Package 'Salsa'

March 21, 2022

Type Package

Title Data mining facilities for Capsicum gene expression profiles

Version 1.0

Date 2022-03-21

Author Alan Flores-Diaz, Christian Escoto-Sandoval and Octavio Martinez

Maintainer <octavio.martinez@cinvestav.mx>

Description Salsa is an auto-contained R package with data and functions to explore the transcriptome of 12 chili pepper (Capsicum annuum L.) accessions during fruit development. Data consist of curated gene expression profiles for more than 29000 genes sampled every ten days from 0 to 60 Days After Anthesis (DAA), incorporating also gene identification plus Gene Ontology (GO) annotations. Functions include extensive resources for expression pattern searching and plotting facilities, GO enrichment analysis as well as methods to evaluate expression pattern differences in sets of genes and accessions. In this version we added the Gene2Gene algorithm that allows the estimation of Gene Functional Networks.

License GPL-3

LazyData TRUE

R topics documented:

Salsa-package 2
acc 4
all.GO 5
analyze.2.SEPs 6
analyze.all.GO 7
analyze.g2g 9
analyze.GO 10
browse.gene 12
browse.GO 13
core.g2g 14
DAA 15
expected 16
FPKM.expr 17
g2g 18
g2g.plot.winers 21
g2g.TFcandidates 23
gene 25
gene.summary 26

get.genes.by.desc . . . 27
get.genes.by.GO.desc . . . 28
get.GO.terms.by.desc . . . 30
get.ids . . . 31
get.mat.from.ids . . . 33
get.SEP . . . 34
GO.annot . . . 36
library.desc . . . 37
pred.error.g2g . . . 38
readcounts . . . 39
SEP . . . 45
SEP.id . . . 46
SEP.summary . . . 48
SEPs.plot . . . 50

Index **52**

`Salsa-package` *Data mining facilities for Capsicum gene expression profiles*

Description

Salsa is an auto-contained R package with data and functions to explore the transcriptome of 12 chili pepper (Capsicum annuum L.) accessions during fruit development. Data consist of curated gene expression profiles for more than 29000 genes sampled every ten days from 0 to 60 Days After Anthesis (DAA), incorporating also gene identification plus Gene Ontology (GO) annotations. Functions include extensive resources for expression pattern searching and plotting facilities, GO enrichment analysis as well as methods to evaluate expression pattern differences in sets of genes and accessions. In this version we added the Gene2Gene algorithm that allows the estimation of Gene Functional Networks.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Author(s)

Alan Flores-Diaz, Christian Escoto-Sandoval and Octavio Martinez

Maintainer: <octavio.martinez@cinvestav.mx>

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
## Assume that the interest is on genes
# of Transcription Factors (TF) in two accessions
# "CM" (Domesticated) and "QU" (Wild)
# Isolate SEPs with the genes of interest
temp.TF.CM <- get.SEP(acc.key="CM", isTF=TRUE)
temp.TF.QU <- get.SEP(acc.key="QU", isTF=TRUE)
temp.TF.CM.QU <- get.SEP(acc.key=c("CM", "QU"), isTF=TRUE)

# Obtain a summary of the content of the
# three SEPs
SEP.summary(temp.TF.CM.QU, temp.TF.CM,
temp.TF.QU, conf.level=0.99)

# Plot the corresponding SEPs to have an idea
# of mean expression per time (DAA)
SEPs.plot(list(temp.TF.CM.QU, temp.TF.CM,
temp.TF.QU), colors=c("grey", "red", "blue"),
conf.level=0.99)
title(main="Transcription Factors in QU and CM")
legend("topright", legend=c("CM and QU", "CM", "QU"),
col=c("grey", "red", "blue"), pch=1, lwd=2)

# Using the previously Isolated SEPs of all
# TFs, Isolate only the ones that have "GATA"
# in their description
temp.GATA.CM <- get.SEP(descr="GATA", previous.sep=temp.TF.CM)
temp.GATA.QU <- get.SEP(descr="GATA", previous.sep=temp.TF.QU)

# Perform a test of Euclidean distances between
# SEPs of GATA TFs in accessions CM and QU
analyze.2.SEPs(temp.GATA.CM, temp.GATA.QU, conf.level=0.99)

# Clean the temporal objects created
rm(list=c(ls(patt="temp.TF"), ls(patt="temp.GATA")))

# Obtain the gene ids of all
# GATA TFs which are expressed
# in all 12 accessions
ids.GATA <- get.ids(descr="GATA", ExistInAll=TRUE, isTF=TRUE)
length(ids.GATA)

