

# Modelling Maize Project: Cob.weight

Normal linear regression

Author: Nisia Trisconi | Zurich Data Scientists  
Reviewer: Dr. Luisa Barbanti | Zurich Data Scientists

October 30, 2023

## Contents

<b>1</b>	<b>Freezing Package versions</b>	<b>2</b>
<b>2</b>	<b>Load packages</b>	<b>2</b>
<b>3</b>	<b>Settings</b>	<b>2</b>
<b>4</b>	<b>Getting data</b>	<b>3</b>
<b>5</b>	<b>Design</b>	<b>5</b>
<b>6</b>	<b>Response variable: <i>Cob_weight</i></b>	<b>5</b>
6.1	Aim . . . . .	5
6.2	Model fitting . . . . .	5
6.3	Model selection . . . . .	8
6.3.1	Confidence intervals . . . . .	15
6.4	Model checking . . . . .	18
6.5	Contrasts . . . . .	26
<b>7</b>	<b>Methods description</b>	<b>28</b>
<b>8</b>	<b>Session information</b>	<b>29</b>

# 1 Freezing Package versions

The following code lines are commented out because the `{checkpoint}` package no longer works.

```
## (messages are omitted in this chunk)
##
# library(checkpoint)
# checkpoint(snapshot_date = "2022-11-15")
```

# 2 Load packages

```
## (messages are omitted from this chunk)
##
library(multcomp)
library(dplyr)
library(kableExtra)
library(ggplot2)
library(tibble) ## function rownames_to_column()
library(plgraphics)
```

# 3 Settings

Global settings:

```
Sys.setenv(lang = "en_US")
theme_set(theme_bw())

if (!dir.exists("Prepared_data_and_models")) {
  dir.create("Prepared_data_and_models")
}
```

## 4 Getting data

```
d.maize <- readRDS(file = paste0("Prepared_data_and_models/",  
                                "d.maize_PreparedData.RDS"))
```

Overview of the data:

```
dim(d.maize)
```

```
[1] 108 33
```

```
head(d.maize)[1:min(ncol(d.maize), 30)]
```

```
# A tibble: 6 x 30  
  pot    soil      well depth seed.weight fungus date.germinated observations  
  <chr> <chr>    <chr> <dbl>    <dbl> <chr>    <chr>          <chr>  
1 A1    Bio garden a      3      30 <NA>    2022-05-11    <NA>  
2 A1    Bio garden b      5      34 <NA>    2022-05-11    <NA>  
3 A1    Bio garden c      2      35 <NA>    2022-05-09    <NA>  
4 A1    Bio garden d      1      40 <NA>    2022-05-10    <NA>  
5 A1    Bio garden e      4      46 <NA>    2022-05-11    <NA>  
6 A1    Bio garden f      6      37 <NA>    2022-05-11    <NA>  
# i 22 more variables: height_2022_07_05 <chr>, cob_weight <chr>, ...12 <dbl>,  
# pot.fac <fct>, soil.fac <fct>, well.fac <fct>, seed.weight.grams <dbl>,  
# fungus.fac <fct>, date.germinated.asDate <date>, obs.time <fct>,  
# broken <lgl>, height_2022_07_05.num <dbl>, plant.found <lgl>,  
# cob_weight.num <dbl>, germinated.in.lab <lgl>, germinated.in.field <lgl>,  
# germinated.yes <lgl>, days.to.germination <dbl>,  
# days.to.germination.censored <dbl>, seed_coord_y <dbl>, ...
```

```
str(d.maize)
```

```
tibble [108 x 33] (S3: tbl_df/tbl/data.frame)  
$ pot           : chr [1:108] "A1" "A1" "A1" "A1" ...  
$ soil          : chr [1:108] "Bio garden" "Bio garden" "Bio garden" "Bio garden" ...  
$ well         : chr [1:108] "a" "b" "c" "d" ...  
$ depth        : num [1:108] 3 5 2 1 4 6 6 4 5 1 ...  
$ seed.weight   : num [1:108] 30 34 35 40 46 37 27 16 23 22 ...  
$ fungus       : chr [1:108] NA NA NA NA ...  
$ date.germinated : chr [1:108] "2022-05-11" "2022-05-11" "2022-05-09" "2022-05-10" ...  
$ observations  : chr [1:108] NA NA NA NA ...  
$ height_2022_07_05 : chr [1:108] "217" "131" "143" "194" ...  
$ cob_weight    : chr [1:108] "117" "26" "61" "109" ...  
$ ...12        : num [1:108] NA NA NA NA NA NA NA NA NA NA ...  
$ pot.fac      : Factor w/ 18 levels "A1","A2","A3",...: 1 1 1 1 1 1 2 2 2 2 ...  
$ soil.fac     : Factor w/ 4 levels "Bio garden","Composana",...: 1 1 1 1 1 1 3 3 3 3 ..  
$ well.fac     : Factor w/ 6 levels "a","b","c","d",...: 1 2 3 4 5 6 1 2 3 4 ...  
$ seed.weight.grams : num [1:108] 0.3 0.34 0.35 0.4 0.46 0.37 0.27 0.16 0.23 0.22 ...  
$ fungus.fac   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...  
$ date.germinated.asDate : Date[1:108], format: "2022-05-11" "2022-05-11" ...  
$ obs.time     : Factor w/ 2 levels "morning","night": 2 2 2 2 2 2 2 2 2 2 ...  
$ broken       : logi [1:108] FALSE FALSE FALSE FALSE FALSE FALSE ...  
$ height_2022_07_05.num : num [1:108] 217 131 143 194 206 233 158 282 241 232 ...  
$ plant.found  : logi [1:108] TRUE TRUE TRUE TRUE TRUE TRUE ...  
$ cob_weight.num : num [1:108] 117 26 61 109 106 156 57 286 51 120 ...  
$ germinated.in.lab : logi [1:108] TRUE TRUE TRUE TRUE TRUE TRUE ...
```

```

$ germinated.in.field      : logi [1:108] FALSE FALSE FALSE FALSE FALSE FALSE ...
$ germinated.yes          : logi [1:108] TRUE TRUE TRUE TRUE TRUE TRUE ...
$ days.to.germination      : num [1:108] 11 11 9 10 11 11 11 11 NA 9 ...
$ days.to.germination.censored: num [1:108] 11 11 9 10 11 11 11 11 14 9 ...
$ seed_coord_y            : num [1:108] 1 1 2 2 3 3 1 1 2 2 ...
$ seed_coord_x            : num [1:108] 1 2 1 2 1 2 3 4 3 4 ...
$ position_field_x        : num [1:108] 1 1 1 1 1 1 1 1 1 1 ...
$ position_field_x_cm     : num [1:108] 50 50 50 50 50 50 50 50 50 50 ...
$ position_field_y        : int [1:108] 1 2 3 4 5 6 7 8 9 10 ...
$ position_field_y_cm     : num [1:108] 25 50 75 100 125 150 175 200 225 250 ...

```

## 5 Design

108 maize seeds are planted in 18 different pots, each with 6 wells.

Inside one pot, the same soil is used. The soils that were used are: Bio garden (4 pots), Composana (4 pots), herbs (6 pots), mixture (4 pots).

In each well, one maize seed is planted at a pre-defined depth (in cm), which is allocated randomly to the well. The maximum value for depth is 6cm and this corresponds to planting the seed directly in the coconut fiber that makes up the pot.

Wells in the same pots are allocated as follows:

```
      [,1] [,2]
[1,] "e"  "f"
[2,] "c"  "d"
[3,] "a"  "b"
```

The pots are arranged as follows on a table in the lab:

```
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,] "C1" "C2" "C3" "C4" "C5" "C6"
[2,] "B1" "B2" "B3" "B4" "B5" "B6"
[3,] "A1" "A2" "A3" "A4" "A5" "A6"
```

Seeds are watered for the first time on 04.30.2022 and are transferred to the field on 05.15.2022 according to the same scheme.

Some seeds are broken when planted, one seed develops a fungus.

Some seeds germinate in the lab, others in the field, while some seeds never germinate.

On 07.05.2022, the height of all maize plants is measured in cm. The plants that were not measured for time reasons receive a height value of `not measured`, while the plants that were not found and could hence not be measured present missing values.

On 09.16.2022, the weight of the cob is measured for all plants that have a cob. The variety of maize that was planted typically yields 1 cob per plant.

## 6 Response variable: *Cob\_weight*

### 6.1 Aim

We are interested in testing whether the cob weight (*cob\_weight.num*) is influenced by:

- type of soil (variable *soil.fac*)
- Well (variable *well.fac*)
- Seed weight (variable *seed.weight*)
- Depth of the seed in the soil (variable *depth*)

### 6.2 Model fitting

The variable *cob\_weight.num* is a continuous variable (in particular it is an amount), whose density is already well-centered, so there is no need to log transform it. For completeness, however, we still fit a model with the log-transformed version of the response and show that there is no evidence that this version is to be preferred.

Based on the graphical analysis, it is not necessary to transform any of the explanatory variables. Therefore, they can be used as they are in the model.

*seed.weight* and *depth* are numeric variables which are introduced in the model as continuous variables; all the others are introduced as factors.

