> RMat[ rowsCap, "I(0.5 * log(qCap/qMat)^2)" ] <- log( dat$qCap / dat$qMat )
> RMat[ rowsCap, "I(log(qCap/qMat) * log(qLab/qMat))" ] <- log( dat$qLab / dat$qMat )
> RMat[ rowsCap, "I(log(qCap/qMat) * log(qApples))" ] <- log( dat$qApples )
> RMat[ rowsCap, "I(log(qCap/qMat) * log(qOtherOut))" ] <- log( dat$qOtherOut )
> # regarding labor input
> rowsLab <- ( nrow(dat) + 1 ):( 2 * nrow(dat) )
> RMat[ rowsLab, "log(qLab/qMat)" ] <- 1
> RMat[ rowsLab, "I(0.5 * log(qLab/qMat)^2)" ] <- log( dat$qLab / dat$qMat )
```

```

> RMat[ rowsLab, "I(log(qCap/qMat) * log(qLab/qMat))" ] <- log( dat$qCap / dat$qMat )
> RMat[ rowsLab, "I(log(qLab/qMat) * log(qApples))" ] <- log( dat$qApples )
> RMat[ rowsLab, "I(log(qLab/qMat) * log(qOtherOut))" ] <- log( dat$qOtherOut )
> # regarding materials input
> rowsMat <- ( 2 * nrow(dat) + 1 ):( 3 * nrow(dat) )
> RMat[ rowsMat, "log(qCap/qMat)" ] <- -1
> RMat[ rowsMat, "log(qLab/qMat)" ] <- -1
> RMat[ rowsMat, "I(0.5 * log(qCap/qMat)^2)" ] <- -log( dat$qCap / dat$qMat )
> RMat[ rowsMat, "I(0.5 * log(qLab/qMat)^2)" ] <- -log( dat$qLab / dat$qMat )
> RMat[ rowsMat, "I(log(qCap/qMat) * log(qLab/qMat))" ] <-
+   - ( log( dat$qCap / dat$qMat ) + log( dat$qLab / dat$qMat ) )
> RMat[ rowsMat, "I(log(qCap/qMat) * log(qApples))" ] <- -log( dat$qApples )
> RMat[ rowsMat, "I(log(qCap/qMat) * log(qOtherOut))" ] <- -log( dat$qOtherOut )
> RMat[ rowsMat, "I(log(qLab/qMat) * log(qApples))" ] <- -log( dat$qApples )
> RMat[ rowsMat, "I(log(qLab/qMat) * log(qOtherOut))" ] <- -log( dat$qOtherOut )
> rVec[ rowsMat ] <- -1
> # regarding apple output
> rowsApp <- ( 3 * nrow(dat) + 1 ):( 4 * nrow(dat) )
> RMat[ rowsApp, "log(qApples)" ] <- -1
> RMat[ rowsApp, "I(0.5 * log(qApples)^2)" ] <- -log( dat$qApples )
> RMat[ rowsApp, "I(log(qApples) * log(qOtherOut))" ] <- -log( dat$qOtherOut )
> RMat[ rowsApp, "I(log(qCap/qMat) * log(qApples))" ] <- -log( dat$qCap / dat$qMat )
> RMat[ rowsApp, "I(log(qLab/qMat) * log(qApples))" ] <- -log( dat$qLab / dat$qMat )
> # regarding other outputs
> rowsOth <- ( 4 * nrow(dat) + 1 ):( 5 * nrow(dat) )
> RMat[ rowsOth, "log(qOtherOut)" ] <- -1
> RMat[ rowsOth, "I(log(qApples) * log(qOtherOut))" ] <- -log( dat$qApples )
> RMat[ rowsOth, "I(0.5 * log(qOtherOut)^2)" ] <- -log( dat$qOtherOut )
> RMat[ rowsOth, "I(log(qCap/qMat) * log(qOtherOut))" ] <- -log( dat$qCap / dat$qMat )
> RMat[ rowsOth, "I(log(qLab/qMat) * log(qOtherOut))" ] <- -log( dat$qLab / dat$qMat )

```

As $R\theta - r$ should be equal to the distance elasticities of the inputs and the negative distance elasticities of the outputs, we can use the following code to check whether we have specified the restriction matrix R and the restriction vector r correctly:

```

> all.equal( c( RMat %*% coef( idfTL )[1:15] - rVec ),
+   c( dat$eCapTLidf, dat$eLabTLidf, dat$eMatTLidf,
+     -dat$eApplesTLidf, -dat$eOtherTLidf ) )

```

```
[1] TRUE
```

We can obtain restricted coefficients as suggested by [Henningsen and Henning \(2009\)](#):

```

> uCoef <- coef( idfTL )[1:15]
> uCovInv <- solve( vcov( idfTL )[ 1:15,1:15] )
> library( "quadprog" )
> minDistResult <- solve.QP( Dmat = uCovInv, dvec = rep( 0, length( uCoef ) ),
+   Amat = t( RMat ), bvec = - RMat %*% uCoef + rVec )
> rCoef <- minDistResult$solution + uCoef

```

We can calculate the distance elasticities of the inputs and the (negative) distance elasticities of the outputs based on the restricted coefficients by:

```

> elaIdfTl <- RMat %*% rCoef - rVec
> dat$eCapTLRidf <- elaIdfTl[ 1:nrow( dat ) ]
> dat$eLabTLRidf <- elaIdfTl[ ( nrow( dat ) + 1 ):( 2 * nrow( dat ) ) ]
> dat$eMatTLRidf <- elaIdfTl[ ( 2 * nrow( dat ) + 1 ):( 3 * nrow( dat ) ) ]
> dat$eApplesTLRidf <- -elaIdfTl[ ( 3 * nrow( dat ) + 1 ):( 4 * nrow( dat ) ) ]
> dat$eOtherTLRidf <- -elaIdfTl[ ( 4 * nrow( dat ) + 1 ):( 5 * nrow( dat ) ) ]
> summary( dat[ , c( "eCapTLRidf", "eLabTLRidf", "eMatTLRidf",
+   "eApplesTLRidf", "eOtherTLRidf" ) ] )

```

eCapTLRidf	eLabTLRidf	eMatTLRidf	eApplesTLRidf
Min. :0.00000	Min. :0.5105	Min. :0.1150	Min. : -0.2958
1st Qu.:0.06868	1st Qu.:0.6228	1st Qu.:0.1987	1st Qu.: -0.2229
Median :0.10036	Median :0.6596	Median :0.2434	Median : -0.1943
Mean :0.09914	Mean :0.6571	Mean :0.2437	Mean : -0.1864
3rd Qu.:0.13391	3rd Qu.:0.6950	3rd Qu.:0.2737	3rd Qu.: -0.1593
Max. :0.20573	Max. :0.7971	Max. :0.4360	Max. : 0.0000

eOtherTLRidf
Min. : -0.19191
1st Qu.: -0.15044
Median : -0.13416
Mean : -0.13396
3rd Qu.: -0.11993
Max. : -0.06537

We can see that all distance elasticities of the three are non-negative and all distance elasticities of the two outputs are non-positive.

The homogeneity condition is—of course—still fulfilled:

```

> range( dat$eCapTLRidf + dat$eLabTLRidf + dat$eMatTLRidf )

[1] 1 1

```

And the elasticities of scale can be calculated by:

```
> dat$eScaleTLRidf <- ( - ( dat$eApplesTLRidf + dat$eOtherTLRidf ) )^(-1)
> summary( dat$eScaleTLRidf )
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
2.050	2.741	2.992	3.418	3.515	15.298

9 Panel Data and Technological Change

Until now, we have only analyzed cross-sectional data, i.e. all observations refer to the same period of time. Hence, it was reasonable to assume that the same technology is available to all firms (observations). However, when analyzing time series data or panel data, i.e. when observations can originate from different time periods, different technologies might be available in the different time periods due to technological change. Hence, the state of the available technologies must be included as an explanatory variable in order to conduct a reasonable production analysis. Often, a time trend is used as a proxy for a gradually changing state of the available technologies.

We will demonstrate how to analyze production technologies with data from different time periods by using a balanced panel data set of annual data collected from 43 smallholder rice producers in the Tarlac region of the Philippines between 1990 and 1997. We loaded this data set (`riceProdPhil`) in section 1.4.2. As it does not contain information about the panel structure, we created a copy of the data set (`pdat`) that includes information on the panel structure.

9.1 Average production functions with technological change

In case of an applied production analysis with time-series data or panel data, usually the time (t) is included as additional explanatory variable in the production function:

$$y = f(x, t). \quad (9.1)$$

This function can be used to analyze how the time (t) affects the (available) production technology.

The *average* production technology (potentially depending on the time period) can be estimated from panel data sets by the OLS method (i.e. “pooled”) or by any of the usual panel data methods (e.g. fixed effects, random effects).

9.1.1 Cobb-Douglas production function with technological change

In case of a Cobb-Douglas production function, usually a linear time trend is added to account for technological change:

$$\ln y = \alpha_0 + \sum_i \alpha_i \ln x_i + \alpha_t t \quad (9.2)$$

Given this specification, the coefficient of the (linear) time trend can be interpreted as the rate of technological change per unit of the time variable t :

$$\alpha_t = \frac{\partial \ln y}{\partial t} = \frac{\partial \ln y}{\partial y} \frac{\partial y}{\partial t} \approx \frac{\frac{\Delta y}{y}}{\Delta t} \quad (9.3)$$

9.1.1.1 Pooled estimation of the Cobb-Douglas production function with technological change

The pooled estimation can be done by:

```
> riceCdTime <- lm( log( PROD ) ~ log( AREA ) + log( LABOR ) + log( NPK ) +
+       mYear, data = riceProdPhil )
> summary( riceCdTime )
```

Call:

```
lm(formula = log(PROD) ~ log(AREA) + log(LABOR) + log(NPK) +
    mYear, data = riceProdPhil)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.83351	-0.16006	0.05329	0.22110	0.86745

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-1.665096	0.248509	-6.700	8.68e-11	***
log(AREA)	0.333214	0.062403	5.340	1.71e-07	***
log(LABOR)	0.395573	0.066421	5.956	6.48e-09	***
log(NPK)	0.270847	0.041027	6.602	1.57e-10	***
mYear	0.010090	0.008007	1.260	0.208	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3299 on 339 degrees of freedom

Multiple R-squared: 0.86, Adjusted R-squared: 0.8583

F-statistic: 520.6 on 4 and 339 DF, p-value: < 2.2e-16

The estimation result indicates an annual rate of technical change of approximately 1%, but this is not statistically different from 0%, i.e., we cannot reject that there has been no technological change. The output elasticities are equal to the coefficients of the (logarithmic) input quantities and the elasticity of scale is almost exactly one (0.99963), indicating (approximately) constant returns to scale.

The command above can be simplified by using the pre-calculated logarithmic (and mean-scaled) quantities:

```
> riceCdTimeS <- lm( lProd ~ lArea + lLabor + lNpk + mYear, data = riceProdPhil )
> summary( riceCdTimeS )
```

Call:

```
lm(formula = lProd ~ lArea + lLabor + lNpk + mYear, data = riceProdPhil)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.83351	-0.16006	0.05329	0.22110	0.86745

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.015590	0.019325	-0.807	0.420
lArea	0.333214	0.062403	5.340	1.71e-07 ***
lLabor	0.395573	0.066421	5.956	6.48e-09 ***
lNpk	0.270847	0.041027	6.602	1.57e-10 ***
mYear	0.010090	0.008007	1.260	0.208

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3299 on 339 degrees of freedom

Multiple R-squared: 0.86, Adjusted R-squared: 0.8583

F-statistic: 520.6 on 4 and 339 DF, p-value: < 2.2e-16

The intercept has changed because of the mean-scaling of the input and output quantities but all slope parameters are unaffected by using the pre-calculated logarithmic (and mean-scaled) quantities:

```
> all.equal( coef( riceCdTime )[-1], coef( riceCdTimeS )[-1],
+   check.attributes = FALSE )
```

```
[1] TRUE
```

9.1.1.2 Panel data estimations of the Cobb-Douglas production function with technological change

The panel data estimation with fixed individual effects can be done by:

```
> riceCdTimeFe <- plm( lProd ~ lArea + lLabor + lNpk + mYear, data = pdat )
> summary( riceCdTimeFe )
```

Oneway (individual) effect Within Model

Call:

```
plm(formula = lProd ~ lArea + lLabor + lNpk + mYear, data = pdat)
```

Balanced Panel: n = 43, T = 8, N = 344

Residuals:

Min.	1st Qu.	Median	3rd Qu.	Max.
-1.585363	-0.156896	0.045606	0.177983	0.818168

Coefficients:

	Estimate	Std. Error	t-value	Pr(> t)
lArea	0.5607756	0.0785370	7.1403	7.195e-12 ***
lLabor	0.2549108	0.0690631	3.6910	0.0002657 ***
lNpk	0.1748528	0.0484684	3.6076	0.0003625 ***
mYear	0.0130908	0.0071824	1.8226	0.0693667 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares: 43.632

Residual Sum of Squares: 24.872

R-Squared: 0.42995

Adj. R-Squared: 0.34165

F-statistic: 56.0008 on 4 and 297 DF, p-value: < 2.22e-16

The estimation result indicates an annual rate of technical change of around 1.3%, which is statistically different from 0% (only) at the 10% significance level. While the elasticity of scale is still approximately one (1.00363), the estimated output elasticities notably differ from those obtained from the pooled regression. For instance, the output elasticity of land obtained from the pooled regression is 0.33 and obtained from the fixed-effects regression is 0.56. A potential reason for these differences is an endogeneity bias in the pooled regression: if there are unserved time-invariant variables that affect the output quantity and are correlated with some of the input quantities (e.g., the average soil quality at each farm is likely time-invariant and positively correlated with the error term and may also be positively correlated with the quantity of fertilizers), the pooled regression gives biased estimates, while the fixed effect estimation gives unbiased results, because the effects of the unserved time-invariant variables are absorbed in the fixed effects.