# Look for GO terms of aspect BP
# which contain "cell cycle",
# "negative regulation", and "mitotic"
# within their description.
get.GO.terms.by.desc("cell cycle",
"negative regulation",
"mitotic", only.aspect="BP")

# Now perform a GO enrichment
# analysis having as target the
# genes in "ids.GATA"
analyze.GO(ids.GATA, aspect="BP", aspect.id=1814)

# See two of the descriptions of the
```

```
# GATA TFs in the gene data.frame.
head(gene[is.element(gene$id, ids.GATA), ], 2)

# Remove the temp object
rm("ids.GATA")

# Finally browse in NCBI the data about
# the first one of these two genes.
browse.gene(ProtId="XP_016577838.1")

# Also browse in GO the meaning of the
# GO term "GO:0045930"
# (previously analyzed)
browse.GO(GO="GO:0045930")
```

---

acc *Accessions*

Description

Accessions (genotypes) available in the data

Usage

```
data("acc")
```

Format

A data frame with 12 observations on the following 3 variables.

`acc.key` a character vector of two letters coding the 12 accessions.

`acc.type` a character vector; D = Domesticated, W = Wild, C = An F1 cross between D and W parents.

`acc.name` a character vector; Common name of the accession.

Details

All 12 accessions are of the specie Capsicum annuum, the 4 W accessions belong to the sub-specie var. glabriusculum (chle piquin or chiltepin).

Source

Cinvestav Irapuato, Mexico.

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process." Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
acc[acc$acc.type=="W", ] # Data for Wild accessions
```

all.GO *Gene Ontology (GO) annotations available*

Description

Aspects, keys and descriptions of GO annotations for the chili pepper genes.

Usage

```
data("all.GO")
```

Format

A data frame with 4518 observations on the following 4 variables.

`aspect` a character vector with values "BP" = Biological processes, "CC" = Cell Component or "MF" = Molecular Function.

`aspect.id` a numeric vector with identifiers for each GO annotation.

`GO` a character vector with the GO identifiers.

`GO.desc` a character vector with the description for each GO term.

Details

This data.frame is used in GO enrichment analyses.

Source

http://geneontology.org/

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process." Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
# A random row of the data.frame
all.GO[sample(c(1:nrow(all.GO)), size=1), ]
```

`analyze.2.SEPs` *Test two SEPs trough Euclidean distances*

Description

Mean Euclidean distances within and between the two SEPs are analyzed via the t-test to decide if there is a global difference in the expression profiles contained within the two sets of SEPs. Null hypothesis is that the mean distances within and between SEPs are equal, while the one-tail alternative hypothesis is that the mean distance between SEPs is larger than the mean distance within SEPs. This last possibility implies that the two SEPs are globally different.

Usage

```
analyze.2.SEPs(sep1, sep2, conf.level = 0.95, do.print = TRUE)
```

Arguments

`sep1`	A SEP `data.frame` obtained with the function `get.SEP`
`sep2`	A SEP `data.frame` obtained with the function `get.SEP`
`conf.level`	Confidence level to perform the `t-test`
`do.print`	`logical`. If `TRUE` the function prints the results, otherwise invisible returns a `data.frame` with main results

Details

Let's `n1` and `n2` be the number of rows of `sep1` and `sep2`, respectively. Then the number of distances within SEPs is given by `(n1*(n1-1/2)) + (n2*(n2-1/2))`, while the number of distances between SEPs is given by `n1*n2`.

Value

The function prints results of the t-test if `do.print = TRUE` and invisible returns a numeric vactor with named components

`n.sep1`	Number of SEPs in sep1
`n.sep2`	Number of SEPs in sep2
`t.value`	Value of the t statistic
`df`	Degrees of freedom for the test
`p-value`	P value reached by the test
`conf.level`	Confidence level
`L.CL`	Lower Confidence Limit of the Confidence Interval
`mean.dis.bet`	Mean distance between SEPs
`mean.dis.wit`	Mean distance within SEPs

Note

This univariate test solves the problem of performing multiple tests (one for each one of the seven time points sampled), but is sensitive to outliers.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

`get.SEP`, `SEP.summary`.

Examples

```
# SEPs of gene with id=3 in "D" versus
# SEPs of gene with id=3 in "W" or "C"
analyze.2.SEPs(sep1=get.SEP(id=3, acc.type="D"),
sep2=get.SEP(id=3, acc.type=c("W", "C")))