The first step is to fit a model using all the interesting available explanatory variables.

```
lm.cob_weight.full <- lm(cob_weight.num ~
                        soil.fac +
                        depth +
                        pot.fac +
                        well.fac +
                        seed.weight,
                        data = d.maize)

##
summary(lm.cob_weight.full)
```

Call:

```
lm(formula = cob_weight.num ~ soil.fac + depth + pot.fac + well.fac +
    seed.weight, data = d.maize)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-98.5039	-18.6240	0.6776	20.4813	137.2356

Coefficients: (3 not defined because of singularities)

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	88.49502	51.86128	1.7064	0.09358 .
soil.facComposana	14.38967	30.27323	0.4753	0.63644
soil.facherbs	-3.54592	33.60586	-0.1055	0.91635
soil.facmixture	34.29305	36.79585	0.9320	0.35542
depth	1.93370	3.63131	0.5325	0.59652
pot.facA2	33.76694	34.72054	0.9725	0.33504
pot.facA3	28.64146	34.15904	0.8385	0.40539
pot.facA4	26.04026	34.36254	0.7578	0.45180
pot.facA5	72.08511	35.91128	2.0073	0.04964 *
pot.facA6	35.46426	37.43460	0.9474	0.34760
pot.facB1	-30.76121	37.49444	-0.8204	0.41552
pot.facB2	7.77553	37.81115	0.2056	0.83783
pot.facB3	1.75388	33.04154	0.0531	0.95786
pot.facB4	17.30449	32.68694	0.5294	0.59866
pot.facB5	15.27163	34.27261	0.4456	0.65764
pot.facB6	-95.17775	46.89952	-2.0294	0.04727 *
pot.facC1	NA	NA	NA	NA
pot.facC2	2.02607	32.76782	0.0618	0.95092
pot.facC3	-5.66174	34.24914	-0.1653	0.86931
pot.facC4	7.53108	33.04479	0.2279	0.82057
pot.facC5	NA	NA	NA	NA
pot.facC6	NA	NA	NA	NA
well.facb	29.46336	21.34958	1.3800	0.17316
well.facc	31.39721	19.43265	1.6157	0.11188
well.facd	-2.60429	20.88567	-0.1247	0.90122
well.face	22.19464	19.32407	1.1485	0.25571
well.facf	22.17486	19.08705	1.1618	0.25034
seed.weight	-0.44686	1.19607	-0.3736	0.71013

---

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

Residual standard error: 50.186 on 55 degrees of freedom

(28 observations deleted due to missingness)

Multiple R-squared: 0.27033, Adjusted R-squared: -0.048075

F-statistic: 0.84901 on 24 and 55 DF, p-value: 0.66249

We observe that certain coefficients associated with the variable *pot.fac* are not estimated.

This is the case because the variable *soil.fac* is nested in the variable *pot.fac*, thus the model is rank-deficient.

Indeed, let's show this.

```
X <- model.matrix(lm.cob_weight.full)
robustbase::rankMM(X)
```

```
[1] 25
```

```
ncol(X)
```

```
[1] 28
```

The number of columns is greater than the rank of the matrix, showing that the matrix is not full rank, thus it is not invertible.

Consequently, we cannot estimate some of these coefficients.

Two options exist at this point: fitting a mixed-effects model (as described in the document “2\_Modelling\_ranef\_maize.pdf”) or fitting a linear model that includes only the lower level of nestitude, and then conducting a post-hoc test to assess the higher level.

The second option is applied in this document. Consequently, the variable *soil.fac* is dropped from the model.

```
lm.cob_weight <- lm(cob_weight.num ~ pot.fac +
                    depth +
                    well.fac +
                    seed.weight,
                    data = d.maize)
##
summary(lm.cob_weight)
```

Call:

```
lm(formula = cob_weight.num ~ pot.fac + depth + well.fac + seed.weight,
    data = d.maize)
```

Residuals:

Min	1Q	Median	3Q	Max
-98.5039	-18.6240	0.6776	20.4813	137.2356

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	88.49502	51.86128	1.7064	0.09358 .
pot.facA2	30.22102	33.70963	0.8965	0.37389
pot.facA3	25.09554	31.55510	0.7953	0.42986
pot.facA4	26.04026	34.36254	0.7578	0.45180
pot.facA5	68.53919	33.42529	2.0505	0.04509 *
pot.facA6	31.91834	36.69504	0.8698	0.38818
pot.facB1	3.53184	30.88707	0.1143	0.90938
pot.facB2	42.06858	33.12431	1.2700	0.20942

```

pot.facB3    16.14356    33.00651    0.4891    0.62671
pot.facB4    31.69416    34.13649    0.9285    0.35723
pot.facB5    11.72571    32.37410    0.3622    0.71860
pot.facB6   -60.88470    42.70378   -1.4257    0.15959
pot.facC1    34.29305    36.79585    0.9320    0.35542
pot.facC2     2.02607    32.76782    0.0618    0.95092
pot.facC3    -5.66174    34.24914   -0.1653    0.86931
pot.facC4    21.92075    35.35609    0.6200    0.53782
pot.facC5    14.38967    30.27323    0.4753    0.63644
pot.facC6    -3.54592    33.60586   -0.1055    0.91635
depth         1.93370     3.63131    0.5325    0.59652
well.facb    29.46336    21.34958    1.3800    0.17316
well.facc    31.39721    19.43265    1.6157    0.11188
well.facd    -2.60429    20.88567   -0.1247    0.90122
well.face    22.19464    19.32407    1.1485    0.25571
well.facf    22.17486    19.08705    1.1618    0.25034
seed.weight  -0.44686     1.19607   -0.3736    0.71013
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

Residual standard error: 50.186 on 55 degrees of freedom
(28 observations deleted due to missingness)
Multiple R-squared:  0.27033,    Adjusted R-squared:  -0.048075
F-statistic: 0.84901 on 24 and 55 DF,  p-value: 0.66249

```

All the coefficients are now correctly estimated.

Looking at the residuals' quartiles, there does not seem to be a strong deviation from the normal distribution.

Examining the estimated coefficients, it appears that none of these variables have a significant influence on the variable *cob\_weight*.

However, we will calculate and plot the confidence intervals to gain a better understanding.

If we look at the estimated values for the “Multiple R-squared” and the “Adjusted R-square”, they seem to indicate a really poor fitting.

### 6.3 Model selection

We fit the log-transformed model to check whether it could be a better fit to the data set.

```

lm.cob_weight.log <- update(lm.cob_weight, formula = log(.) ~ .)
##
summary(lm.cob_weight.log)

```

Call:

```

lm(formula = log(cob_weight.num) ~ pot.fac + depth + well.fac +
    seed.weight, data = d.maize)

```

Residuals:

```

      Min       1Q   Median       3Q      Max
-3.61021 -0.20355  0.04337  0.35384  1.41624

```

Coefficients:

```

              Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.08715136  0.77126527  5.2993 2.113e-06 ***

```

pot.facA2	0.26249982	0.50131944	0.5236	0.6026
pot.facA3	0.36607178	0.46927789	0.7801	0.4387
pot.facA4	0.36561706	0.51102928	0.7155	0.4774
pot.facA5	0.55704180	0.49709085	1.1206	0.2673
pot.facA6	0.46496071	0.54571746	0.8520	0.3979
pot.facB1	0.06102232	0.45934323	0.1328	0.8948
pot.facB2	0.52388176	0.49261477	1.0635	0.2922
pot.facB3	0.27540974	0.49086289	0.5611	0.5770
pot.facB4	0.45531626	0.50766755	0.8969	0.3737
pot.facB5	0.20076935	0.48145782	0.4170	0.6783
pot.facB6	-0.72585353	0.63507765	-1.1429	0.2580
pot.facC1	0.45343236	0.54721673	0.8286	0.4109
pot.facC2	0.02729596	0.48731312	0.0560	0.9555
pot.facC3	-0.62076026	0.50934281	-1.2187	0.2281
pot.facC4	0.28232549	0.52580510	0.5369	0.5935
pot.facC5	0.29357996	0.45021431	0.6521	0.5171
pot.facC6	-0.02343987	0.49977613	-0.0469	0.9628
depth	0.05556140	0.05400371	1.0288	0.3081
well.facb	0.22018676	0.31750442	0.6935	0.4909
well.facc	0.31415868	0.28899652	1.0871	0.2817
well.facd	0.01407761	0.31060535	0.0453	0.9640
well.face	0.31332767	0.28738170	1.0903	0.2803
well.facf	0.03883021	0.28385685	0.1368	0.8917
seed.weight	-0.00024546	0.01778756	-0.0138	0.9890

---

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

Residual standard error: 0.74635 on 55 degrees of freedom

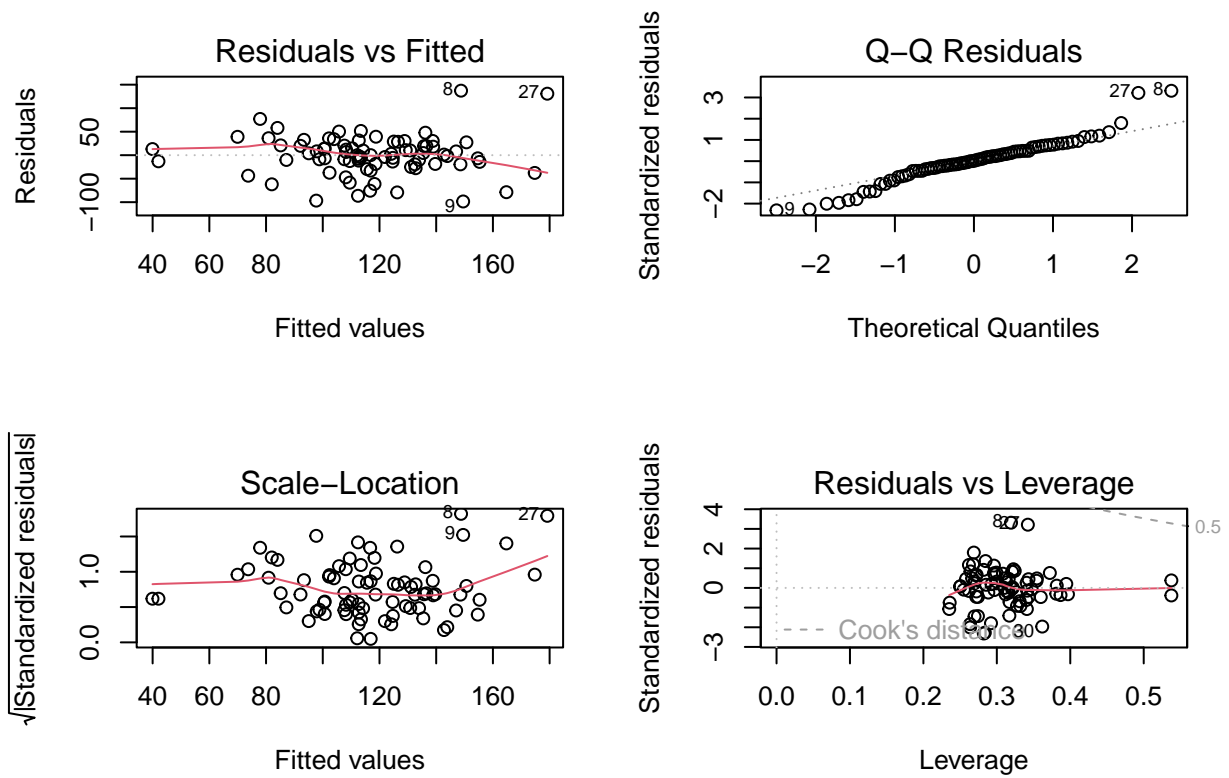
(28 observations deleted due to missingness)

Multiple R-squared: 0.23202, Adjusted R-squared: -0.1031

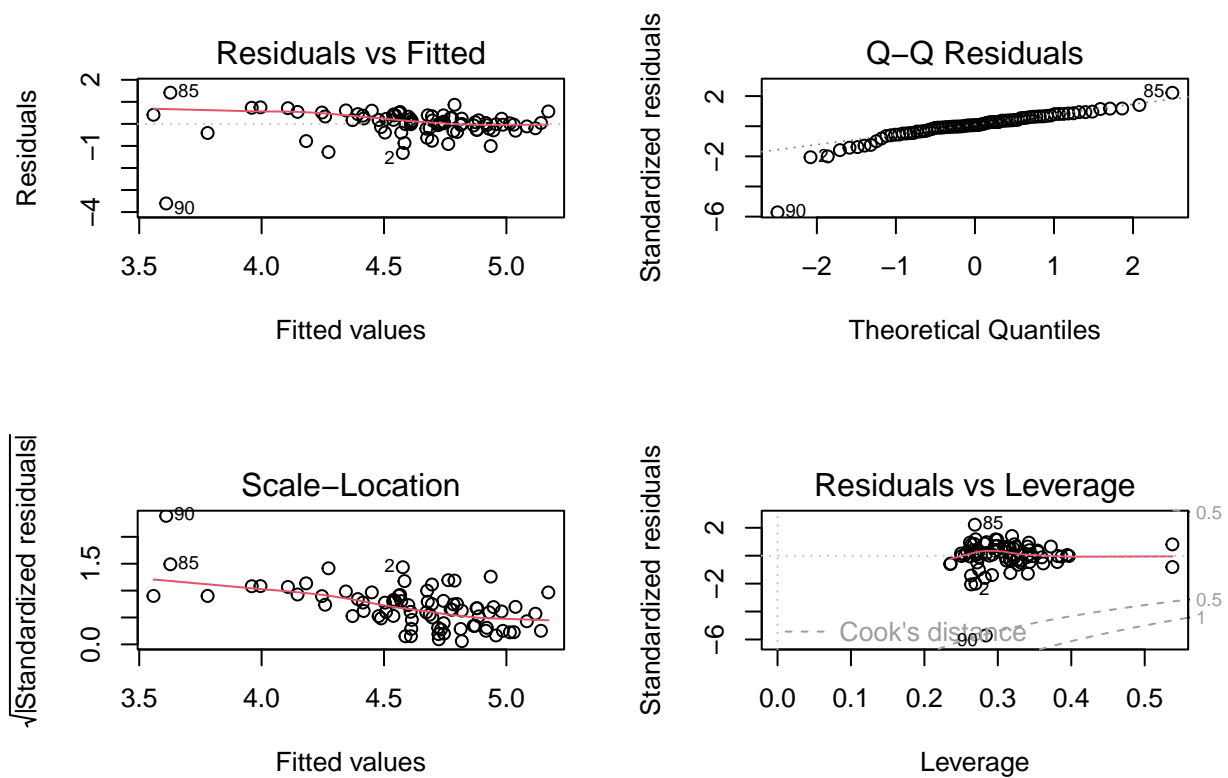
F-statistic: 0.69236 on 24 and 55 DF, p-value: 0.8372

The two models are not nested, which means we cannot use the `anova()` function to compare them. Additionally, since they are on different scales, the AIC and BIC criteria cannot be used to assess the best model. Therefore, we must evaluate the best model by examining the residual plots.