The panel data estimation with random individual effects can be done by:

```
> riceCdTimeRan <- plm( lProd ~ lArea + lLabor + lNpk + mYear, data = pdat,
```

```
+   model = "random" )
> summary( riceCdTimeRan )
```

```
Oneway (individual) effect Random Effect Model
(Swamy-Arora's transformation)
```

```
Call:
```

```
plm(formula = lProd ~ lArea + lLabor + lNpk + mYear, data = pdat,
     model = "random")
```

```
Balanced Panel: n = 43, T = 8, N = 344
```

```
Effects:
```

```
                var std.dev share
idiosyncratic 0.08375 0.28939   0.8
individual    0.02088 0.14451   0.2
theta: 0.4222
```

```
Residuals:
```

```
      Min.   1st Qu.   Median   3rd Qu.   Max.
-1.749676 -0.142852  0.048474  0.190936  0.851886
```

```
Coefficients:
```

```
                Estimate Std. Error z-value Pr(>|z|)
(Intercept) -0.0213044   0.0292268 -0.7289   0.4660
lArea        0.4563002   0.0662979  6.8826 5.878e-12 ***
lLabor       0.3190041   0.0647524  4.9265 8.371e-07 ***
lNpk         0.2268399   0.0426651  5.3168 1.056e-07 ***
mYear        0.0115453   0.0071921  1.6053   0.1084
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Total Sum of Squares:    117.05
```

```
Residual Sum of Squares: 29.195
```

```
R-Squared:    0.75058
```

```
Adj. R-Squared: 0.74764
```

```
Chisq: 1020.18 on 4 DF, p-value: < 2.22e-16
```

A variable-coefficient model for panel model with individual-specific coefficients can be estimated by:

```
> riceCdTimeVc <- pvcmm( lProd ~ lArea + lLabor + lNpk + mYear,
+   model = "within", data = pdat )
> summary( riceCdTimeVc )
```

Oneway (individual) effect No-pooling model

Call:

```
pvcmm(formula = lProd ~ lArea + lLabor + lNpk + mYear, data = pdat,
      model = "within")
```

Balanced Panel: n = 43, T = 8, N = 344

Residuals:

	Min.	1st Qu.	Median	3rd Qu.	Max.
	-0.817465311	-0.081971866	0.006676688	0.093984652	0.554124266

Coefficients:

(Intercept)	lArea	lLabor	lNpk
Min. : -3.8110	Min. : -5.2850	Min. : -2.72761	Min. : -1.3094
1st Qu.: -0.3006	1st Qu.: -0.4200	1st Qu.: -0.30989	1st Qu.: -0.1867
Median : 0.1145	Median : 0.6978	Median : 0.08778	Median : 0.1050
Mean : 0.1839	Mean : 0.5896	Mean : 0.06079	Mean : 0.1265
3rd Qu.: 0.5617	3rd Qu.: 1.8914	3rd Qu.: 0.61479	3rd Qu.: 0.3808
Max. : 3.7270	Max. : 4.7633	Max. : 1.75595	Max. : 1.7180
	NA's : 18		

mYear
Min. : -0.471049
1st Qu.: -0.044359
Median : -0.008111
Mean : -0.012327
3rd Qu.: 0.054743
Max. : 0.275875

Total Sum of Squares: 2861.8

Residual Sum of Squares: 8.9734

Multiple R-Squared: 0.99686

A pooled estimation can also be done by

```
> riceCdTimePool <- plm( lProd ~ lArea + lLabor + lNpk + mYear, data = pdat,
```

```
+    model = "pooling" )
```

This gives the same estimated coefficients as the model estimated by `lm`:

```
> all.equal( coef( riceCdTimeS ), coef( riceCdTimePool ) )
```

```
[1] TRUE
```

A Hausman test can be used to check the consistency of the random-effects estimator:

```
> phptest( riceCdTimeRan, riceCdTimeFe )
```

Hausman Test

```
data:  lProd ~ lArea + lLabor + lNpk + mYear
chisq = 14.62, df = 4, p-value = 0.005557
alternative hypothesis: one model is inconsistent
```

The Hausman test clearly shows that the random-effects estimator is inconsistent (due to correlation between the individual effects and the explanatory variables).

Now, we test the poolability of the model:

```
> pooltest( riceCdTimePool, riceCdTimeFe )
```

F statistic

```
data:  lProd ~ lArea + lLabor + lNpk + mYear
F = 3.4175, df1 = 42, df2 = 297, p-value = 4.038e-10
alternative hypothesis: unstability
```

```
> pooltest( riceCdTimePool, riceCdTimeVc )
```

F statistic

```
data:  lProd ~ lArea + lLabor + lNpk + mYear
F = 1.9113, df1 = 210, df2 = 129, p-value = 4.022e-05
alternative hypothesis: unstability
```

```
> pooltest( riceCdTimeFe, riceCdTimeVc )
```

F statistic

```
data:  lProd ~ lArea + lLabor + lNpk + mYear
F = 1.3605, df1 = 168, df2 = 129, p-value = 0.03339
alternative hypothesis: unstability
```

The pooled model (`riceCdTimePool`) is clearly rejected in favor of the model with fixed individual effects (`riceCdTimeFe`) and the variable-coefficient model (`riceCdTimeVc`). The model with fixed individual effects (`riceCdTimeFe`) is rejected in favor of the variable-coefficient model (`riceCdTimeVc`) at 5% significance level but not at 1% significance level.

9.1.2 Translog production function with constant and neutral technological change

A Translog production function that accounts for constant and neutral (unbiased) technological change has following specification:

$$\ln y = \alpha_0 + \sum_i \alpha_i \ln x_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij} \ln x_i \ln x_j + \alpha_t t \quad (9.4)$$

In this specification, the rate of technological change is

$$\frac{\partial \ln y}{\partial t} = \alpha_t \quad (9.5)$$

and the output elasticities are the same as in the time-invariant Translog production function (2.134):

$$\epsilon_i = \frac{\partial \ln y}{\partial \ln x_i} = \alpha_i + \sum_j \alpha_{ij} \ln x_j \quad (9.6)$$

In order to be able to interpret the first-order coefficients of the (logarithmic) input quantities (α_i) as output elasticities (ϵ_i) at the sample mean, we use the mean-scaled input quantities. We also use the mean-scaled output quantity in order to use the same variables as Coelli et al. (2005, p. 250).

9.1.2.1 Pooled estimation of the Translog production function with constant and neutral technological change

The following command estimates a Translog production function that can account for constant and neutral technical change:

```
> riceTlTime <- lm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) + mYear,
+   data = riceProdPhil )
> summary( riceTlTime )
```

Call:

```
lm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
    I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
    I(lArea * lNpk) + I(lLabor * lNpk) + mYear, data = riceProdPhil)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.52184	-0.18121	0.04356	0.22298	0.87019

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	0.013756	0.024645	0.558	0.57712
lArea	0.588097	0.085162	6.906	2.54e-11 ***
lLabor	0.191764	0.080876	2.371	0.01831 *
lNpk	0.197875	0.051605	3.834	0.00015 ***
I(0.5 * lArea^2)	-0.435547	0.247491	-1.760	0.07935 .
I(0.5 * lLabor^2)	-0.742242	0.303236	-2.448	0.01489 *
I(0.5 * lNpk^2)	0.020367	0.097907	0.208	0.83534
I(lArea * lLabor)	0.678647	0.216594	3.133	0.00188 **
I(lArea * lNpk)	0.063920	0.145613	0.439	0.66097
I(lLabor * lNpk)	-0.178286	0.138611	-1.286	0.19926
mYear	0.012682	0.007795	1.627	0.10468

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3184 on 333 degrees of freedom

Multiple R-squared: 0.8719, Adjusted R-squared: 0.868

F-statistic: 226.6 on 10 and 333 DF, p-value: < 2.2e-16

In the Translog production function that accounts for constant and neutral technological change, the monotonicity conditions are fulfilled at the sample mean and the estimated output elasticities of land, labor and fertilizer are 0.588, 0.192, and 0.198, respectively, at the sample mean. The estimated (constant) annual rate of technological progress is around 1.3%.

Conduct a Wald test to test whether the Translog production function outperforms the Cobb-Douglas production function:

```
> library( "lmtest" )
> waldtest( riceCdTimeS, riceTlTime )
```

Wald test

Model 1: lProd ~ lArea + lLabor + lNpk + mYear

```
Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
lNpk) + mYear
```

```

      Res.Df Df       F    Pr(>F)
1         339
2         333   6 5.1483 4.451e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

The Cobb-Douglas specification is clearly rejected in favor of the Translog specification for the pooled estimation.

9.1.2.2 Panel-data estimations of the Translog production function with constant and neutral technological change

The following command estimates a Translog production function that can account for constant and neutral technical change with fixed individual effects:

```

> riceTlTimeFe <- plm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) + mYear,
+   data = pdat, model = "within" )
> summary( riceTlTimeFe )

```

Oneway (individual) effect Within Model

Call:

```

plm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
     I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
     I(lArea * lNpk) + I(lLabor * lNpk) + mYear, data = pdat,
     model = "within")

```

Balanced Panel: n = 43, T = 8, N = 344

Residuals:

	Min.	1st Qu.	Median	3rd Qu.	Max.
	-1.012473	-0.144573	0.019129	0.167687	0.745525

Coefficients:

	Estimate	Std. Error	t-value	Pr(> t)
lArea	0.5828102	0.1173298	4.9673	1.16e-06 ***
lLabor	0.0473355	0.0848594	0.5578	0.577402
lNpk	0.1211928	0.0610114	1.9864	0.047927 *
I(0.5 * lArea^2)	-0.8543901	0.2861292	-2.9860	0.003067 **


```

I(0.5 * lLabor^2) -0.6217163  0.2935429 -2.1180 0.035025 *
I(0.5 * lNpk^2)    0.0429446  0.0987119  0.4350 0.663849
I(lArea * lLabor)  0.5867063  0.2125686  2.7601 0.006145 **
I(lArea * lNpk)    0.1167509  0.1461380  0.7989 0.424995
I(lLabor * lNpk)   -0.2371219  0.1268671 -1.8691 0.062619 .
mYear              0.0165309  0.0069206  2.3887 0.017547 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

Total Sum of Squares:    43.632
Residual Sum of Squares: 21.912
R-Squared:               0.49781
Adj. R-Squared: 0.40807
F-statistic: 28.8456 on 10 and 291 DF, p-value: < 2.22e-16

```

And the panel data estimation with random individual effects can be done by:

```

> riceTlTimeRan <- plm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) + mYear,
+   data = pdat, model = "random" )
> summary( riceTlTimeRan )

```

Oneway (individual) effect Random Effect Model
(Swamy-Arora's transformation)

Call:

```

plm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
      I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
      I(lArea * lNpk) + I(lLabor * lNpk) + mYear, data = pdat,
      model = "random")

```

Balanced Panel: n = 43, T = 8, N = 344

Effects:

```

              var std.dev share
idiosyncratic 0.07530 0.27440  0.79
individual    0.01997 0.14130  0.21
theta: 0.434

```

Residuals:

9 Panel Data and Technological Change

Min.	1st Qu.	Median	3rd Qu.	Max.
-1.393176	-0.162097	0.045567	0.184209	0.798214

Coefficients:

	Estimate	Std. Error	z-value	Pr(> z)
(Intercept)	0.0213211	0.0347371	0.6138	0.539357
lArea	0.6831045	0.0922069	7.4084	1.278e-13 ***
lLabor	0.0974523	0.0804060	1.2120	0.225511
lNpk	0.1708366	0.0546853	3.1240	0.001784 **
I(0.5 * lArea^2)	-0.4275328	0.2468086	-1.7322	0.083230 .
I(0.5 * lLabor^2)	-0.6367899	0.2872825	-2.2166	0.026651 *
I(0.5 * lNpk^2)	0.0307547	0.0957745	0.3211	0.748122
I(lArea * lLabor)	0.5666863	0.2059076	2.7521	0.005921 **
I(lArea * lNpk)	0.1037657	0.1421739	0.7299	0.465481
I(lLabor * lNpk)	-0.2055786	0.1277476	-1.6093	0.107560
mYear	0.0142202	0.0070184	2.0261	0.042752 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares: 114.08

Residual Sum of Squares: 26.624

R-Squared: 0.76662

Adj. R-Squared: 0.75961

Chisq: 1093.86 on 10 DF, p-value: < 2.22e-16

The Translog production function cannot be estimated by a variable-coefficient model for panel model with our data set, because the number of time periods in the data set is smaller than the number of the coefficients.