# Test of SEPs for two different genes
analyze.2.SEPs(sep1=get.SEP(id=3),
sep2=get.SEP(id=19))
```

---

analyze.all.GO *General GO enrichment analysis*

Description

Performs a general GO enrichment analysis for a set of genes identified by their ids and for all the terms in a given GO aspect. The analysis is performed using the Fisher's exact test. Optionally filters results per a False Discovery Rate (FDR).

Usage

```
analyze.all.GO(ids, aspect = "BP", only.FDR.le = NULL)
```

Arguments

`ids`	(`numeric`) Set of target gene identifiers to form the 2 x 2 contingency table to be tested.
`aspect`	Must be one of the valid aspects of GO: "BP" (Biological Process) or "CC" (Cell Component) or "MF" (Molecular function).
`only.FDR.le`	`NULL` (if no filtering of the results is desired) or a number between 0 and 1 with the threshold for the maximum False Discovery Rate (FDR) to be reported.

Details

Using the `ids` and EACH ONE of the terms of the GO aspect, a contingency table is formed by crossing the criteria Annotated and Target (in each case FALSE or TRUE) and the resulting 2 x 2 contingency table is analyzed using Fisher's exact test under the null hypothesis of criteria independence. Results are returned in a `data.frame` with rows corresponding to each GO aspect (if not filtered by FDR, i.e., if `only.FDR.le = NULL` (the default) or only with rows that fulfill the FDR asked. If there are no rows that fulfill the FDR criterion, a `data.frame` with 0 rows is returned without warning.

Value

A data.frame with the following columns

`aspect`	GO aspect analyzed.
`aspect.id`	Numerical identifier of the GO aspect.
`desc`	Description of the GO aspect analyzed.
`odds`	Odds ratio for the contingency table.
`P`	P value for the test of independence.
`AnnTarg`	Annotated with aspect and in Target set
`NotAnnTarg`	Not Annotated with aspect and in Target set
`AnnNotTarg`	Annotated with aspect and Not in Target set
`NotAnnNotTarg`	Not Annotated with aspect and Not in Target set

Note

Running can take some time, be patient. The total of genes taken into account corresponds to the total of genes with valid expression in the data and which are annotated in the GO aspect.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

`analyze.GO`, `get.ids`, `get.GO.terms.by.desc`, `browse.GO`.

Examples

```
# A dummy set of ids that will give
# "significant" results.
my.temp.ids <- c(unique(get.genes.by.GO.desc(
"mitochondrial outer membrane translocase complex"
, only.aspect="CC")$id), gene$id[1:26])
```

```
# Calling the function with that set of ids
# and filtering by a minimum FDR of 1%
# (can take some time)
my.temp.res <- analyze.all.GO(ids=my.temp.ids,
aspect = "CC", only.FDR.le = 0.01)

# See some of the results
head(my.temp.res)
tail(my.temp.res)

# Clean temporary objects
rm(my.temp.ids, my.temp.res)
```

`analyze.g2g` *Analysis of results from the Gene2Gene algorithm*

Description

Having the results of applying the Gene2Gene algorithm, via the `g2g` function, the function `analyze.g2g` perform and prints an analysis of the results.

Usage

```
analyze.g2g(x)
```

Arguments

`x`	Must be an object resulting from running function `g2g`.

Value

No value is returned; the function is rune by its side effects, which are to print a summary of the analysis of the object in the input.

Note

If you want to keep the results in a text file, you could for example use the function `sink()`. See examples.

Author(s)

O. Martinez

References

A constructive method to estimate Gene Functional Networks demonstrated in chili pepper. (In preparation).

See Also

`core.g2g`, `g2g`.

Examples

```
# Obtain an object suitable to be analyzed
# (genes annotated in "response to virus" are employed)
vir.ids <- GO.annot$id[GO.annot$GO=="GO:0009615"]
vir.ids <- intersect(vir.ids, get.SEP(ExistInAll=TRUE)$id)
vir.core <- core.g2g(ids=vir.ids)
vir.g2g.12 <- g2g(vir.core, min.fdr=0.1, min.r2=0.7, n.min.acc=12)
# Run the analyze.g2g with that output
analyze.g2g(vir.g2g.12)

## Do not run
# If you want to send the results to a file
# named "my_results.txt" you could use the
# three lines below:
# sink("my_results.txt")
# analyze.g2g(vir.g2g.12)
# sink()
## Ends do not run.