```
par(mfrow = c(2, 2))
plot(lm.cob_weight)
```



```
plot(lm.cob_weight.log)
```



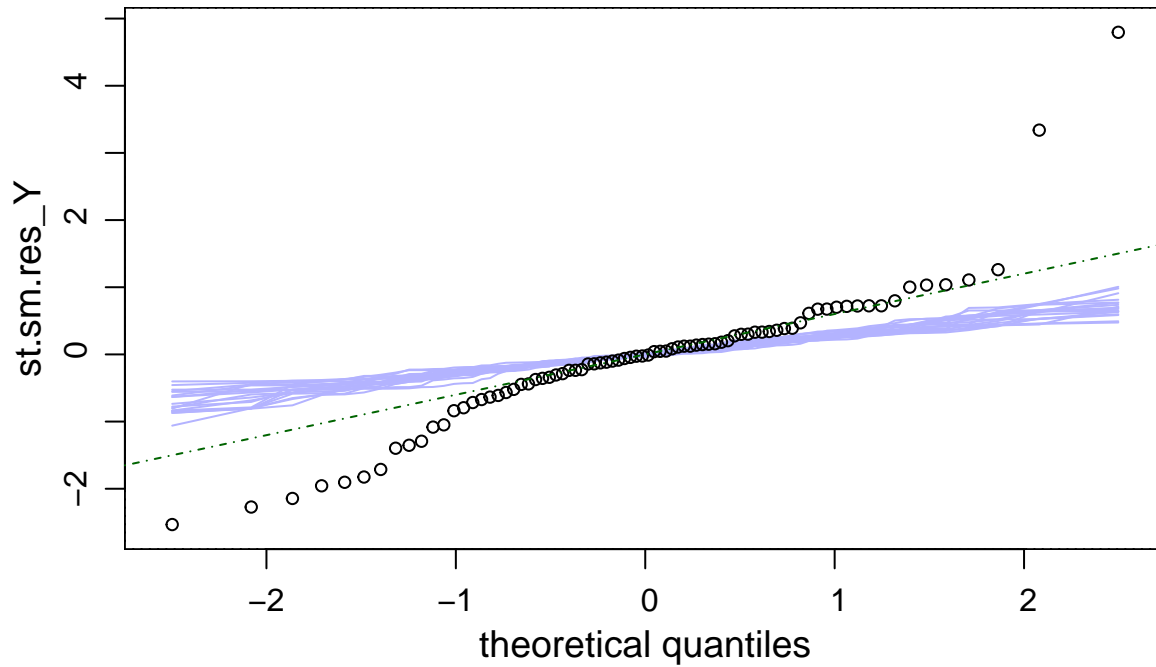
```
par(mfrow = c(1, 1))
```

The normality is doubtful in both cases, thus we use the `plregr()` function present in the `{plgraphics}` package to test whether the normality assumption makes sense.

The `plregr()` function performs a simulation to evaluate whether the actual values could be sampled from a normal distribution; this approach is particularly valuable when there are limited observations available.

```
## (warnings are omitted from this chunk)
set.seed(2023)
plregr(lm.cob_weight,
       plotselect = c(default = 0,
                      qq = 1),
       xvar = FALSE)
```

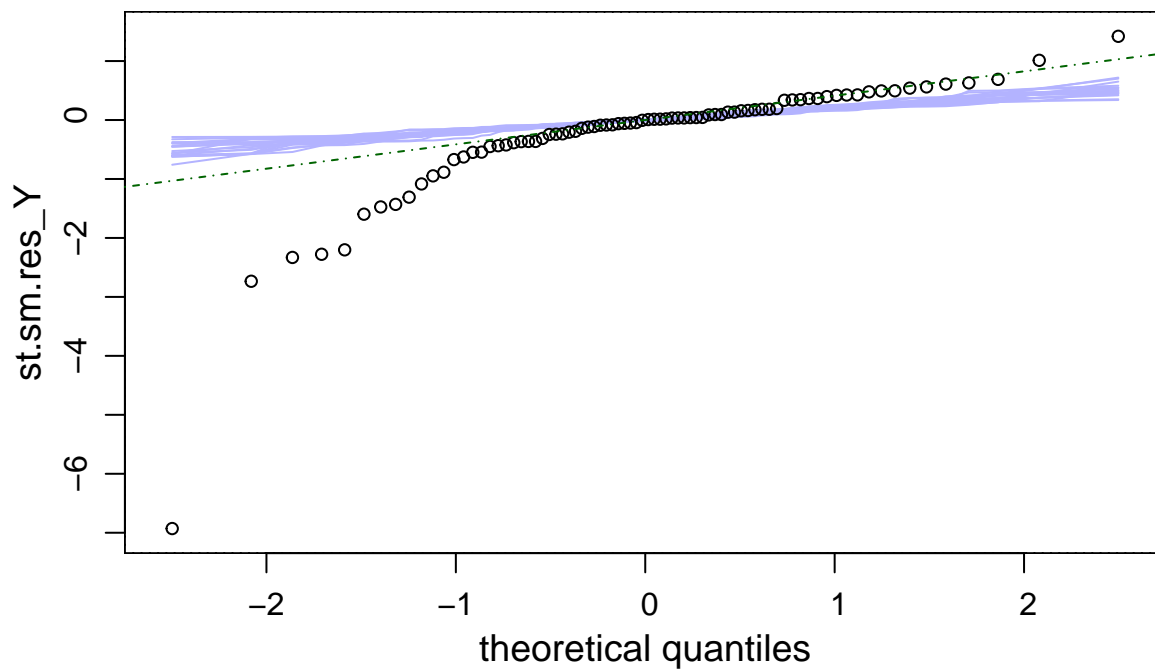
cob\_weight.num ~ pot.fac + depth + well.fac + seed.weight