A pooled estimation can be done by

```
> riceTlTimePool <- plm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) + mYear,
+   data = pdat, model = "pooling" )
> summary(riceTlTimePool)
```

Pooling Model

Call:

```
plm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
     I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
```

9 Panel Data and Technological Change

```
I(lArea * lNpk) + I(lLabor * lNpk) + mYear, data = pdat,
model = "pooling")
```

Balanced Panel: n = 43, T = 8, N = 344

Residuals:

Min.	1st Qu.	Median	3rd Qu.	Max.
-1.521838	-0.181205	0.043555	0.222979	0.870190

Coefficients:

	Estimate	Std. Error	t-value	Pr(> t)
(Intercept)	0.0137557	0.0246454	0.5581	0.5771201
lArea	0.5880972	0.0851622	6.9056	2.542e-11 ***
lLabor	0.1917638	0.0808764	2.3711	0.0183052 *
lNpk	0.1978747	0.0516045	3.8344	0.0001505 ***
I(0.5 * lArea^2)	-0.4355466	0.2474913	-1.7598	0.0793520 .
I(0.5 * lLabor^2)	-0.7422415	0.3032362	-2.4477	0.0148916 *
I(0.5 * lNpk^2)	0.0203673	0.0979072	0.2080	0.8353358
I(lArea * lLabor)	0.6786472	0.2165937	3.1333	0.0018822 **
I(lArea * lNpk)	0.0639200	0.1456135	0.4390	0.6609677
I(lLabor * lNpk)	-0.1782859	0.1386111	-1.2862	0.1992559
mYear	0.0126820	0.0077947	1.6270	0.1046801

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares: 263.52

Residual Sum of Squares: 33.761

R-Squared: 0.87189

Adj. R-Squared: 0.86804

F-statistic: 226.623 on 10 and 333 DF, p-value: < 2.22e-16

This gives the same estimated coefficients as the model estimated by `lm`:

```
> all.equal( coef( riceTlTime ), coef( riceTlTimePool ) )
```

```
[1] TRUE
```

A Hausman test can be used to check the consistency of the random-effects estimator:

```
> phptest( riceTlTimeRan, riceTlTimeFe )
```

Hausman Test

```
data:  lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + ...
chisq = 66.071, df = 10, p-value = 2.528e-10
alternative hypothesis: one model is inconsistent
```

The Hausman test clearly rejects the consistency of the random-effects estimator.

The following command tests the poolability of the model:

```
> pooltest( riceTlTimePool, riceTlTimeFe )
```

F statistic

```
data:  lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + ...
F = 3.7469, df1 = 42, df2 = 291, p-value = 1.525e-11
alternative hypothesis: unstability
```

The pooled model (`riceCdTimePool`) is clearly rejected in favor of the model with fixed individual effects (`riceCdTimeFe`), i.e. the individual effects are statistically significant.

The following commands test if the fit of Translog specification is significantly better than the fit of the Cobb-Douglas specification:

```
> waldtest( riceCdTimeFe, riceTlTimeFe )
```

Wald test

Model 1: $\text{lProd} \sim \text{lArea} + \text{lLabor} + \text{lNpk} + \text{mYear}$

Model 2: $\text{lProd} \sim \text{lArea} + \text{lLabor} + \text{lNpk} + I(0.5 * \text{lArea}^2) + I(0.5 * \text{lLabor}^2) + I(0.5 * \text{lNpk}^2) + I(\text{lArea} * \text{lLabor}) + I(\text{lArea} * \text{lNpk}) + I(\text{lLabor} * \text{lNpk}) + \text{mYear}$

	Res.Df	Df	Chisq	Pr(>Chisq)
1	297			
2	291	6	39.321	6.191e-07 ***

1 297

2 291 6 39.321 6.191e-07 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> waldtest( riceCdTimeRan, riceTlTimeRan )
```

Wald test

Model 1: $\text{lProd} \sim \text{lArea} + \text{lLabor} + \text{lNpk} + \text{mYear}$

Model 2: $\text{lProd} \sim \text{lArea} + \text{lLabor} + \text{lNpk} + I(0.5 * \text{lArea}^2) + I(0.5 * \text{lLabor}^2) + I(0.5 * \text{lNpk}^2) + I(\text{lArea} * \text{lLabor}) + I(\text{lArea} * \text{lNpk}) + I(\text{lLabor} * \text{lNpk}) + \text{mYear}$

```

Res.Df Df  Chisq Pr(>Chisq)
1      339
2      333  6 30.077    3.8e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> waldtest( riceCdTimePool, riceTlTimePool )

Wald test

Model 1: lProd ~ lArea + lLabor + lNpk + mYear
Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
      I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
      lNpk) + mYear
Res.Df Df  Chisq Pr(>Chisq)
1      339
2      333  6 30.89    2.66e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

The Cobb-Douglas functional form is rejected in favor of the Translog functional for all three panel-specifications that we estimated above. The Wald test for the pooled model differs from the Wald test that we did in section 9.1.2.1, because `waldtest` by default uses a finite sample F statistic for models estimated by `lm` but uses a large sample Chi-squared statistic for models estimated by `plm`. The test statistic used by `waldtest` can be specified by argument `test`.

9.1.3 Translog production function with non-constant and non-neutral technological change

Technological change is not always constant and is not always neutral (unbiased). Therefore, it might be more suitable to estimate a production function that can account for increasing or decreasing rates of technological change as well as biased (e.g. labor saving) technological change. This can be done by including a quadratic time trend and interaction terms between time and input quantities:

$$\ln y = \alpha_0 + \sum_i \alpha_i \ln x_i + \frac{1}{2} \sum_i \sum_j \alpha_{ij} \ln x_i \ln x_j + \alpha_t t + \sum_i \alpha_{ti} t \ln x_i + \frac{1}{2} \alpha_{tt} t^2 \quad (9.7)$$

In this specification, the rate of technological change depends on the input quantities and the time period:

$$\frac{\partial \ln y}{\partial t} = \alpha_t + \sum_i \alpha_{ti} \ln x_i + \alpha_{tt} t \quad (9.8)$$

and the output elasticities might change over time:

$$\epsilon_i = \frac{\partial \ln y}{\partial \ln x_i} = \alpha_i + \sum_j \alpha_{ij} \ln x_j + \alpha_{ti} t. \quad (9.9)$$

9.1.3.1 Pooled estimation of a translog production function with non-constant and non-neutral technological change

The following command estimates a Translog production function that can account for non-constant rates of technological change as well as biased technological change:

```
> riceTlTimeNn <- lm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) +
+   mYear + I( mYear * lArea ) + I( mYear * lLabor ) + I( mYear * lNpk ) +
+   I( 0.5 * mYear^2 ), data = riceProdPhil )
> summary( riceTlTimeNn )
```

Call:

```
lm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
    I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
    I(lArea * lNpk) + I(lLabor * lNpk) + mYear + I(mYear * lArea) +
    I(mYear * lLabor) + I(mYear * lNpk) + I(0.5 * mYear^2), data = riceProdPhil)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.54976	-0.17245	0.04623	0.21624	0.87075

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	0.001255	0.031934	0.039	0.96867
lArea	0.579682	0.085892	6.749	6.73e-11 ***
lLabor	0.187505	0.081359	2.305	0.02181 *
lNpk	0.207193	0.052130	3.975	8.67e-05 ***
I(0.5 * lArea^2)	-0.468372	0.265363	-1.765	0.07849 .
I(0.5 * lLabor^2)	-0.688940	0.308046	-2.236	0.02599 *
I(0.5 * lNpk^2)	0.055993	0.099848	0.561	0.57533
I(lArea * lLabor)	0.676833	0.223271	3.031	0.00263 **
I(lArea * lNpk)	0.082374	0.151312	0.544	0.58654
I(lLabor * lNpk)	-0.226885	0.145568	-1.559	0.12005
mYear	0.008746	0.008513	1.027	0.30497
I(mYear * lArea)	0.003482	0.028075	0.124	0.90136

```

I(mYear * lLabor)  0.034661    0.029480    1.176  0.24054
I(mYear * lNpk)    -0.037964    0.020355   -1.865  0.06305 .
I(0.5 * mYear^2)   0.007611    0.007954    0.957  0.33933

```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.3184 on 329 degrees of freedom
```

```
Multiple R-squared:  0.8734,      Adjusted R-squared:  0.868
```

```
F-statistic: 162.2 on 14 and 329 DF,  p-value: < 2.2e-16
```

We conduct a Wald test to test whether the Translog production function with non-constant and non-neutral technological change outperforms the Cobb-Douglas production function and the Translog production function with constant and neutral technological change:

```
> waldtest( riceCdTimeS, riceTlTimeNn )
```

```
Wald test
```

```
Model 1: lProd ~ lArea + lLabor + lNpk + mYear
```

```
Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear *
lNpk) + I(0.5 * mYear^2)
```

```
Res.Df Df      F Pr(>F)
```

```
1      339
```

```
2      329 10 3.488 0.00022 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> waldtest( riceTlTime, riceTlTimeNn )
```

```
Wald test
```

```
Model 1: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
lNpk) + mYear
```

```
Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear *
lNpk) + I(0.5 * mYear^2)
```

```
Res.Df Df      F Pr(>F)
```

```

1    333
2    329  4 0.9976 0.4089

```

The fit of the Translog specification with non-constant and non-neutral technological change is significantly better than the fit of the Cobb-Douglas specification but it is not significantly better than the fit of the Translog specification with constant and neutral technological change.