```
set.seed(2023)
plregr(lm.cob_weight.log,
  plotselect = c(default = 0,
    qq = 1),
  xvar = FALSE)
```

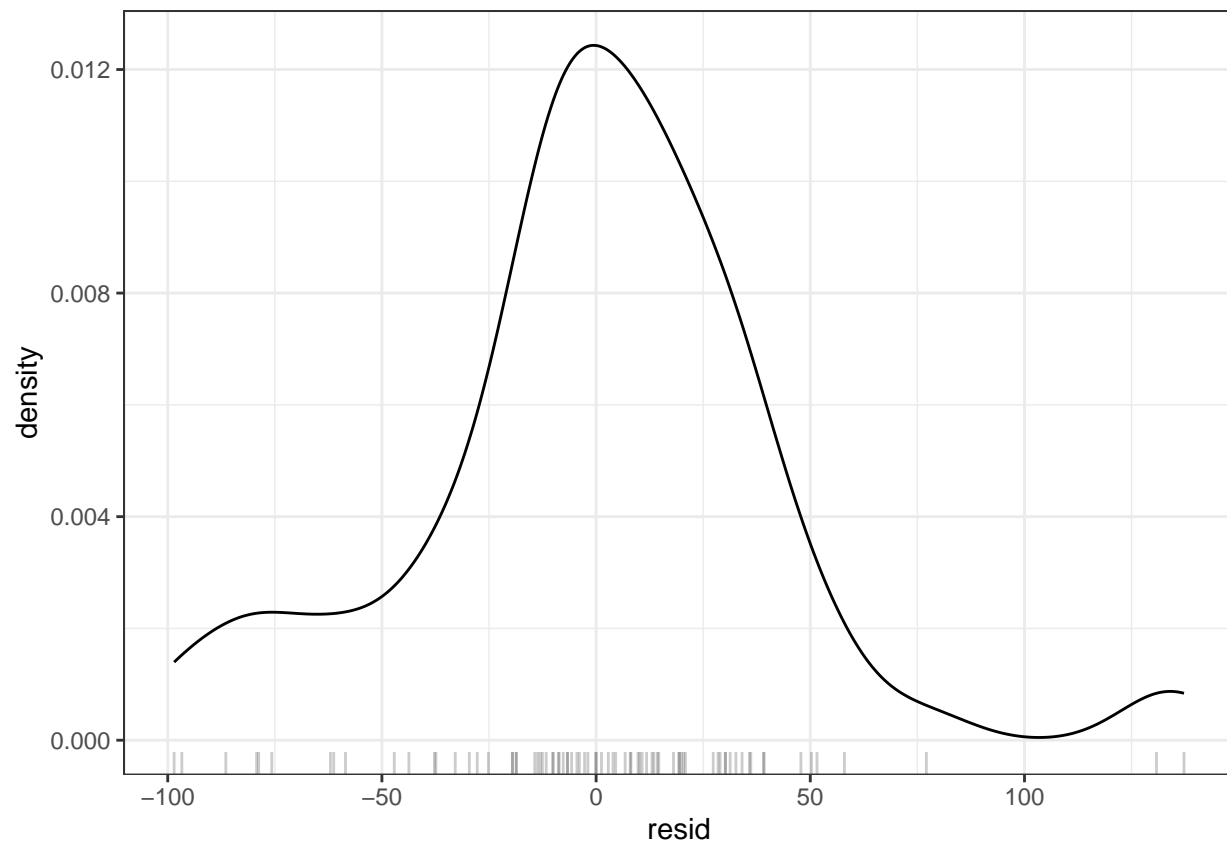
Oct 30, 23

$\log(\text{cob\_weight.num}) \sim \text{pot.fac} + \text{depth} + \text{well.fac} + \text{seed.weight}$

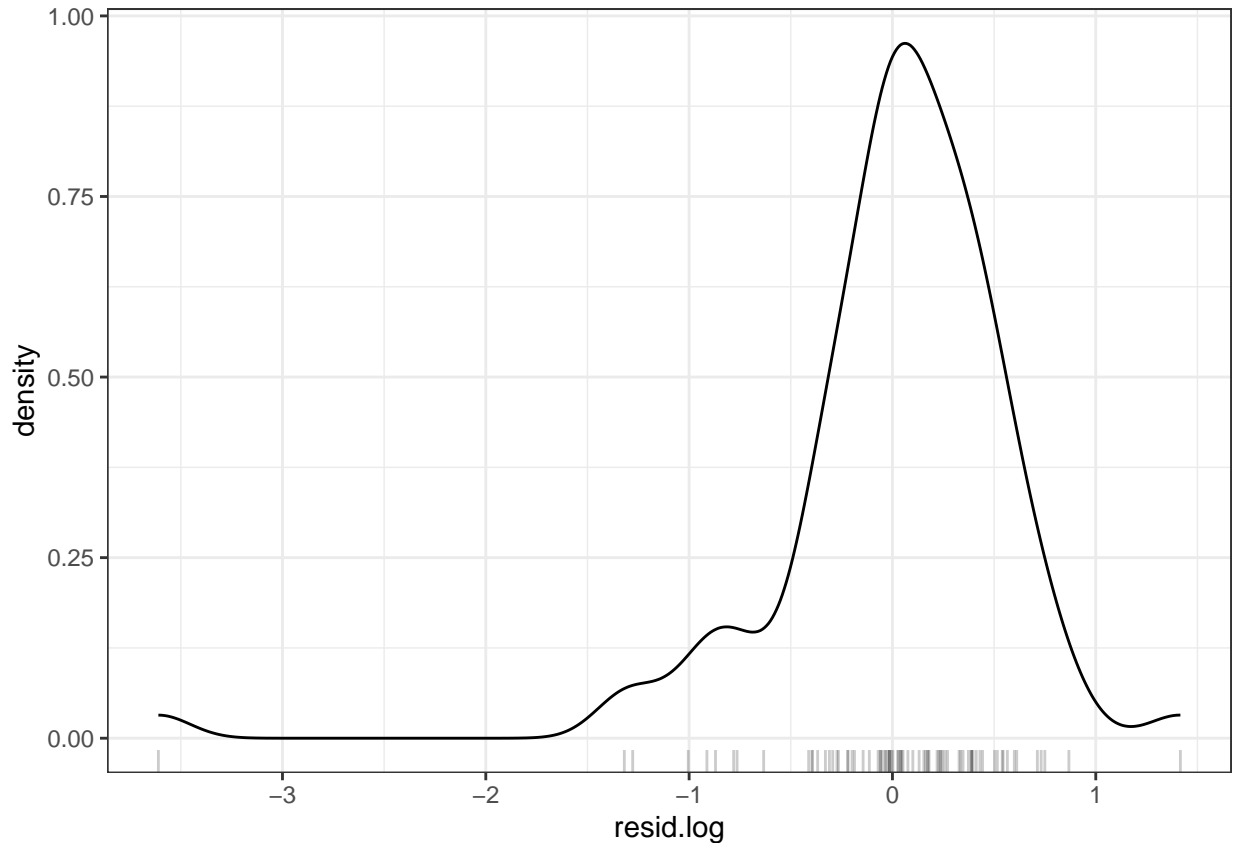


```
##  
d.maize.no.na <- d.maize %>%  
  filter(!is.na(cob_weight)) %>%  
  mutate(resid = resid(lm.cob_weight, na.action = na.omit),  
         resid.log = resid(lm.cob_weight.log, na.action = na.omit))  
  
ggplot(d.maize.no.na, mapping = aes(x = resid)) +  
  geom_density() +  
  geom_rug(alpha = 0.2)
```

Oct 30, 23



```
ggplot(d.maize.no.na, mapping = aes(x = resid.log)) +  
  geom_density() +  
  geom_rug(alpha = 0.2)
```



None of the models shows strong evidence of superiority over the other. Therefore, we opt to retain the initial fitted model without the log transformation. This choice aligns with the principle of preferring simpler models when in doubt.

The next step is to verify whether the variables *well.fac* and *pot.fac* have an influence on the response variable as a whole. Indeed, the above summary only shows the relative influence of each level compared to the reference level.

To achieve this result, we use the `drop1()` function.

```
drop1(lm.cob_weight, test = "F")
```

Single term deletions

Model:

```
cob_weight.num ~ pot.fac + depth + well.fac + seed.weight
```

	Df	Sum of Sq	RSS	AIC	F value	Pr(>F)
<none>			138523	646.541		
pot.fac	17	37010.5	175534	631.485	0.86440	0.61569
depth	1	714.2	139238	644.953	0.28356	0.59652
well.fac	5	12896.7	151420	643.663	1.02412	0.41257
seed.weight	1	351.6	138875	644.744	0.13958	0.71013

Dropping any of the explanatory variable does not seem to have an influence on the model.

### 6.3.1 Confidence intervals

We calculate the confidence intervals.

```
( CI.cob_weight <- confint(lm.cob_weight) )
```

	2.5 %	97.5 %
(Intercept)	-15.4373063	192.4273530
pot.facA2	-37.3345943	97.7766303
pot.facA3	-38.1422881	88.3333777
pot.facA4	-42.8238063	94.9043306
pot.facA5	1.5534012	135.5249745
pot.facA6	-41.6201515	105.4568398
pot.facB1	-58.3672399	65.4309230
pot.facB2	-24.3140281	108.4511901
pot.facB3	-50.0029776	82.2900888
pot.facB4	-36.7168932	100.1052174
pot.facB5	-53.1534405	76.6048550
pot.facB6	-146.4649852	24.6955813
pot.facC1	-39.4474837	108.0335775
pot.facC2	-63.6421104	67.6942529
pot.facC3	-74.2985501	62.9750619
pot.facC4	-48.9344409	92.7759453
pot.facC5	-46.2792357	75.0585808
pot.facC6	-70.8935626	63.8017226
depth	-5.3436066	9.2109998
well.facb	-13.3221486	72.2488625
well.facc	-7.5466951	70.3411167
well.facd	-44.4601109	39.2515221
well.face	-16.5316570	60.9209417
well.facf	-16.0764418	60.4261713
seed.weight	-2.8438366	1.9501109

Firstly, we create a data frame that includes the parameter estimates and their corresponding confidence intervals.

```
## Store the estimated values as dataframe
( d.coef.cob_weight <- data.frame(coef.cob_weight = coef(lm.cob_weight)) )
```

	coef.cob_weight
(Intercept)	88.49502333
pot.facA2	30.22101797
pot.facA3	25.09554477
pot.facA4	26.04026216
pot.facA5	68.53918781
pot.facA6	31.91834414
pot.facB1	3.53184158
pot.facB2	42.06858098
pot.facB3	16.14355559
pot.facB4	31.69416206
pot.facB5	11.72570723
pot.facB6	-60.88470196
pot.facC1	34.29304690
pot.facC2	2.02607125
pot.facC3	-5.66174410
pot.facC4	21.92075221
pot.facC5	14.38967257
pot.facC6	-3.54591999
depth	1.93369658

```

well.facb      29.46335697
well.facc      31.39721081
well.facd      -2.60429440
well.face      22.19464236
well.facf      22.17486473
seed.weight    -0.44686283

##
d.CI.cob_weight <- as.data.frame(CI.cob_weight)

## Join the two dataframe by rowname
d.est.cob_weight <- left_join(rownames_to_column(d.coef.cob_weight),
                              rownames_to_column(d.CI.cob_weight),
                              by = c("rowname" = "rowname"))

##
## visualise the dataframe
d.est.cob_weight %>%
  kable(caption = paste0("Estimates and 95\\% CI."),
        label = "tab_coef_cob_weight",
        booktabs = TRUE,
        longtable = TRUE,
        linesep = c("")) %>%
  # landscape() %>%
  kable_styling(font_size = 7,
                latex_options = c("striped", "repeat_header", "hold_position"))

```

Table 1: Estimates and 95% CI.

rowname	coef.cob_weight	2.5 %	97.5 %
(Intercept)	88.49502333	-15.4373063	192.4273530
pot.facA2	30.22101797	-37.3345944	97.7766303
pot.facA3	25.09554477	-38.1422881	88.3333777
pot.facA4	26.04026216	-42.8238063	94.9043306
pot.facA5	68.53918781	1.5534012	135.5249745
pot.facA6	31.91834414	-41.6201515	105.4568398
pot.facB1	3.53184158	-58.3672399	65.4309230
pot.facB2	42.06858098	-24.3140281	108.4511901
pot.facB3	16.14355559	-50.0029776	82.2900888
pot.facB4	31.69416206	-36.7168932	100.1052174
pot.facB5	11.72570723	-53.1534405	76.6048550
pot.facB6	-60.88470196	-146.4649852	24.6955813
pot.facC1	34.29304690	-39.4474837	108.0335775
pot.facC2	2.02607125	-63.6421104	67.6942529
pot.facC3	-5.66174410	-74.2985501	62.9750619
pot.facC4	21.92075221	-48.9344409	92.7759453
pot.facC5	14.38967257	-46.2792357	75.0585808
pot.facC6	-3.54591999	-70.8935626	63.8017226
depth	1.93369658	-5.3436066	9.2109998
well.facb	29.46335697	-13.3221486	72.2488625
well.facc	31.39721081	-7.5466951	70.3411167
well.facd	-2.60429440	-44.4601109	39.2515221
well.face	22.19464236	-16.5316570	60.9209417
well.facf	22.17486473	-16.0764418	60.4261713
seed.weight	-0.44686283	-2.8438366	1.9501109

We provide an example on how to read this table.

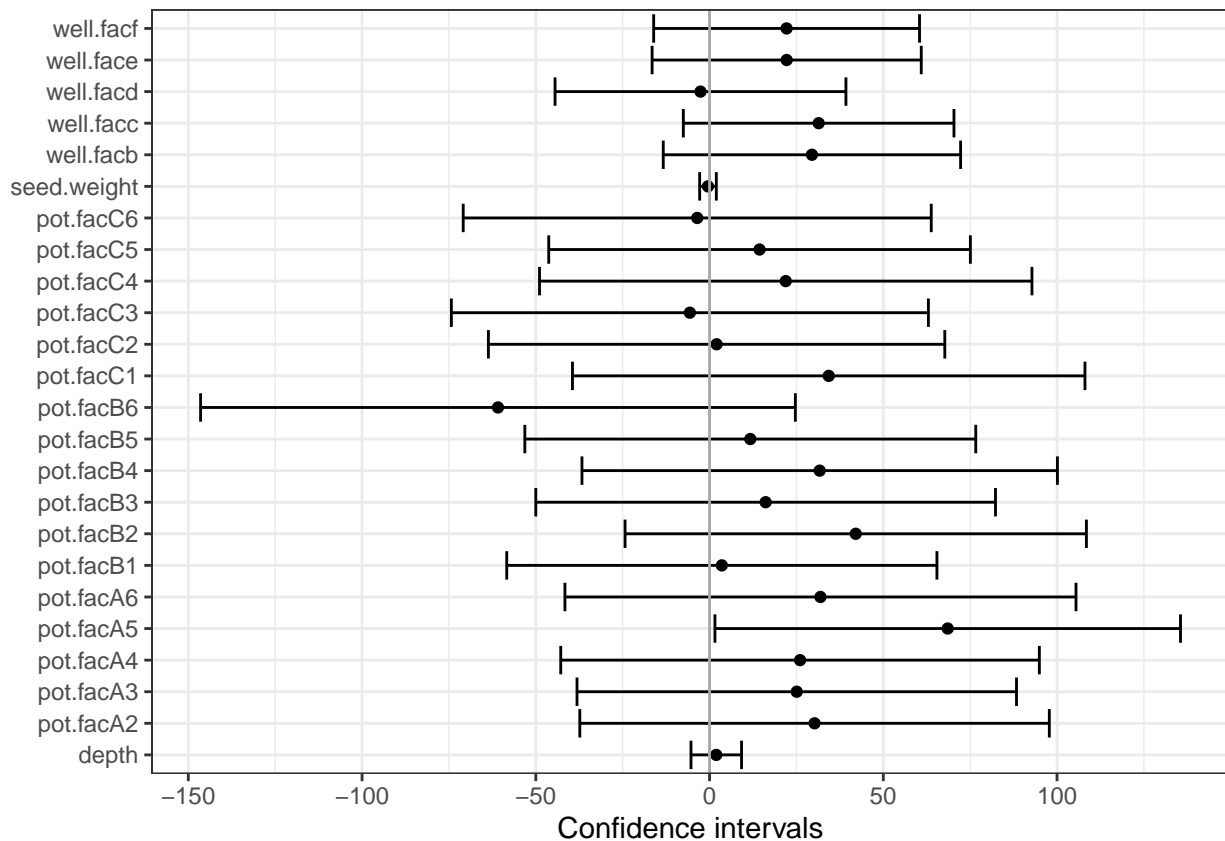
An increase of 1 cm in the variable *depth* is associated with a 1.93369658 grams increase in cob weight.

We visualise the estimated values of the model along with their corresponding confidence intervals. This will provide us with a better understanding of which variables have a more significant influence on the variable

*cob\_weight*.

We do not plot the intercept because it lacks practical interpretation. The intercept represents the reference level, corresponding to a cob with seed weight of 0 gr., which does not make any sense.

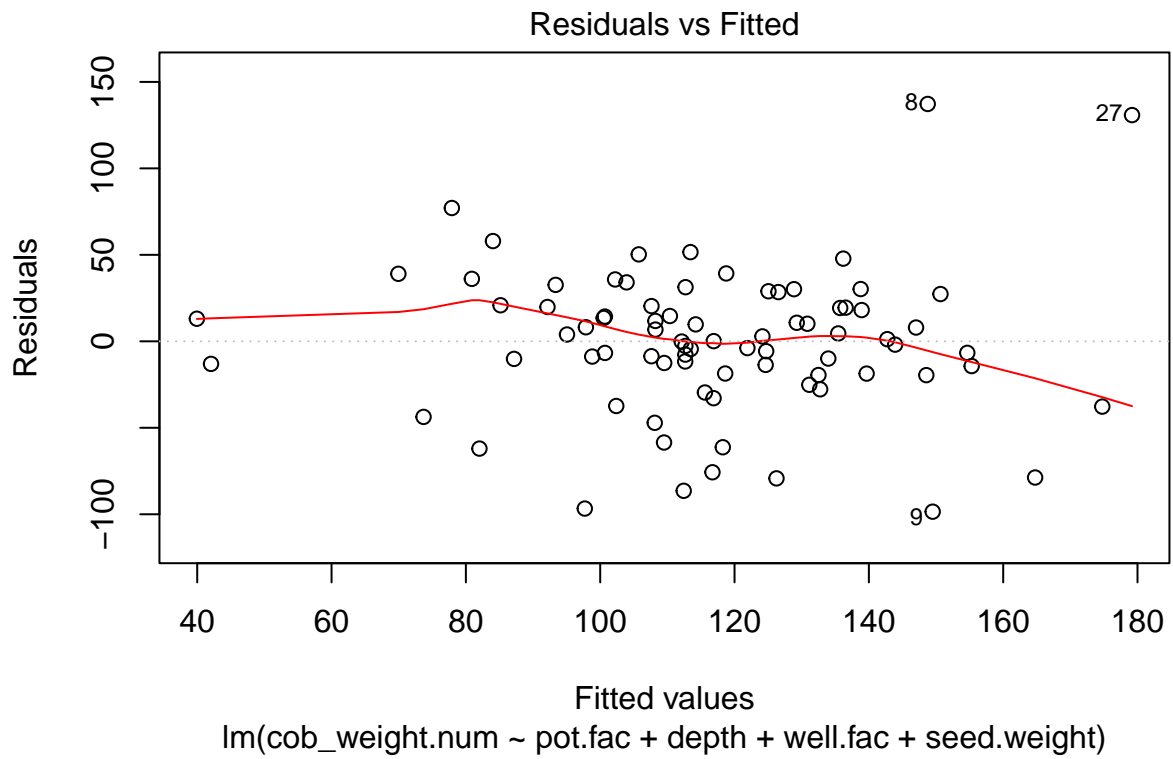
```
d.est.cob_weight %>%  
  filter(rowname != "(Intercept)") %>%  
  ggplot(mapping = aes(y = rowname, x = coef.cob_weight)) +  
  geom_point() +  
  geom_errorbar(mapping = aes(xmin = `2.5 %`, xmax = `97.5 %`)) +  
  xlab("Confidence intervals") +  
  theme(axis.title.y = element_blank()) +  
  geom_vline(xintercept = 0, color = "darkgrey")
```

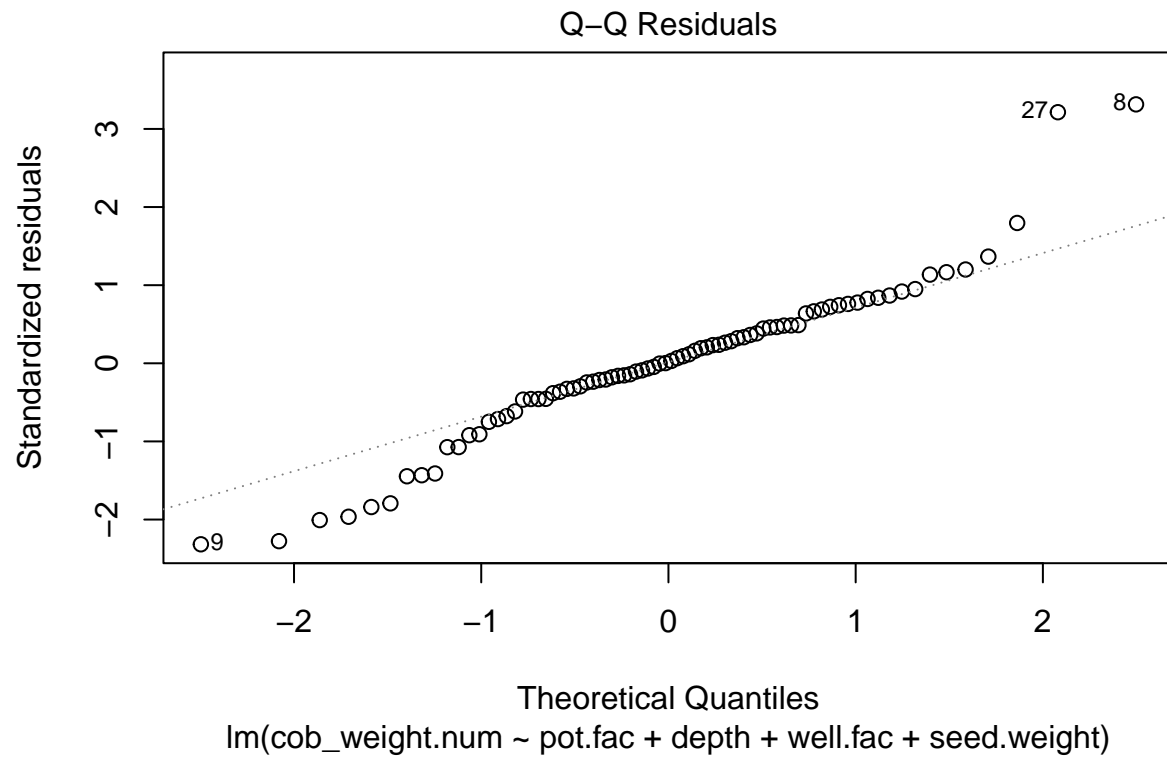


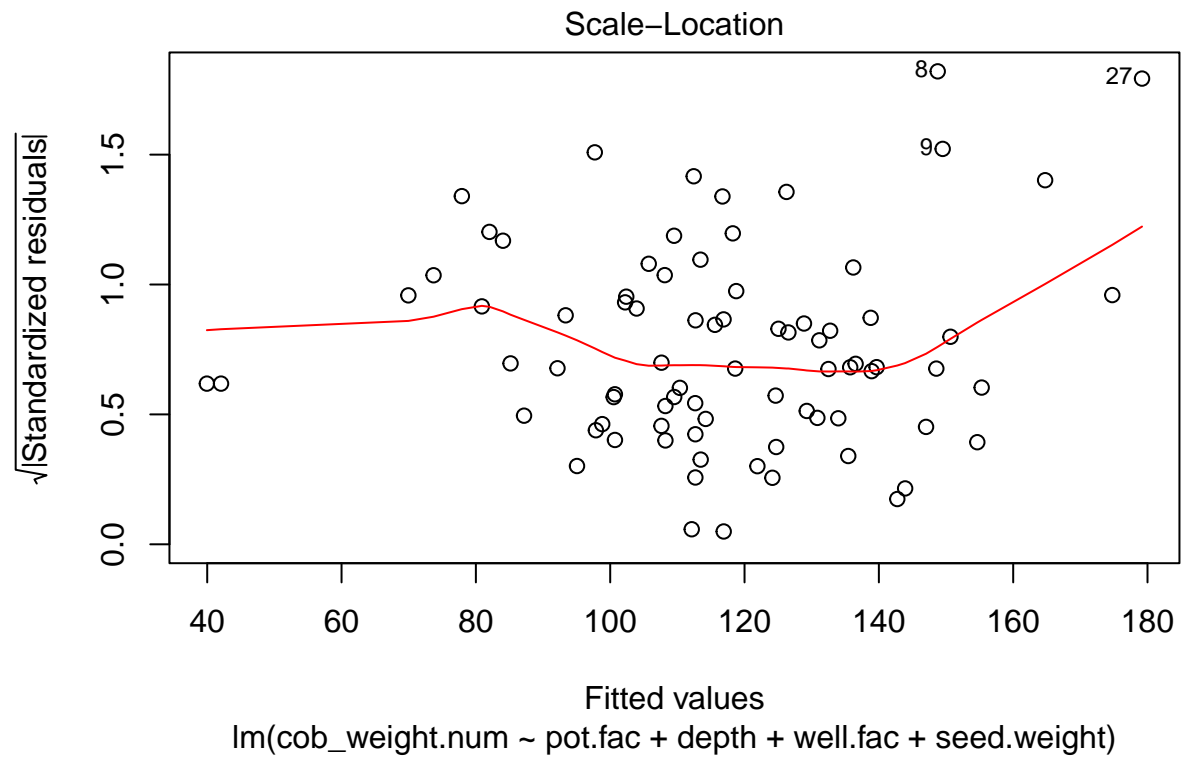
## 6.4 Model checking

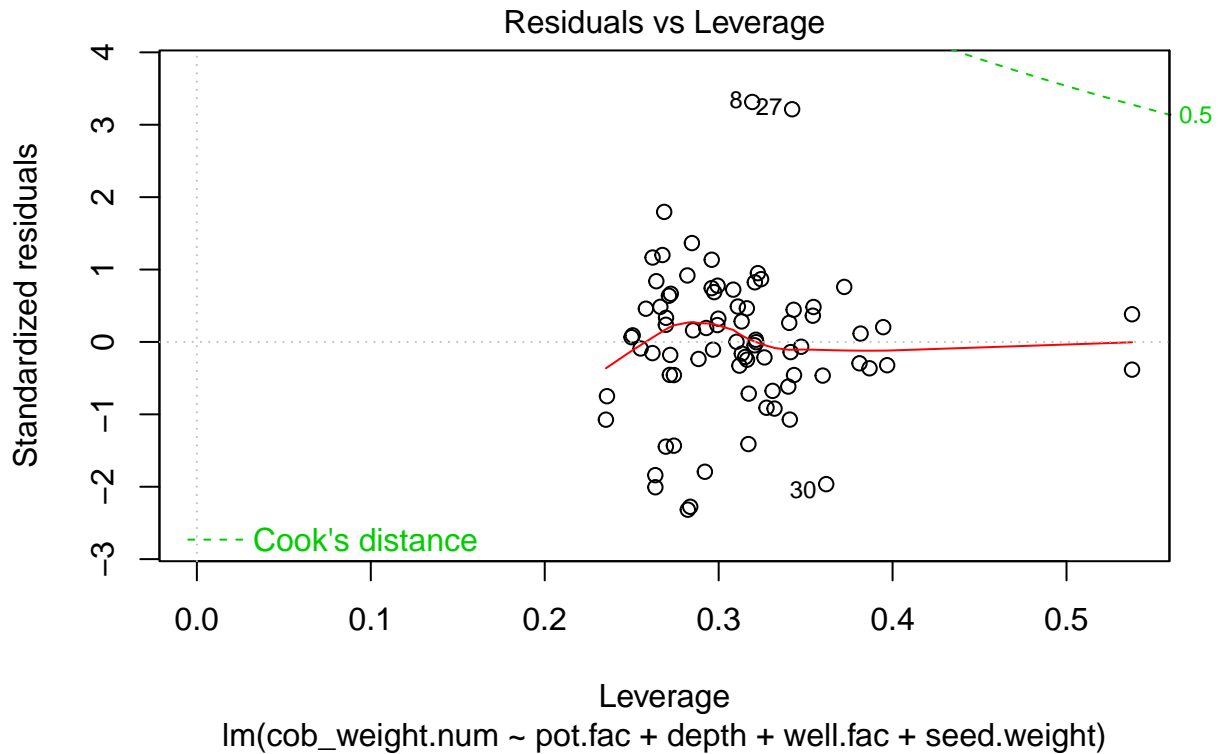
In the following, we check the hypotheses of the model.