In order to simplify the calculation of the output elasticities (with equation 9.9) and the annual rates of technological change (with equation 9.8), we create shortcuts for the estimated coefficients:

```

> a1 <- coef( riceTlTimeNn )[ "lArea" ]
> a2 <- coef( riceTlTimeNn )[ "lLabor" ]
> a3 <- coef( riceTlTimeNn )[ "lNpk" ]
> at <- coef( riceTlTimeNn )[ "mYear" ]
> a11 <- coef( riceTlTimeNn )[ "I(0.5 * lArea^2)" ]
> a22 <- coef( riceTlTimeNn )[ "I(0.5 * lLabor^2)" ]
> a33 <- coef( riceTlTimeNn )[ "I(0.5 * lNpk^2)" ]
> att <- coef( riceTlTimeNn )[ "I(0.5 * mYear^2)" ]
> a12 <- a21 <- coef( riceTlTimeNn )[ "I(lArea * lLabor)" ]
> a13 <- a31 <- coef( riceTlTimeNn )[ "I(lArea * lNpk)" ]
> a23 <- a32 <- coef( riceTlTimeNn )[ "I(lLabor * lNpk)" ]
> a1t <- at1 <- coef( riceTlTimeNn )[ "I(mYear * lArea)" ]
> a2t <- at2 <- coef( riceTlTimeNn )[ "I(mYear * lLabor)" ]
> a3t <- at3 <- coef( riceTlTimeNn )[ "I(mYear * lNpk)" ]

```

Now, we can use the following commands to calculate the partial output elasticities:

```

> riceProdPhil$eArea <- with( riceProdPhil,
+   a1 + a11 * lArea + a12 * lLabor + a13 * lNpk + a1t * mYear )
> riceProdPhil$eLabor <- with( riceProdPhil,
+   a2 + a21 * lArea + a22 * lLabor + a23 * lNpk + a2t * mYear )
> riceProdPhil$eNpk <- with( riceProdPhil,
+   a3 + a31 * lArea + a32 * lLabor + a33 * lNpk + a3t * mYear )

```

We can calculate the elasticity of scale by taken the sum over all partial output elasticities:

```

> riceProdPhil$eScale <- with( riceProdPhil, eArea + eLabor + eNpk )

```

We can visualize (the variation of) the output elasticities and the elasticity of scale with histograms:

```

> hist( riceProdPhil$eArea, 15 )
> hist( riceProdPhil$eLabor, 15 )
> hist( riceProdPhil$eNpk, 15 )
> hist( riceProdPhil$eScale, 15 )

```

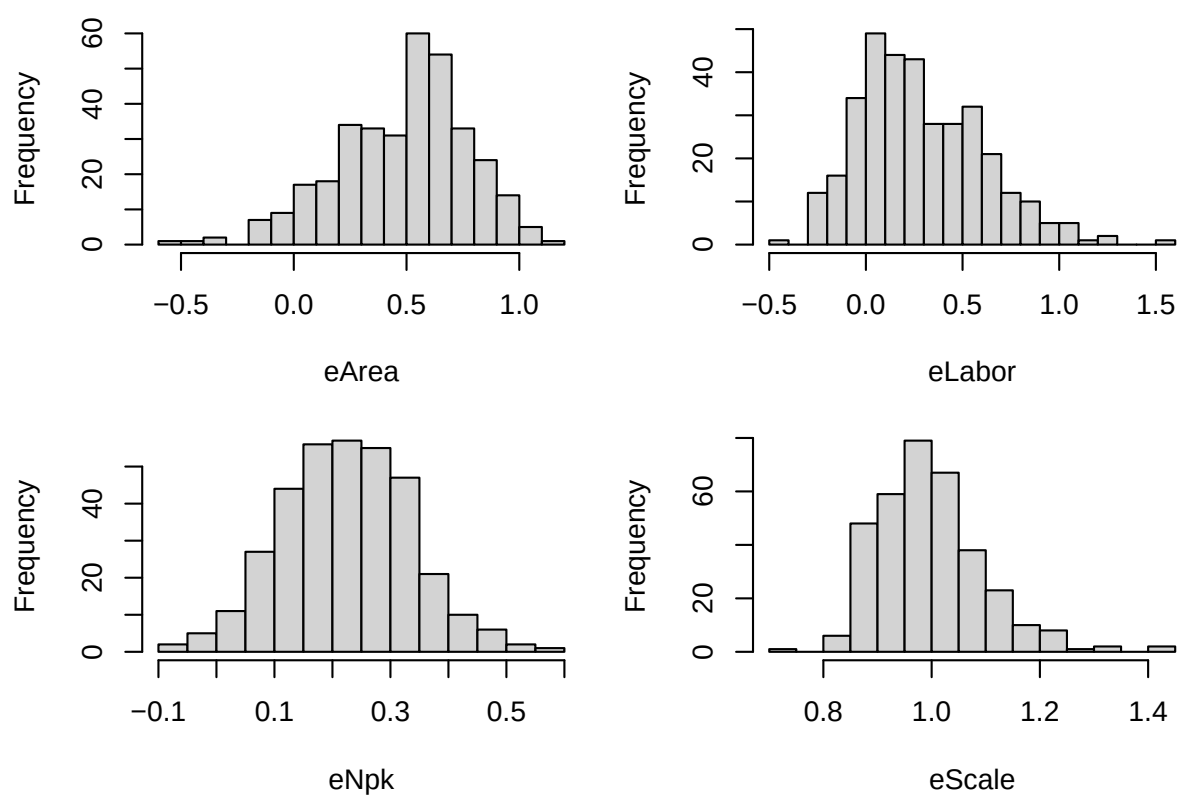



Figure 9.1: Output elasticities and elasticities of scale

The resulting graphs are shown in figure 9.1. If the firms increase the land area by one percent, the output of most firms will increase by around 0.6 percent. If the firms increase labor input by one percent, the output of most firms will increase by around 0.2 percent. If the firms increase fertilizer input by one percent, the output of most firms will increase by around 0.25 percent. If the firms increase all input quantities by one percent, the output of most firms will also increase by around 1 percent. These graphs also show that the monotonicity condition is not fulfilled for some observations:

```
> table( riceProdPhil$eArea >= 0 )

FALSE  TRUE
   20   324

> table( riceProdPhil$eLabor >= 0 )

FALSE  TRUE
   63   281

> table( riceProdPhil$eNpk >= 0 )

FALSE  TRUE
    7   337

> riceProdPhil$monoTl <- with( riceProdPhil, eArea >= 0 & eLabor >= 0 & eNpk >= 0 )
> table( riceProdPhil$monoTl )

FALSE  TRUE
   85   259
```

20 firms have a negative output elasticity of the land area, 63 firms have a negative output elasticity of labor, and 7 firms have a negative output elasticity of fertilizers. In total the monotonicity condition is not fulfilled at 85 out of 344 observations. Although the monotonicity conditions are fulfilled for a large part of firms in our data set, these frequent violations indicate a possible model misspecification.

We can use the following command to calculate the annual rates of technological change:

```
> riceProdPhil$tc <- with( riceProdPhil,
+   at + at1 * lArea + at2 * lLabor + at3 * lNpk + att * mYear )
```

We can visualize (the variation of) the annual rates of technological change with a histogram:

```
> hist( riceProdPhil$tc, 15 )
```

The resulting graph is shown in figure 9.2. For most observations, the annual rate of technological change was between 0% and 3%.

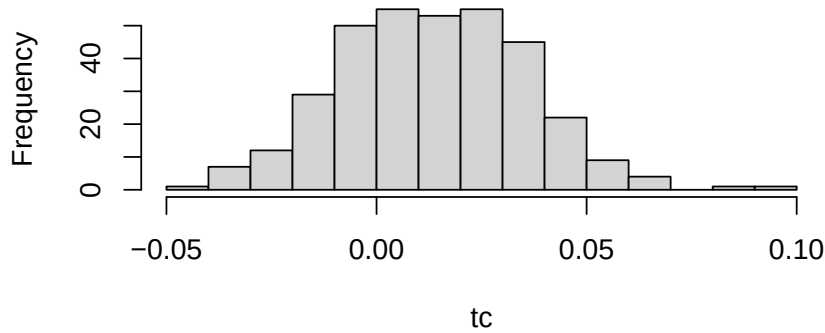


Figure 9.2: Annual rates of technological change

9.1.3.2 Panel-data estimations of a Translog production function with non-constant and non-neutral technological change

The panel data estimation with fixed individual effects can be done by:

```
> riceTlTimeNnFe <- plm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) +
+   mYear + I( mYear * lArea ) + I( mYear * lLabor ) + I( mYear * lNpk ) +
+   I( 0.5 * mYear^2 ), data = pdat )
> summary( riceTlTimeNnFe )
```

Oneway (individual) effect Within Model

Call:

```
plm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
     I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
     I(lArea * lNpk) + I(lLabor * lNpk) + mYear + I(mYear * lArea) +
     I(mYear * lLabor) + I(mYear * lNpk) + I(0.5 * mYear^2), data = pdat)
```

Balanced Panel: n = 43, T = 8, N = 344

Residuals:

Min.	1st Qu.	Median	3rd Qu.	Max.
-1.014803	-0.143180	0.017469	0.166852	0.749361

Coefficients:

	Estimate	Std. Error	t-value	Pr(> t)
lArea	0.5857359	0.1191164	4.9173	1.479e-06 ***
lLabor	0.0336966	0.0869044	0.3877	0.698494

9 Panel Data and Technological Change

```

lNpk          0.1276970  0.0623919  2.0467  0.041599 *
I(0.5 * lArea^2) -0.8588620  0.2952677 -2.9088  0.003912 **
I(0.5 * lLabor^2) -0.6154568  0.2979094 -2.0659  0.039733 *
I(0.5 * lNpk^2)  0.0673038  0.1014542  0.6634  0.507613
I(lArea * lLabor)  0.6016538  0.2164953  2.7791  0.005811 **
I(lArea * lNpk)   0.1205064  0.1549834  0.7775  0.437479
I(lLabor * lNpk) -0.2660519  0.1353699 -1.9654  0.050336 .
mYear           0.0148796  0.0076143  1.9542  0.051654 .
I(mYear * lArea)  0.0105012  0.0270130  0.3887  0.697752
I(mYear * lLabor) 0.0230156  0.0286066  0.8046  0.421743
I(mYear * lNpk)   -0.0279542  0.0199045 -1.4044  0.161277
I(0.5 * mYear^2)  0.0058526  0.0069948  0.8367  0.403458

```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares: 43.632

Residual Sum of Squares: 21.733

R-Squared: 0.50189

Adj. R-Squared: 0.40469

F-statistic: 20.6552 on 14 and 287 DF, p-value: < 2.22e-16

And the panel data estimation with random individual effects can be done by:

```

> riceTlTimeNnRan <- plm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) +
+   mYear + I( mYear * lArea ) + I( mYear * lLabor ) + I( mYear * lNpk ) +
+   I( 0.5 * mYear^2 ), data = pdat, model = "random" )
> summary( riceTlTimeNnRan )

```

Oneway (individual) effect Random Effect Model

(Swamy-Arora's transformation)

Call:

```

plm(formula = lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) +
    I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) +
    I(lArea * lNpk) + I(lLabor * lNpk) + mYear + I(mYear * lArea) +
    I(mYear * lLabor) + I(mYear * lNpk) + I(0.5 * mYear^2), data = pdat,
    model = "random")

```

Balanced Panel: n = 43, T = 8, N = 344

Effects:

```

                var std.dev share
idiosyncratic 0.07573 0.27518 0.796
individual    0.01941 0.13933 0.204
theta: 0.4275

```

Residuals:

```

      Min.    1st Qu.    Median    3rd Qu.    Max.
-1.392702 -0.162022  0.045553  0.180242  0.790301

```

Coefficients:

```

                Estimate Std. Error z-value Pr(>|z|)
(Intercept)      0.0101183  0.0389961  0.2595  0.795272
lArea             0.6809764  0.0930789  7.3161 2.552e-13 ***
lLabor           0.0865327  0.0813309  1.0640  0.287348
lNpk              0.1800677  0.0554226  3.2490  0.001158 **
I(0.5 * lArea^2) -0.4749163  0.2627102 -1.8078  0.070644 .
I(0.5 * lLabor^2) -0.6146891  0.2907148 -2.1144  0.034481 *
I(0.5 * lNpk^2)   0.0614961  0.0980315  0.6273  0.530456
I(lArea * lLabor) 0.5916989  0.2113078  2.8002  0.005107 **
I(lArea * lNpk)   0.1224789  0.1488815  0.8227  0.410701
I(lLabor * lNpk) -0.2531048  0.1350400 -1.8743  0.060890 .
mYear             0.0116511  0.0077140  1.5104  0.130947
I(mYear * lArea)  0.0028675  0.0265731  0.1079  0.914068
I(mYear * lLabor) 0.0355897  0.0279156  1.2749  0.202343
I(mYear * lNpk)   -0.0344049  0.0195392 -1.7608  0.078269 .
I(0.5 * mYear^2)  0.0069525  0.0071510  0.9722  0.330936

```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares: 115.71

Residual Sum of Squares: 26.417

R-Squared: 0.77169

Adj. R-Squared: 0.76198

Chisq: 1112.04 on 14 DF, p-value: < 2.22e-16

The Translog production function cannot be estimated by a variable-coefficient model for panel model with our data set, because the number of time periods in the data set is smaller than the number of the coefficients.