```
plot(lm.cob_weight)
```







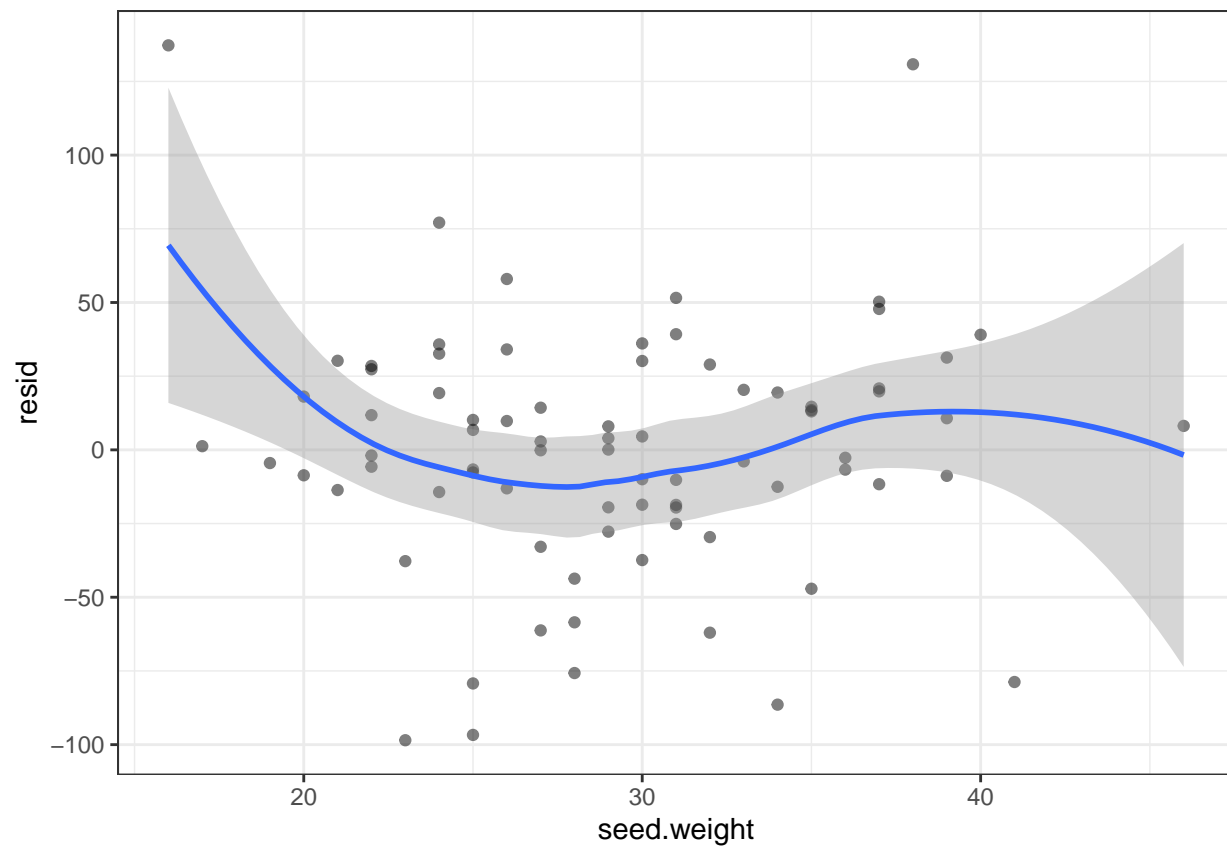


It seems that for the small residuals there is a departure from the normality assumption, but as already tested, applying a log transformation would not improve our model.

Next, we plot the residuals against each explanatory variable.

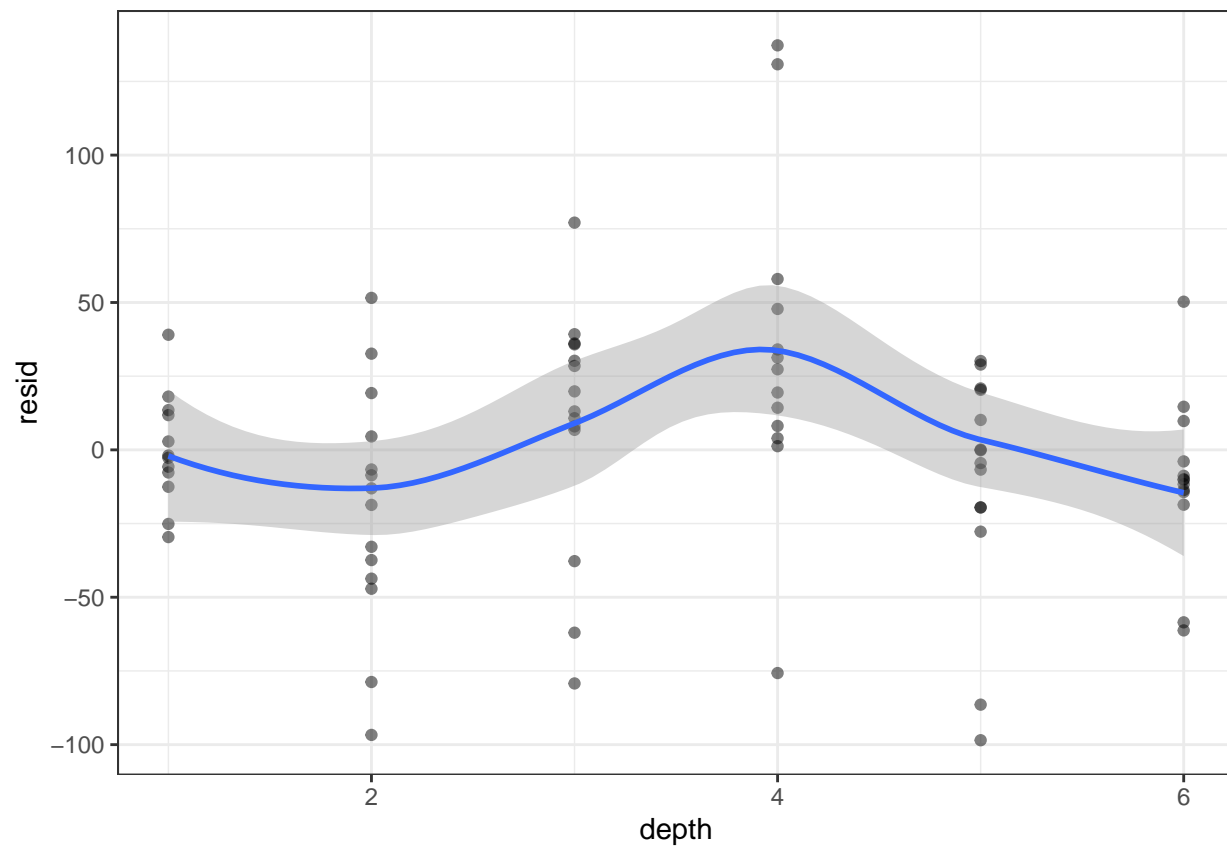
```
ggplot(d.maize.no.na, mapping = aes(x = seed.weight, y = resid)) +
  geom_point(alpha = 0.5) +
  geom_smooth()
```

`geom\_smooth()` using method = 'loess' and formula = 'y ~ x'

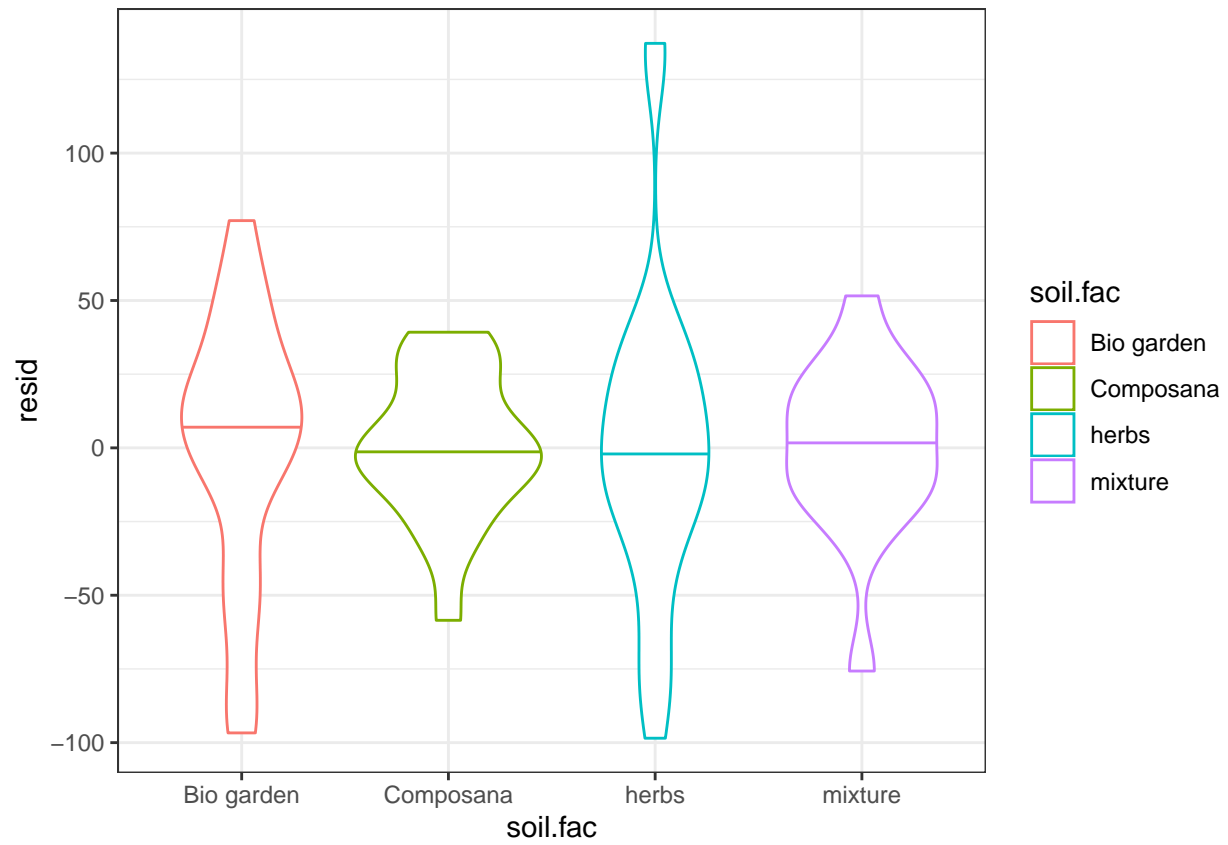