A pooled estimation can be done by

```
> riceTlTimeNnPool <- plm( lProd ~ lArea + lLabor + lNpk +
+   I( 0.5 * lArea^2 ) + I( 0.5 * lLabor^2 ) + I( 0.5 * lNpk^2 ) +
+   I( lArea * lLabor ) + I( lArea * lNpk ) + I( lLabor * lNpk ) +
+   mYear + I( mYear * lArea ) + I( mYear * lLabor ) + I( mYear * lNpk ) +
+   I( 0.5 * mYear^2 ), data = pdat, model = "pooling" )
```

This gives the same estimated coefficients as the model estimated by `lm`:

```
> all.equal( coef( riceTlTimeNn ), coef( riceTlTimeNnPool ) )

[1] TRUE
```

A Hausman test can be used to check the consistency of the random-effects estimator:

```
> phptest( riceTlTimeNnRan, riceTlTimeNnFe )
```

Hausman Test

```
data:  lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + ...
chisq = 21.731, df = 14, p-value = 0.08432
alternative hypothesis: one model is inconsistent
```

The Hausman test rejects the consistency of the random-effects estimator at the 10% significance level but it cannot reject the consistency of the random-effects estimator at the 5% significance level.

The following command tests the poolability of the model:

```
> pooltest( riceTlTimeNnPool, riceTlTimeNnFe )
```

F statistic

```
data:  lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + ...
F = 3.6544, df1 = 42, df2 = 287, p-value = 4.266e-11
alternative hypothesis: unstability
```

The pooled model (`riceCdTimePool`) is clearly rejected in favor of the model with fixed individual effects (`riceCdTimeFe`), i.e. the individual effects are statistically significant.

The following commands test if the fit of Translog specification is significantly better than the fit of the Cobb-Douglas specification:

```
> waldtest( riceTlTimeNnFe, riceCdTimeFe )
```

Wald test

Model 1: $lProd \sim lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor * lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear * lNpk) + I(0.5 * mYear^2)$

Model 2: $lProd \sim lArea + lLabor + lNpk + mYear$

	Res.Df	Df	Chisq	Pr(>Chisq)
1	287			
2	297	-10	41.45	9.392e-06 ***

1 287

2 297 -10 41.45 9.392e-06 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> waldtest(riceTlTimeNnRan, riceCdTimeRan)

Wald test

Model 1: $lProd \sim lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor * lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear * lNpk) + I(0.5 * mYear^2)$

Model 2: $lProd \sim lArea + lLabor + lNpk + mYear$

	Res.Df	Df	Chisq	Pr(>Chisq)
1	329			
2	339	-10	33.666	0.0002103 ***

1 329

2 339 -10 33.666 0.0002103 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> waldtest(riceTlTimeNnPool, riceCdTimePool)

Wald test

Model 1: $lProd \sim lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) + I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor * lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear * lNpk) + I(0.5 * mYear^2)$

Model 2: $lProd \sim lArea + lLabor + lNpk + mYear$

	Res.Df	Df	Chisq	Pr(>Chisq)
1	329			
2	339	-10	34.88	0.0001309 ***

1 329

2 339 -10 34.88 0.0001309 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Finally, we test whether the fit of Translog specification with non-constant and non-neutral technological change is significantly better than the fit of Translog specification with constant and neutral technological change:

```
> waldtest( riceTlTimeNnFe, riceTlTimeFe )
```

Wald test

```
Model 1: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
  I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
  lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear *
  lNpk) + I(0.5 * mYear^2)
```

```
Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
  I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
  lNpk) + mYear
```

	Res.Df	Df	Chisq	Pr(>Chisq)
1	287			
2	291	-4	2.3512	0.6715

```
> waldtest( riceTlTimeNnRan, riceTlTimeRan )
```

Wald test

```
Model 1: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
  I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
  lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear *
  lNpk) + I(0.5 * mYear^2)
```

```
Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
  I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
  lNpk) + mYear
```

	Res.Df	Df	Chisq	Pr(>Chisq)
1	329			
2	333	-4	3.6633	0.4535

```
> waldtest( riceTlTimeNnPool, riceTlTimePool )
```

Wald test

```
Model 1: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
  I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
  lNpk) + mYear + I(mYear * lArea) + I(mYear * lLabor) + I(mYear *
  lNpk) + I(0.5 * mYear^2)
```



```

Model 2: lProd ~ lArea + lLabor + lNpk + I(0.5 * lArea^2) + I(0.5 * lLabor^2) +
          I(0.5 * lNpk^2) + I(lArea * lLabor) + I(lArea * lNpk) + I(lLabor *
          lNpk) + mYear
Res.Df Df  Chisq Pr(>Chisq)
1      329
2      333 -4  3.9905      0.4073

```

The tests indicate that the fit of Translog specification with constant and neutral technological change is not significantly worse than the fit of Translog specification with non-constant and non-neutral technological change.

The difference between the Wald tests for the pooled model and the Wald test that we did in section 9.1.3.1 is explained at the end of section 9.1.2.2.

9.2 Frontier production functions with technological change

The *frontier* production technology can be estimated by many different specifications of the stochastic frontier model. We will focus on four specifications that are all nested in the general specification:

$$\ln y_{kt} = \ln f(x_{kt}, t, k) - u_{kt} + v_{kt}, \quad (9.10)$$

where the subscript $k = 1, \dots, K$ indicates the firm, $t = 1, \dots, T$ indicates the time period, and all other variables are defined as before. We will apply the following four model specifications:

1. the same frontier for all firms, i.e., $f(x_{kt}, t, k) = f(x_{kt}, t) \forall k, t$, and time-invariant individual efficiencies, i.e., $u_{kt} = u_k \forall k, t$, which means that each firm has an individual fixed efficiency that remains constant over time;
2. the same frontier for all firms, i.e., $f(x_{kt}, t, k) = f(x_{kt}, t) \forall k, t$, and time-variant individual efficiencies, with $u_{kt} = u_k \exp(-\eta (t - T)) \forall k, t$, which means that each firm has an individual efficiency and the inefficiency terms u_{kt} of all firms can change over time with the same rate (and in the same direction) as indicated by the additional coefficient η ;
3. the same frontier for all firms, i.e., $f(x_{kt}, t, k) = f(x_{kt}, t) \forall k, t$, and observation-specific efficiencies, i.e., no restrictions on u_{kt} , which means that the efficiency term of each observation is estimated independently from the other efficiencies of the firm so that basically the panel structure of the data is ignored; and
4. a different frontier for each firm, i.e., $f(x_{kt}, t, k) = f(x_{kt}, t) e^{\delta_k} \forall k, t$ (i.e., individual fixed effects in the frontier), and observation-specific efficiencies, i.e., no restrictions on u_{kt} .

9.2.1 Cobb-Douglas production frontier with technological change

We will use the specification in equation (9.2).

9.2.1.1 Time-invariant individual efficiencies

We start with estimating a Cobb-Douglas production frontier with time-invariant individual efficiencies. The following commands estimate two Cobb-Douglas production frontiers with time-invariant individual efficiencies, the first does not account for technological change, while the second does:

```
> riceCdSfaInv <- sfa( lProd ~ lArea + lLabor + lNpk, data = pdat )
> summary( riceCdSfaInv )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 9 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	0.182636	0.035340	5.1680	2.366e-07	***
lArea	0.453900	0.064382	7.0501	1.788e-12	***
lLabor	0.288922	0.063853	4.5248	6.045e-06	***
lNpk	0.227542	0.040644	5.5983	2.164e-08	***
sigmaSq	0.155377	0.024144	6.4354	1.232e-10	***
gamma	0.464317	0.088270	5.2602	1.439e-07	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -86.43042

panel data

number of cross-sections = 43

number of time periods = 8

total number of observations = 344

thus there are 0 observations not in the panel

mean efficiency: 0.8187935

```
> riceCdTimeSfaInv <- sfa( lProd ~ lArea + lLabor + lNpk + mYear, data = pdat )
> summary( riceCdTimeSfaInv )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 11 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	0.1832751	0.0345895	5.2986	1.167e-07	***
lArea	0.4625174	0.0644245	7.1792	7.011e-13	***
lLabor	0.3029415	0.0641323	4.7237	2.316e-06	***
lNpk	0.2098907	0.0418709	5.0128	5.364e-07	***
mYear	0.0116003	0.0071758	1.6166	0.106	
sigmaSq	0.1556806	0.0242951	6.4079	1.475e-10	***
gamma	0.4706143	0.0869549	5.4122	6.227e-08	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -85.0743

panel data

number of cross-sections = 43

number of time periods = 8

total number of observations = 344

thus there are 0 observations not in the panel

mean efficiency: 0.8176333

In the Cobb-Douglas production frontier that accounts for technological change, the monotonicity conditions are globally fulfilled and the (constant) output elasticities of land, labor and fertilizer are 0.463, 0.303, and 0.21, respectively. The estimated (constant) annual rate of technological progress is around 1.2%. However, both the *t*-test for the coefficient of the time trend and a likelihood ratio test give rise to doubts whether the production technology indeed changes over time (P-values around 10%):

```
> lrtest( riceCdTimeSfaInv, riceCdSfaInv )
```

Likelihood ratio test

Model 1: riceCdTimeSfaInv

Model 2: riceCdSfaInv

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	7	-85.074			

```
2    6 -86.430 -1 2.7122    0.09958 .
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Further likelihood ratio tests show that OLS models are clearly rejected in favor of the corresponding stochastic frontier models (no matter whether the production frontier accounts for technological change or not):

```
> lrtest( riceCdSfaInv )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Error Components Frontier (ECF)

```
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -104.91
2    6 -86.43  1 36.953  6.051e-10 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> lrtest( riceCdTimeSfaInv )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)

Model 2: Error Components Frontier (ECF)

```
  #Df   LogLik Df  Chisq Pr(>Chisq)
1    6 -104.103
2    7 -85.074  1 38.057  3.434e-10 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This model estimates only a single efficiency estimate for each of the 43 firms. Hence, the vector returned by the `efficiencies` method only has 43 elements by default:

```
> length( efficiencies( riceCdSfaInv ) )
```

```
[1] 43
```

One can obtain the efficiency estimates for each observation by setting argument `asInData` equal to `TRUE`:

```
> pdat$effCdInv <- efficiencies( riceCdSfaInv, asInData = TRUE )
```

Please note that the efficiency estimates for each firm still do not vary between time periods.

9.2.1.2 Time-variant individual efficiencies

Now we estimate a Cobb-Douglas production frontier with time-variant individual efficiencies. Again, we estimate two Cobb-Douglas production frontiers, the first does not account for technological change, while the second does:

```
> riceCdSfaVar <- sfa( lProd ~ lArea + lLabor + lNpk,
+   timeEffect = TRUE, data = pdat )
> summary( riceCdSfaVar )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 11 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.182016	0.035251	5.1635	2.424e-07 ***
lArea	0.474919	0.066213	7.1726	7.360e-13 ***
lLabor	0.300094	0.063872	4.6983	2.623e-06 ***
lNpk	0.199461	0.042740	4.6669	3.058e-06 ***
sigmaSq	0.129957	0.021098	6.1598	7.285e-10 ***
gamma	0.369639	0.104045	3.5527	0.0003813 ***
time	0.058909	0.030863	1.9087	0.0563017 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -84.55036

panel data

number of cross-sections = 43

number of time periods = 8

total number of observations = 344

thus there are 0 observations not in the panel

mean efficiency of each year

1	2	3	4	5	6	7	8
0.7848433	0.7950303	0.8048362	0.8142652	0.8233226	0.8320146	0.8403483	0.8483313

mean efficiency: 0.817874

```
> riceCdTimeSfaVar <- sfa( lProd ~ lArea + lLabor + lNpk + mYear,
+   timeEffect = TRUE, data = pdat )
> summary( riceCdTimeSfaVar )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 13 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.1817471	0.0360859	5.0365	4.741e-07 ***
lArea	0.4761177	0.0657003	7.2468	4.267e-13 ***
lLabor	0.2987917	0.0647805	4.6124	3.981e-06 ***
lNpk	0.1991399	0.0428877	4.6433	3.429e-06 ***
mYear	-0.0031907	0.0155009	-0.2058	0.83692
sigmaSq	0.1255592	0.0295753	4.2454	2.182e-05 ***
gamma	0.3478660	0.1507342	2.3078	0.02101 *
time	0.0711165	0.0674356	1.0546	0.29162

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -84.52871

panel data

number of cross-sections = 43

number of time periods = 8

total number of observations = 344

thus there are 0 observations not in the panel

mean efficiency of each year

1	2	3	4	5	6	7	8
0.7780285	0.7905809	0.8025753	0.8140187	0.8249202	0.8352907	0.8451431	0.8544916

mean efficiency: 0.8181311

In the Cobb-Douglas production frontier that accounts for technological change, the monotonicity conditions are globally fulfilled and the (constant) output elasticities of land, labor and fertilizer are 0.476, 0.299, and 0.199, respectively. The estimated (constant) annual rate of technological change is around -0.3%, which indicates technological regress. However, the *t*-test for the

coefficient of the time trend and a likelihood ratio test indicate that the production technology (frontier) does not change over time, i.e. there is neither technological regress nor technological progress:

```
> lrtest( riceCdTimeSfaVar, riceCdSfaVar )
```

Likelihood ratio test

Model 1: riceCdTimeSfaVar

Model 2: riceCdSfaVar

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	8	-84.529			
2	7	-84.550	-1	0.0433	0.8352

A positive sign of the coefficient η (named `time`) indicates that efficiency is increasing over time. However, in the model without technological change, the t -test for the coefficient η and the corresponding likelihood ratio test indicate that the effect of time on the efficiencies only is significant at the 10% level:

```
> lrtest( riceCdSfaInv, riceCdSfaVar )
```

Likelihood ratio test

Model 1: riceCdSfaInv

Model 2: riceCdSfaVar

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	6	-86.43			
2	7	-84.55	1	3.7601	0.05249 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

In the model that accounts for technological change, the t -test for the coefficient η and the corresponding likelihood ratio test indicate that the efficiencies do not change over time:

```
> lrtest( riceCdTimeSfaInv, riceCdTimeSfaVar )
```

Likelihood ratio test

Model 1: riceCdTimeSfaInv

Model 2: riceCdTimeSfaVar

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	7	-85.074			
2	8	-84.529	1	1.0912	0.2962

Finally, we can use a likelihood ratio test to simultaneously test whether the technology and the technical efficiencies change over time:

```
> lrtest( riceCdSfaInv, riceCdTimeSfaVar )
```

Likelihood ratio test

Model 1: riceCdSfaInv

Model 2: riceCdTimeSfaVar

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	6	-86.430			
2	8	-84.529	2	3.8034	0.1493

All together, these tests indicate that there is no significant technological change, while it remains unclear whether the technical efficiencies significantly change over time.

In econometric estimations of frontier models, where one variable (e.g. time) can affect both the frontier and the efficiency, the two effects of this variable can often be hardly separated, because the corresponding parameters can be simultaneously adjusted with only marginally reducing the log-likelihood value. This can be checked by taking a look at the correlation matrix of the estimated parameters:

```
> round( cov2cor( vcov( riceCdTimeSfaVar ) ), 2 )
```

	(Intercept)	lArea	lLabor	lNpk	mYear	sigmaSq	gamma	time
(Intercept)	1.00	0.18	-0.12	0.01	0.06	0.44	0.47	-0.19
lArea	0.18	1.00	-0.68	-0.39	-0.06	0.04	0.06	0.07
lLabor	-0.12	-0.68	1.00	-0.27	0.08	-0.07	-0.09	0.00
lNpk	0.01	-0.39	-0.27	1.00	0.01	0.02	0.00	-0.11
mYear	0.06	-0.06	0.08	0.01	1.00	0.71	0.70	-0.88
sigmaSq	0.44	0.04	-0.07	0.02	0.71	1.00	0.94	-0.85
gamma	0.47	0.06	-0.09	0.00	0.70	0.94	1.00	-0.85
time	-0.19	0.07	0.00	-0.11	-0.88	-0.85	-0.85	1.00

The estimate of the parameter for technological change (`mYear`) is highly correlated with the estimate of the parameter that indicates the change of the efficiencies (`time`).

Again, further likelihood ratio tests show that OLS models are clearly rejected in favor of the corresponding stochastic frontier models:

```
> lrtest( riceCdSfaVar )
```

Likelihood ratio test

Model 1: OLS (no inefficiency)


```

Model 2: Error Components Frontier (ECF)
#Df  LogLik Df  Chisq Pr(>Chisq)
1    5 -104.91
2    7 -84.55  2 40.713  4.489e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> lrtest( riceCdTimeSfaVar )

```

Likelihood ratio test

```

Model 1: OLS (no inefficiency)
Model 2: Error Components Frontier (ECF)
#Df  LogLik Df  Chisq Pr(>Chisq)
1    6 -104.103
2    8 -84.529  2 39.149  9.85e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

In case of time-variant efficiencies, the `efficiencies` method returns a matrix, where each row corresponds to one of the 43 firms and each column corresponds to one of the 0 time periods:

```

> dim( efficiencies( riceCdSfaVar ) )

[1] 43  8

```

One can obtain a vector of efficiency estimates for each observation by setting argument `asInData` equal to `TRUE`:

```

> pdat$effCdVar <- efficiencies( riceCdSfaVar, asInData = TRUE )

```

9.2.1.3 Observation-specific efficiencies

In this section, we estimate a Cobb-Douglas production frontier with observation-specific efficiencies. The following commands estimate two Cobb-Douglas production frontiers, the first does not account for technological change, while the second does:

```

> riceCdSfa <- sfa( lProd ~ lArea + lLabor + lNpk, data = riceProdPhil )
> summary( riceCdSfa )

```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 9 iterations:
log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.333747	0.024468	13.6400	< 2.2e-16 ***
lArea	0.355511	0.060125	5.9128	3.363e-09 ***
lLabor	0.333302	0.063026	5.2883	1.234e-07 ***
lNpk	0.271277	0.035364	7.6709	1.708e-14 ***
sigmaSq	0.238627	0.025941	9.1987	< 2.2e-16 ***
gamma	0.885382	0.033524	26.4103	< 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -86.20268

cross-sectional data

total number of observations = 344

mean efficiency: 0.7229764

```
> riceCdTimeSfa <- sfa( lProd ~ lArea + lLabor + lNpk + mYear,
+   data = riceProdPhil )
> summary( riceCdTimeSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 10 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.3375352	0.0240787	14.0180	< 2.2e-16 ***
lArea	0.3557511	0.0596403	5.9649	2.447e-09 ***
lLabor	0.3507357	0.0631077	5.5577	2.733e-08 ***
lNpk	0.2565321	0.0351012	7.3083	2.704e-13 ***
mYear	0.0148902	0.0068853	2.1626	0.03057 *
sigmaSq	0.2418364	0.0259495	9.3195	< 2.2e-16 ***

```
gamma          0.8979766  0.0304374 29.5024 < 2.2e-16 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
log likelihood value: -83.76704
```

```
cross-sectional data
```

```
total number of observations = 344
```

```
mean efficiency: 0.7201094
```

Please note that we used the data set `riceProdPhil` for these estimations, because the panel structure should be ignored in these specifications and the data set `riceProdPhil` does not include information on the panel structure.

In the Cobb-Douglas production frontier that accounts for technological change, the monotonicity conditions are globally fulfilled and the (constant) output elasticities of land, labor and fertilizer are 0.356, 0.351, and 0.257, respectively. The estimated (constant) annual rate of technological change is around 1.5%.

A likelihood ratio test confirms the *t*-test for the coefficient of the time trend, i.e. the production technology significantly changes over time:

```
> lrtest( riceCdTimeSfa, riceCdSfa )
```

```
Likelihood ratio test
```

```
Model 1: riceCdTimeSfa
```

```
Model 2: riceCdSfa
```

```
  #Df  LogLik Df  Chisq Pr(>Chisq)
1    7 -83.767
2    6 -86.203 -1  4.8713    0.02731 *
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

9.2.1.4 Fixed effects in the production frontier

Finally, we estimate a Cobb-Douglas production frontier with individual fixed effects in the production frontier and observation-specific efficiencies. The following command estimates a Cobb-Douglas production frontier with fixed individual effects (not taking into account technological change:

```
> riceCdSfaFe <- sfa( lProd ~ lArea + lLabor + lNpk +
+   factor( FMERCODE ), data = riceProdPhil )
> summary( riceCdSfaFe )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 16 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.15387863	0.14069324	1.0937	0.274079
lArea	0.47233861	0.13314926	3.5474	0.000389 ***
lLabor	0.21814852	0.20500184	1.0641	0.287270
lNpk	0.15996035	0.11180783	1.4307	0.152524
factor(FMERCODE)2	0.38666552	0.94482481	0.4092	0.682359
factor(FMERCODE)3	0.06006811	0.94431233	0.0636	0.949280
factor(FMERCODE)4	0.28047324	0.92834233	0.3021	0.762559
factor(FMERCODE)5	0.10347919	0.88469352	0.1170	0.906887
factor(FMERCODE)6	-0.12885816	0.93300694	-0.1381	0.890153
factor(FMERCODE)7	0.26414148	0.94820476	0.2786	0.780575
factor(FMERCODE)8	0.05056796	0.93695106	0.0540	0.956958
factor(FMERCODE)9	0.10850284	0.94108468	0.1153	0.908211
factor(FMERCODE)10	0.19886092	0.95589067	0.2080	0.835200
factor(FMERCODE)11	-0.25014169	0.63033186	-0.3968	0.691485
factor(FMERCODE)12	0.20877250	0.83804960	0.2491	0.803270
factor(FMERCODE)13	-0.04603965	0.91817316	-0.0501	0.960009
factor(FMERCODE)14	0.05674580	0.97365044	0.0583	0.953524
factor(FMERCODE)15	-0.32714010	0.92265393	-0.3546	0.722916
factor(FMERCODE)16	0.00080999	0.87478739	0.0009	0.999261
factor(FMERCODE)17	0.40772295	0.84550955	0.4822	0.629649
factor(FMERCODE)18	0.39051951	0.79343994	0.4922	0.622588
factor(FMERCODE)19	0.37686324	0.81760639	0.4609	0.644845
factor(FMERCODE)20	0.24190587	0.93457208	0.2588	0.795758
factor(FMERCODE)21	0.18704412	0.92687142	0.2018	0.840072
factor(FMERCODE)22	0.13381531	0.79596839	0.1681	0.866492
factor(FMERCODE)23	0.07095477	0.92004850	0.0771	0.938528
factor(FMERCODE)24	0.04113838	0.91898711	0.0448	0.964295
factor(FMERCODE)25	0.25307508	0.92742348	0.2729	0.784946
factor(FMERCODE)26	0.05766269	0.94544161	0.0610	0.951367
factor(FMERCODE)27	0.11471950	0.96752114	0.1186	0.905616

9 Panel Data and Technological Change

```

factor(FMERCODE)28  0.21469079  0.91753698  0.2340  0.814996
factor(FMERCODE)29  0.03464250  0.86888015  0.0399  0.968197
factor(FMERCODE)30 -0.42673430  0.69841342 -0.6110  0.541196
factor(FMERCODE)31  0.28453713  0.94031605  0.3026  0.762197
factor(FMERCODE)32  0.52517557  0.41089775  1.2781  0.201208
factor(FMERCODE)33  0.10655964  0.97348249  0.1095  0.912836
factor(FMERCODE)34 -0.59248448  0.91660955 -0.6464  0.518029
factor(FMERCODE)35  0.25681291  0.94203109  0.2726  0.785148
factor(FMERCODE)36  0.03796698  0.79347793  0.0478  0.961837
factor(FMERCODE)37  0.25174598  0.93294408  0.2698  0.787283
factor(FMERCODE)38  0.34948385  0.95820893  0.3647  0.715316
factor(FMERCODE)39 -0.09070768  0.83054088 -0.1092  0.913032
factor(FMERCODE)40 -0.08208637  0.96734121 -0.0849  0.932375
factor(FMERCODE)41  0.23076919  0.91519353  0.2522  0.800922
factor(FMERCODE)42  0.25959721  0.93334414  0.2781  0.780907
factor(FMERCODE)43  0.17479501  0.72549382  0.2409  0.809607
sigmaSq              0.17985657  0.01777074 10.1209 < 2.2e-16 ***
gamma                0.98666690  0.04219155 23.3854 < 2.2e-16 ***

```

```

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -7.971807

```

cross-sectional data

total number of observations = 344

mean efficiency: 0.7395925

The (fixed) individual effects are highly statistically significant:

```
> lrtest( riceCdSfa, riceCdSfaFe )
```

Likelihood ratio test

Model 1: riceCdSfa

Model 2: riceCdSfaFe

```

#Df  LogLik Df  Chisq Pr(>Chisq)
1    6 -86.203
2   48 -7.972 42 156.46  4.285e-15 ***

```

```

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

The following command should estimate a Cobb-Douglas production frontier with fixed individual effects taking into account technological change but the estimation has problems to converge (at least at my computer):