```
ggplot(d.maize.no.na, mapping = aes(x = depth, y = resid)) +  
  geom_point(alpha = 0.5) +  
  geom_smooth()
```

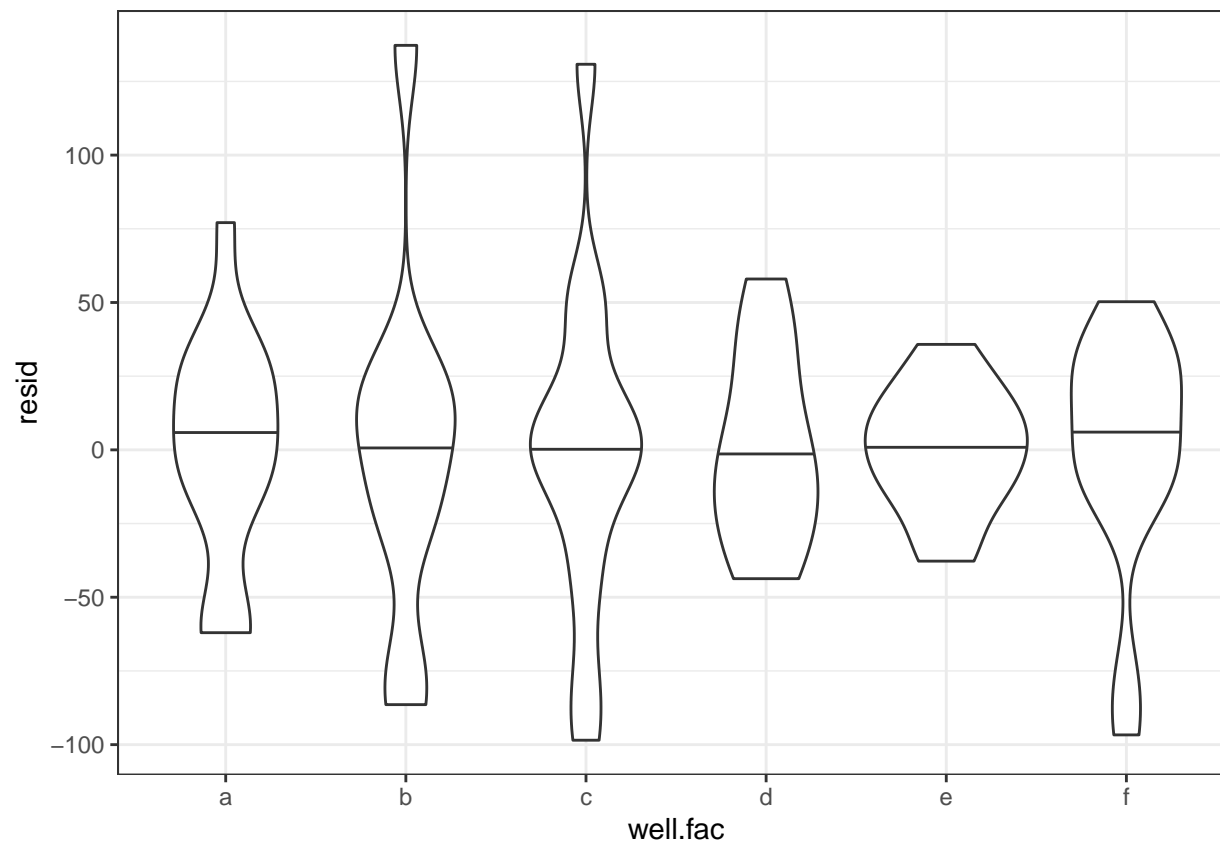
`geom\_smooth()` using method = 'loess' and formula = 'y ~ x'