```
> riceCdSfaFeTime <- sfa( lProd ~ lArea + lLabor + lNpk + mYear +
+   factor( FMERCODE ), data = riceProdPhil )
> summary( riceCdSfaFeTime )
```

9.2.2 Translog production frontier with constant and neutral technological change

The specification of a Translog production function that accounts for constant and neutral (un-biased) technological change is given in (9.4).¹

9.2.2.1 Observation-specific efficiencies

The following commands estimate a two Translog production frontiers with observation-specific efficiencies, the first does not account for technological change, while the second can account for constant and neutral technical change:

```
> riceTlSfa <- sfa( log( prod ) ~ log( area ) + log( labor ) + log( npk ) +
+   I( 0.5 * log( area )^2 ) + I( 0.5 * log( labor )^2 ) + I( 0.5 * log( npk )^2 ) +
+   I( log( area ) * log( labor ) ) + I( log( area ) * log( npk ) ) +
+   I( log( labor ) * log( npk ) ), data = riceProdPhil )
> summary( riceTlSfa )
```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 16 iterations:

cannot find a parameter vector that results in a log-likelihood value

larger than the log-likelihood value obtained in the previous step

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	3.3719e-01	2.8747e-02	11.7298	< 2.2e-16 ***
log(area)	5.3429e-01	7.9139e-02	6.7513	1.466e-11 ***
log(labor)	2.0910e-01	7.4439e-02	2.8090	0.0049699 **
log(npk)	2.2145e-01	4.5141e-02	4.9057	9.309e-07 ***
I(0.5 * log(area)^2)	-5.1502e-01	2.0692e-01	-2.4889	0.0128124 *

¹We use not only mean-scaled input quantities but also the mean-scaled output quantity in order to obtain the same estimates as Coelli et al. (2005, p. 250). Please note that the order of coefficients/regressors is different in Coelli et al. (2005, p. 250): intercept, mYear, log(area), log(labor), log(npk), 0.5*log(area)^2, log(area)*log(labor), log(area)*log(npk), 0.5*log(labor)^2, log(labor)*log(npk), 0.5*log(npk)^2.

```

I(0.5 * log(labor)^2)      -5.6134e-01  2.7039e-01 -2.0761 0.0378885 *
I(0.5 * log(npk)^2)        -7.1029e-05  9.4128e-02 -0.0008 0.9993979
I(log(area) * log(labor))  6.2604e-01  1.7284e-01  3.6221 0.0002922 ***
I(log(area) * log(npk))    8.1749e-02  1.3867e-01  0.5895 0.5555218
I(log(labor) * log(npk))   -1.5750e-01  1.4027e-01 -1.1228 0.2615321
sigmaSq                    2.1856e-01  2.4990e-02  8.7458 < 2.2e-16 ***
gamma                      8.6930e-01  3.9456e-02 22.0319 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
log likelihood value: -76.95413

```

```

cross-sectional data
total number of observations = 344

```

```

mean efficiency: 0.7326115

```

```

> riceTlTimeSfa <- sfa( log( prod ) ~ log( area ) + log( labor ) + log( npk ) +
+   I( 0.5 * log( area )^2 ) + I( 0.5 * log( labor )^2 ) + I( 0.5 * log( npk )^2 ) +
+   I( log( area ) * log( labor ) ) + I( log( area ) * log( npk ) ) +
+   I( log( labor ) * log( npk ) ) + mYear, data = riceProdPhil )
> summary( riceTlTimeSfa )

```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 17 iterations:

log likelihood values and parameters of two successive iterations
are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.3423626	0.0285089	12.0090	< 2.2e-16 ***
log(area)	0.5313816	0.0786313	6.7579	1.400e-11 ***
log(labor)	0.2308950	0.0744167	3.1027	0.0019174 **
log(npk)	0.2032741	0.0448189	4.5355	5.748e-06 ***
I(0.5 * log(area)^2)	-0.4758612	0.2021533	-2.3540	0.0185745 *
I(0.5 * log(labor)^2)	-0.5644708	0.2652593	-2.1280	0.0333374 *
I(0.5 * log(npk)^2)	-0.0072200	0.0923371	-0.0782	0.9376756
I(log(area) * log(labor))	0.6088402	0.1658019	3.6721	0.0002406 ***
I(log(area) * log(npk))	0.0617400	0.1383298	0.4463	0.6553627

```

I(log(labor) * log(npk))  -0.1370538  0.1407360 -0.9738  0.3301377
mYear                    0.0151111  0.0069164  2.1848  0.0289024 *
sigmaSq                  0.2217092  0.0251305  8.8223 < 2.2e-16 ***
gamma                    0.8835549  0.0367095 24.0688 < 2.2e-16 ***

```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
log likelihood value: -74.40992
```

```
cross-sectional data
```

```
total number of observations = 344
```

```
mean efficiency: 0.7294192
```

In the Translog production frontier that accounts for constant and neutral technological change, the monotonicity conditions are fulfilled at the sample mean and the estimated output elasticities of land, labor and fertilizer are 0.531, 0.231, and 0.203, respectively, at the sample mean. The estimated (constant) annual rate of technological progress is around 1.5%. A likelihood ratio test confirms the t -test for the coefficient of the time trend, i.e. the production technology (frontier) significantly changes over time:

```
> lrtest( riceTlTimeSfa, riceTlSfa )
```

```
Likelihood ratio test
```

```
Model 1: riceTlTimeSfa
```

```
Model 2: riceTlSfa
```

```

#Df  LogLik Df  Chisq Pr(>Chisq)
1   13 -74.410
2   12 -76.954 -1  5.0884   0.02409 *

```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Two further likelihood ratio tests indicate that the Translog specification is superior to the Cobb-Douglas specification, no matter whether the two models allow for technological change or not.

```
> lrtest( riceTlSfa, riceCdSfa )
```

```
Likelihood ratio test
```

```
Model 1: riceTlSfa
```

```
Model 2: riceCdSfa
```

```

#Df  LogLik Df  Chisq Pr(>Chisq)

```



```

1  12 -76.954
2   6 -86.203 -6 18.497   0.005103 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> lrtest( riceTlTimeSfa, riceCdTimeSfa )

Likelihood ratio test

Model 1: riceTlTimeSfa
Model 2: riceCdTimeSfa
   #Df  LogLik Df  Chisq Pr(>Chisq)
1   13 -74.410
2    7 -83.767 -6 18.714   0.004674 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

9.2.3 Translog production frontier with non-constant and non-neutral technological change

The specification of a Translog production function with non-Constant and non-Neutral technological change is given in (9.7).

9.2.3.1 Observation-specific efficiencies

The following command estimates a Translog production frontier with observation-specific efficiencies that can account for non-constant rates of technological change as well as biased technological change:

```

> riceTlTimeNnSfa <- sfa( log( prod ) ~ log( area ) + log( labor ) + log( npk ) +
+   I( 0.5 * log( area )^2 ) + I( 0.5 * log( labor )^2 ) + I( 0.5 * log( npk )^2 ) +
+   I( log( area ) * log( labor ) ) + I( log( area ) * log( npk ) ) +
+   I( log( labor ) * log( npk ) ) + mYear + I( mYear * log( area ) ) +
+   I( mYear * log( labor ) ) + I( mYear * log( npk ) ) + I( 0.5 * mYear^2 ),
+   data = riceProdPhil )
> summary( riceTlTimeNnSfa )

```

Error Components Frontier (see Battese & Coelli 1992)

Inefficiency decreases the endogenous variable (as in a production function)

The dependent variable is logged

Iterative ML estimation terminated after 21 iterations:

log likelihood values and parameters of two successive iterations

are within the tolerance limit

final maximum likelihood estimates

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	0.3106562	0.0314015	9.8931	< 2.2e-16 ***
log(area)	0.5126722	0.0786138	6.5214	6.965e-11 ***
log(labor)	0.2380479	0.0745020	3.1952	0.0013974 **
log(npk)	0.2151253	0.0443911	4.8461	1.259e-06 ***
I(0.5 * log(area)^2)	-0.5095014	0.2245191	-2.2693	0.0232500 *
I(0.5 * log(labor)^2)	-0.5394564	0.2627781	-2.0529	0.0400826 *
I(0.5 * log(npk)^2)	0.0212605	0.0923038	0.2303	0.8178341
I(log(area) * log(labor))	0.6132444	0.1687103	3.6349	0.0002781 ***
I(log(area) * log(npk))	0.0683940	0.1436778	0.4760	0.6340580
I(log(labor) * log(npk))	-0.1590173	0.1481054	-1.0737	0.2829677
mYear	0.0090026	0.0074634	1.2062	0.2277309
I(mYear * log(area))	0.0050527	0.0235232	0.2148	0.8299271
I(mYear * log(labor))	0.0241186	0.0254828	0.9465	0.3439117
I(mYear * log(npk))	-0.0335256	0.0176150	-1.9032	0.0570093 .
I(0.5 * mYear^2)	0.0149772	0.0069298	2.1613	0.0306744 *
sigmaSq	0.2227259	0.0243844	9.1339	< 2.2e-16 ***
gamma	0.8957679	0.0320612	27.9393	< 2.2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

log likelihood value: -70.5919

cross-sectional data

total number of observations = 344

mean efficiency: 0.728398

At the mean values of the input quantities and the middle of the observation period, the monotonicity conditions are fulfilled, the estimated output elasticities of land, labor and fertilizer are 0.513, 0.238, and 0.215, respectively, and the estimated annual rate of technological progress is around 0.9%.

The following likelihood ratio tests compare the Translog production frontier that can account for non-constant rates of technological change as well as biased technological change with the Translog production frontier that does not account for technological change and with the Translog production frontier that only accounts for constant and neutral technological change:

```
> lrtest( riceTlTimeNnSfa, riceTlSfa )
```

Likelihood ratio test

Model 1: riceTlTimeNnSfa

Model 2: riceTlSfa

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	17	-70.592			
2	12	-76.954	-5	12.725	0.0261 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
> lrtest( riceTlTimeNnSfa, riceTlTimeSfa )
```

Likelihood ratio test

Model 1: riceTlTimeNnSfa

Model 2: riceTlTimeSfa

	#Df	LogLik	Df	Chisq	Pr(>Chisq)
1	17	-70.592			
2	13	-74.410	-4	7.636	0.1059

These tests indicate that the Translog production frontier that can account for non-constant rates of technological change as well as biased technological change is superior to the Translog production frontier that does not account for any technological change but it is not significantly better than the Translog production frontier that accounts for constant and neutral technological change. Although it seems to be unnecessary to use the Translog production frontier that can account for non-constant rates of technological change as well as biased technological change, we use it in our further analysis for demonstrative purposes.

The following commands create short-cuts for some of the estimated coefficients and calculate the rates of technological change at each observation:

```
> at <- coef(riceTlTimeNnSfa)["mYear"]
> atArea <- coef(riceTlTimeNnSfa)["I(mYear * log(area))"]
> atLabor <- coef(riceTlTimeNnSfa)["I(mYear * log(labor))"]
> atNpk <- coef(riceTlTimeNnSfa)["I(mYear * log(npk))"]
> att <- coef(riceTlTimeNnSfa)["I(0.5 * mYear^2)"]
> riceProdPhil$tc <- with( riceProdPhil, at + atArea * log( area ) +
+   atLabor * log( labor ) + atNpk * log( npk ) + att * mYear )
```

The following command visualizes the variation of the individual rates of technological change:

```
> hist( riceProdPhil$tc, 20 )
```

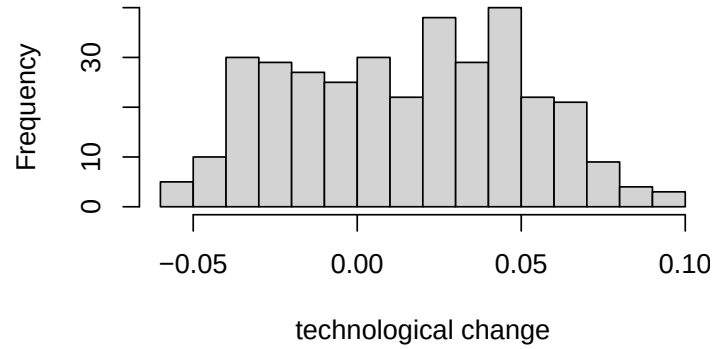


Figure 9.3: Annual rates of technological change

The resulting graph is shown in figure 9.3. Most individual rates of technological change are between -4% and $+7\%$, i.e. there is technological regress at some observations, while there is strong technological progress at other observations. This wide variation of annual rates of technological change is not unusual in applied agricultural production analysis because of the stochastic nature of agricultural production.

9.2.4 Decomposition of productivity growth

In the beginning of this course, we have discussed and calculated different productivity measures, of which the total factor productivity (TFP) is a particularly important determinant of a firm's competitiveness. During this course, we have—amongst other things—analyzed all three measures that affect a firm's total factor productivity, i.e.

- the current state of the technology (T) in the firm's sector, which might change due to technological change,
- the firm's technical efficiency (TE), which might change if the firm's distance to the current technology changes, and
- the firm's scale efficiency (SE), which might change if the firm's size relative to the optimal firm size changes.

Hence, changes of a firm's (or a sector's) total factor productivity (ΔTFP) can be decomposed into technological changes (ΔT), technical efficiency changes (ΔTE), and scale efficiency changes (ΔSE):

$$\Delta TFP \approx \Delta T + \Delta TE + \Delta SE \quad (9.11)$$

This decomposition often helps to understand the reasons for improved or reduced total factor productivity and competitiveness.

9.3 Analyzing productivity growths with Data Envelopment Analysis (DEA)

```
> library( "Benchmarking" )
```

We create a matrix of input quantities and a vector of output quantities:

```
> xMat <- cbind( riceProdPhil$AREA, riceProdPhil$LABOR, riceProdPhil$NPK )
> yVec <- riceProdPhil$PROD
```

The following commands calculate and decompose productivity changes:

```
> xMat0 <- xMat[ riceProdPhil$YEARDUM == 1, ]
> xMat1 <- xMat[ riceProdPhil$YEARDUM == 2, ]
> yVec0 <- yVec[ riceProdPhil$YEARDUM == 1 ]
> yVec1 <- yVec[ riceProdPhil$YEARDUM == 2 ]
> c00 <- eff( dea( xMat0, yVec0, RTS = "crs" ) )
> c01 <- eff( dea( xMat0, yVec0, XREF = xMat1, YREF = yVec1, RTS = "crs" ) )
> c11 <- eff( dea( xMat1, yVec1, RTS = "crs" ) )
> c10 <- eff( dea( xMat1, yVec1, XREF = xMat0, YREF = yVec0, RTS = "crs" ) )
```

Productivity changes (Malmquist):

```
> dProd0 <- c10 / c00
> hist( dProd0 )
> dProd1 <- c11 / c01
> plot( dProd0, dProd1 )
> dProd <- sqrt( dProd0 * dProd1 )
> hist( dProd )
```

Technological changes:

```
> dTech0 <- c00 / c01
> dTech1 <- c10 / c11
> plot( dTech0, dTech1 )
> dTech <- sqrt( dTech0 * dTech1 )
> hist( dTech )
```

Efficiency changes:

```
> dEff <- c11 / c00
> hist( dEff )
```

Checking Malmquist decomposition:

```
> all.equal( dProd, dTech * dEff )
```

```
[1] TRUE
```

9.4 Cost functions with technical change

9.4.1 Translog cost function with constant and neutral technical change

The general specification of a Translog cost function with constant and neutral technical change is:

$$\begin{aligned} \ln c(w, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln w_i + \alpha_y \ln y \\ & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln w_i \ln w_j + \frac{1}{2} \alpha_{yy} (\ln y)^2 \\ & + \sum_{i=1}^N \alpha_{iy} \ln w_i \ln y + \alpha_t t \end{aligned} \quad (9.12)$$

with $\alpha_{ij} = \alpha_{ji} \forall i, j$.

This specification assumes that the rate of technical change:

$$\frac{\partial \ln c(w, y)}{\partial t} = \alpha_t \quad (9.13)$$

is constant over time and does not depend on the input prices or the output quantity. The value of $\partial \ln c(w, y)/\partial t$ multiplied by 100 indicates by how many percent *ceteris paribus* the total cost changes per time period due to technical change. A negative value of $\partial \ln c(w, y)/\partial t$ indicates technical progress, because this means that total cost *ceteris paribus* decreases over time, while a positive value of $\partial \ln c(w, y)/\partial t$ indicates technical regress, because this means that total cost—*ceteris paribus*—increases over time.

Furthermore, this specification assumes that the optimal cost shares of the inputs:

$$s_i(w, y) = \frac{\partial \ln c(w, y)}{\partial \ln w_i} = \alpha_i + \sum_{j=1}^N \alpha_{ij} \ln w_j + \alpha_{iy} \ln y \quad \forall i \quad (9.14)$$

are not affected by technical change. This means that technical change is Hicks-neutral.

9.4.2 Translog cost function with non-constant and non-neutral technical change

The general specification of a Translog cost function with non-constant and non-neutral technical change is:

$$\begin{aligned} \ln c(w, y) = & \alpha_0 + \sum_{i=1}^N \alpha_i \ln w_i + \alpha_y \ln y \\ & + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_{ij} \ln w_i \ln w_j + \frac{1}{2} \alpha_{yy} (\ln y)^2 + \sum_{i=1}^N \alpha_{iy} \ln w_i \ln y \end{aligned} \quad (9.15)$$

$$+ \alpha_t t + \sum_{i=1}^N \alpha_{ti} t \ln w_i + \alpha_{ty} t \ln y + \frac{1}{2} \alpha_{tt} t^2$$

with $\alpha_{ij} = \alpha_{ji} \forall i, j$.

In this specification, the rate of technical change is:

$$\frac{\partial \ln c(w, y)}{\partial t} = \alpha_t + \sum_{i=1}^N \alpha_{ti} \ln w_i + \alpha_{ty} \ln y + \alpha_{tt} t \quad (9.16)$$

and, hence, it can change over time (t) and it depends on the input prices (w) and the output quantity (y). The interpretation of the rate of technical change $\partial \ln c(w, y)/\partial t$ is explained in section 9.4.1.

Furthermore, in this specification, the optimal cost shares of the inputs:

$$s_i(w, y) = \frac{\partial \ln c(w, y)}{\partial \ln w_i} = \alpha_i + \sum_{j=1}^N \alpha_{ij} \ln w_j + \alpha_{iy} \ln y + \alpha_{ti} t \quad \forall i \quad (9.17)$$

are affected by technical change. For instance, if α_{ti} is strictly positive, the optimal cost share and, thus, the optimal quantity of input i will *ceteris paribus* increase over time so that this type of technical change is called “input i using technical change.” In contrast, if α_{ti} is strictly negative, the optimal cost share and, thus, the optimal quantity of input i will *ceteris paribus* decrease over time so that this type of technical change is called “input i saving technical change.”

Bibliography

- Aigner, D., C.A.K. Lovell, and P. Schmidt. 1977. "Formulation and Estimation of Stochastic Frontier Production Function Models." *Journal of Econometrics* 6:21–37.
- Battese, G.E., and T.J. Coelli. 1992. "Frontier Production Functions, Technical Efficiency and Panel Data: With Application to Paddy Farmers in India." *Journal of Productivity Analysis* 3:153–169.
- . 1995. "A Model for Technical Inefficiency Effects in a Stochastic Frontier Production Function for Panel Data." *Empirical Economics* 20:325–332.
- Bogetoft, P., and L. Otto. 2011. *Benchmarking with DEA, SFA, and R*, vol. 157 of *International Series in Operations Research & Management Science*. Springer.
- Chambers, R.G. 1988. *Applied Production Analysis. A Dual Approach*. Cambridge University Press, Cambridge.
- Chand, R., and J.L. Kaul. 1986. "A Note on the Use of the Cobb-Douglas Profit Function." *American Journal of Agricultural Economics* 68:162–164.
- Chiang, A.C. 1984. *Fundamental Methods of Mathematical Economics*, 3rd ed. McGraw-Hill.
- Coelli, T.J. 1995. "Estimators and Hypothesis Tests for a Stochastic: A Monte Carlo Analysis." *Journal of Productivity Analysis* 6:247–268.
- . 1996. "A Guide to FRONTIER Version 4.1: A Computer Program for Stochastic Frontier Production and Cost Function Estimation." CEPA Working Paper 96/7, Department of Econometrics, University of New England, Armidale NSW Australia.
- Coelli, T.J., D.S.P. Rao, C.J. O'Donnell, and G.E. Battese. 2005. *An Introduction to Efficiency and Productivity Analysis*, 2nd ed. New York: Springer.
- Croissant, Y., and G. Millo. 2008. "Panel Data Econometrics in R: The plm Package." *Journal of Statistical Software* 27:1–43.
- Czekaj, T., and A. Henningsen. 2012. "Comparing Parametric and Nonparametric Regression Methods for Panel Data: the Optimal Size of Polish Crop Farms." FOI Working Paper No. 2012/12, Institute of Food and Resource Economics, University of Copenhagen.

- Färe, R., and D. Primont. 1995. *Multi-Output Product and Duality: Theory and Applications*. Boston: Kluwer Academic Publishers.
- Gantmacher, F.R. 1959. *The Theory of Matrices*. Chelsea Publishing.
- Hayfield, T., and J.S. Racine. 2008. “Nonparametric Econometrics: The np Package.” *Journal of Statistical Software* 27:1–32.
- Henning, C.H.C.A., and A. Henningsen. 2007. “Modeling Farm Households’ Price Responses in the Presence of Transaction Costs and Heterogeneity in Labor Markets.” *American Journal of Agricultural Economics* 89:665–681.
- Henningsen, A., and C.H.C.A. Henning. 2009. “Imposing Regional Monotonicity on Translog Stochastic Production Frontiers with a Simple Three-Step Procedure.” *Journal of Productivity Analysis* 32:217–229.
- Hurvich, C.M., J.S. Simonoff, and C.L. Tsai. 1998. “Smoothing Parameter Selection in Nonparametric Regression Using an Improved Akaike Information Criterion.” *Journal of the Royal Statistical Society Series B* 60:271–293.
- Ivaldi, M., N. Ladoux, H. Ossard, and M. Simioni. 1996. “Comparing Fourier and Translog Specifications of Multiproduct Technology: Evidence from an Incomplete Panel of French Farmers.” *Journal of Applied Econometrics* 11:649–667.
- Kleiber, C., and A. Zeileis. 2008. *Applied Econometrics with R*. New York: Springer-Verlag, ISBN 978-0-387-77316-2.
- Li, Q., and J.S. Racine. 2007. *Nonparametric Econometrics: Theory and Practice*. Princeton: Princeton University Press.
- Löthgren, M. 1997. “Generalized Stochastic Frontier Production Models.” *Economics Letters* 57:255–259.
- . 2000. “Specification and Estimation of Stochastic Multiple-output Production and Technical Inefficiency.” *Applied Economics* 32:1533–1540.
- McClelland, J.W., M.E. Wetzstein, and W.N. Musserwetz. 1986. “Returns to Scale and Size in Agricultural Economics.” *Western Journal of Agricultural Economics* 11:129–133.
- Meeusen, W., and J. van den Broeck. 1977. “Efficiency Estimation from Cobb-Douglas Production Functions with Composed Error.” *International Economic Review* 18:435–444.
- O’Donnell, C.J., and T.J. Coelli. 2005. “A Bayesian Approach to Imposing Curvature on Distance Functions.” *Journal of Econometrics* 126:493–523.

Bibliography

- Olsen, J.V., and A. Henningsen. 2011. "Investment Utilization and Farm Efficiency in Danish Agriculture." FOI Working Paper No. 2011/13, Institute of Food and Resource Economics, University of Copenhagen.
- Quiggin, J., and A. Bui-Lan. 1984. "The Use of Cross-Sectional Estimates of Profit Functions for Tests of Relative Efficiency: A Critical Review." *Australian Journal of Agricultural Economics* 28:44–55.
- Racine, J.S. 2008. "Nonparametric Econometrics: A Primer." *Foundations and Trends in Econometrics* 3:1–88.
- Ramsey, J.B. 1969. "Tests for Specification Errors in Classical Linear Least-Squares Regression Analysis." *Journal of the Royal Statistical Society. Series B (Methodological)* 31:350–371.
- Reig-Martinez, E., A. Picazo-Tadeo, and F. Hernandez-Sancho. 2001. "The Calculation of Shadow Prices for Industrial Wastes Using Distance Functions: An Analysis for Spanish Ceramic Pavements Firms." *International Journal of Production Economics* 69:277–285.
- Teetor, P. 2011. *R Cookbook*. O'Reilly Media.
- Zuur, A., E.N. Ieno, and E. Meesters. 2009. *A Beginner's Guide to R*. Use R!, Springer.