```
ggplot(d.maize.no.na, mapping = aes(x = soil.fac,
                                     y = resid,
                                     colour = soil.fac)) +
  geom_violin(draw_quantiles = 0.5)
```



```
ggplot(d.maize.no.na, mapping = aes(x = well.fac,
                                     y = resid)) +
  geom_violin(draw_quantiles = 0.5)
```



There does not seem to remain any unexplained important patterns in the residuals.

## 6.5 Contrasts

We now want to further investigate the soil effect.

*soil.fac* is nested in *pot.fac*, thus it is possible to reconstruct *soil.fac* using *pot.fac*.

Indeed, we can see this in the following table.

```
xtabs(formula = ~ pot.fac + soil.fac, data = d.maize)
```

	soil.fac			
pot.fac	Bio garden	Composana	herbs	mixture
A1	6	0	0	0
A2	0	0	6	0
A3	0	0	6	0
A4	6	0	0	0
A5	0	0	6	0
A6	0	0	6	0
B1	0	0	0	6
B2	0	0	0	6
B3	0	6	0	0
B4	0	6	0	0
B5	0	0	6	0
B6	0	0	0	6
C1	0	0	0	6
C2	6	0	0	0

C3	6	0	0	0
C4	0	6	0	0
C5	0	6	0	0
C6	0	0	6	0

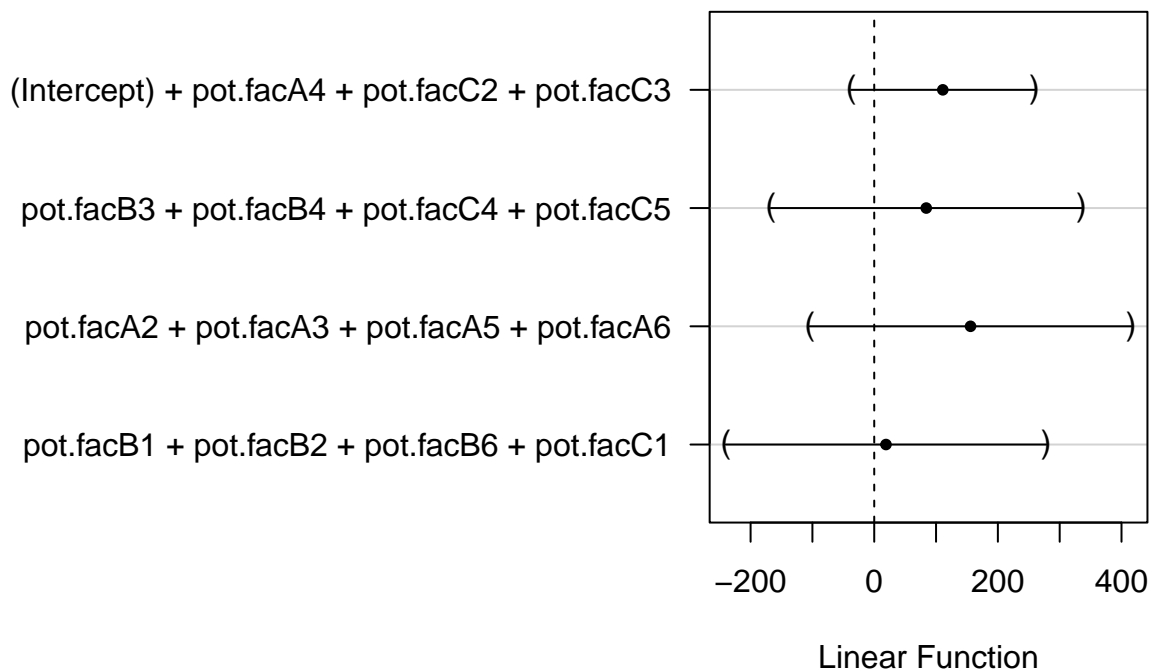
The Bio garden soil is entirely determined by pots A1, A4, C2, C3. The Composana soil is entirely determined by pots B3, B4, C4, C5. The Herbs soil is entirely determined by pots A2, A3, A5, A6. The Mixture soil is entirely determined by pots B1, B2, B6, C1.

Consequently, we can calculate the confidence intervals for these different types of soil.

```
test.soil <- glht(lm.cob_weight,
  linfct = c("(Intercept) + pot.facA4 + pot.facC2 + pot.facC3 = 0", ## Bio Garden
    "pot.facB3 + pot.facB4 + pot.facC4 + pot.facC5 = 0", ## Composana
    "pot.facA2 + pot.facA3 + pot.facA5 + pot.facA6 = 0", ## Herbs
    "pot.facB1 + pot.facB2 + pot.facB6 + pot.facC1 = 0"))
par("mar") ## the second value refers to the left margin (to be enlarged)

[1] 5.1 4.1 4.1 2.1
par(mar = c(5.1, 19, 4.1, 2.1))
plot(test.soil)
```

## 95% family-wise confidence level



The type of soil does not seem to play a role in the explanation of the variable *cob\_weight.num*.

## 7 Methods description

To understand which factors do influence the cob weight, a linear model was employed. This approach was used because the response variable, *cob\_weight*, is almost normally distributed, thus a linear model is the simpler and more appropriate model for this kind of data.

The statistical analysis was performed using the R programming language, specifically version 4.3.1 (see citation below). The linear model was fitted with the `lm()` function in the `{stats}` add-on package (see citation below).

### Citations

```
citation()
```

To cite R in publications use:

```
R Core Team (2023). _R: A Language and Environment for Statistical
Computing_. R Foundation for Statistical Computing, Vienna, Austria.
<https://www.R-project.org/>.
```

A BibTeX entry for LaTeX users is

```
@Manual{,
  title = {R: A Language and Environment for Statistical Computing},
  author = {{R Core Team}},
  organization = {R Foundation for Statistical Computing},
  address = {Vienna, Austria},
  year = {2023},
  url = {https://www.R-project.org/},
}
```

We have invested a lot of time and effort in creating R, please cite it when using it for data analysis. See also 'citation("pkgname")' for citing R packages.

```
citation("stats")
```

The 'stats' package is part of R. To cite R in publications use:

```
R Core Team (2023). _R: A Language and Environment for Statistical
Computing_. R Foundation for Statistical Computing, Vienna, Austria.
<https://www.R-project.org/>.
```

A BibTeX entry for LaTeX users is

```
@Manual{,
  title = {R: A Language and Environment for Statistical Computing},
  author = {{R Core Team}},
  organization = {R Foundation for Statistical Computing},
  address = {Vienna, Austria},
  year = {2023},
  url = {https://www.R-project.org/},
}
```

We have invested a lot of time and effort in creating R, please cite it when using it for data analysis. See also 'citation("pkgname")' for citing R packages.

## 8 Session information

```
sessionInfo()
```

```
R version 4.3.1 (2023-06-16)
```

```
Platform: aarch64-apple-darwin20 (64-bit)
```

```
Running under: macOS Sonoma 14.0
```

```
Matrix products: default
```

```
BLAS: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
```

```
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib; LAPACK vers
```

```
locale:
```

```
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
time zone: Europe/Zurich
```

```
tzcode source: internal
```

```
attached base packages:
```

```
[1] stats      graphics  grDevices  utils      datasets  methods    base
```

```
other attached packages:
```

```
[1] plgraphics_1.2    tibble_3.2.1      ggplot2_3.4.4     kableExtra_1.3.4  
[5] dplyr_1.1.3       multcomp_1.4-25   TH.data_1.1-2     MASS_7.3-60  
[9] survival_3.5-5    mvtnorm_1.2-3     knitr_1.44
```

```
loaded via a namespace (and not attached):
```

```
[1] sandwich_3.0-2    utf8_1.2.4        generics_0.1.3    robustbase_0.99-0  
[5] xml2_1.3.5        stringi_1.7.12    lattice_0.21-8    lme4_1.1-34  
[9] digest_0.6.33     magrittr_2.0.3    evaluate_0.22     grid_4.3.1  
[13] fastmap_1.1.1     Matrix_1.6-1.1    mgcv_1.8-42       httr_1.4.7  
[17] rvest_1.0.3       fansi_1.0.5       viridisLite_0.4.2 scales_1.2.1  
[21] codetools_0.2-19 cli_3.6.1         rlang_1.1.1       munsell_0.5.0  
[25] splines_4.3.1     withr_2.5.1       yaml_2.3.7        tools_4.3.1  
[29] nloptr_2.0.3      minqa_1.2.6       colorspace_2.1-0  webshot_0.5.5  
[33] boot_1.3-28.1     vctrs_0.6.4       R6_2.5.1          zoo_1.8-12  
[37] lifecycle_1.0.3   stringr_1.5.0     pkgconfig_2.0.3   pillar_1.9.0  
[41] gtable_0.3.4      Rcpp_1.0.11       glue_1.6.2        systemfonts_1.0.5  
[45] DEoptimR_1.1-3    xfun_0.40         tidyselect_1.2.0  rstudioapi_0.15.0  
[49] farver_2.1.1      nlme_3.1-162      htmltools_0.5.6.1 labeling_0.4.3  
[53] rmarkdown_2.25    svglite_2.1.2     compiler_4.3.1    chron_2.3-61
```