## Remove the objects that we created here
rm(list=ls(patt="vir."))
```

`analyze.GO` *GO enrichment analysis*

Description

Performs a GO enrichment analysis for a set of genes identified by their ids. The analysis is performed using the Fisher's exact test.

Usage

```
analyze.GO(ids, aspect, aspect.id, print.all = TRUE)
```

Arguments

`ids`	(`numeric`) Set of target gene identifiers to form the 2 x 2 contingency table to be tested.
`aspect`	Must be one of the valid aspects of GO: "BP" (Biological Process) or "CC" (Cell Component) or "MF" (Molecular function).
`aspect.id`	Numeric identifier of the aspect.
`print.all`	`logic`; `TRUE` will print the results, `FALSE` will only invisibly return the results.

Details

A contingency table is formed by crossing the criteria Annotated and Target (in each case FALSE or TRUE) and the resulting 2 x 2 contingency table is analyzed using Fisher's exact test under the null hypothesis of criteria independence. By default results are printed by the function and a data.frame with the results is invisibly returned.

Value

A data.frame with the following columns

aspect	GO aspect analyzed.
aspect.id	Numerical identifier of the GO aspect.
desc	Description of the GO aspect analyzed.
odds	Odds ratio for the contingency table.
P	P value for the test of independence.
AnnTarg	Annotated with aspect and in Target set
NotAnnTarg	Not Annotated with aspect and in Target set
AnnNotTarg	Annotated with aspect and Not in Target set
NotAnnNotTarg	Not Annotated with aspect and Not in Target set

Note

The total of genes taken into account corresponds to the total of genes with valid expression in the data and which are annotated in the GO aspect.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

analyze.all.GO, get.ids, get.GO.terms.by.desc, browse.GO.

Examples

```
# An example with the first 1000 gene identifiers
# (a fully arbitraty set)
analyze.GO(ids=gene$id[1:1000])

# Performs a GO enrichment analysis in
# a set of 100 genes taken at random.
# Also GO aspect and term are random
temp.asp <- sample(c("BP", "CC", "MF"), size=1)
analyze.GO(ids=sample(gene$id, size=100), aspect=temp.asp, aspect.id=sample(all.GO$aspect.id[all.GO$aspect=
rm(temp.asp)

# Same random experiment but this time
# with 1000 genes
temp.asp <- sample(c("BP", "CC", "MF"), size=1)
analyze.GO(ids=sample(gene$id, size=1000), aspect=temp.asp, aspect.id=sample(all.GO$aspect.id[all.GO$aspect
rm(temp.asp)
```

`browse.gene` *Opens a window in your browser*

Description

The input of the function is a protein identifier for a gene in the data (`ProtId` in the `data.frame` gene). It will cause to open a window containing the information of the NCBI that exist about that protein.

Usage

```
browse.gene(ProtId = "XP_016567627.1")
```

Arguments

`ProtId`	One of the protein identifiers existent in the `data.frame` gene

Value

The function is used by its side effect (open a window in your web browser)

Author(s)

O. MArtinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

`gene`, `browse.gene`, `get.genes.by.desc`.

Examples

```
# If you have a gene of interest, say
gene[gene$id==35802,]
# You can obtain information of
# that protein in the NCBI:
browse.gene("XP_016555511.1")
```

`browse.GO` *Opens an URL in the page corresponding to the GO identifier*

Description

Opens in your web browser the URL in THE GENE ONTOLOGY RESOURCE page corresponding to the GO identifier in the input.

Usage

```
browse.GO(GO = "GO:0000002")
```

Arguments

`GO`	Valid `GO` are the ones in the column `GO` of the data.frame `all.GO`

Details

See http://www.informatics.jax.org/vocab/gene_ontology/

Value

The function is used by its side effect (to open a web page).

Note

This function is a convenient wrapper for the function `browseURL`.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

`get.GO.terms.by.desc`

Examples

```
# Opens the page corresponding to GO:0000003 (Reproduction)
# (in your web browser)
browse.GO("GO:0000003")
```

core.g2g *Calculates statistics to apply the Gene2Gene algorithm*

Description

From an arbitrary set of gene identifiers, the function calculates all necessary statistics to apply the Gene2Gene algorithm. You can give the accession keys that you want to employ; by default all 12 accessions will be employed by the procedure. The result of this function is the main input to function g2g.

Usage

```
core.g2g(ids, in.acc = NULL)
```

Arguments

ids	An integer vector with the identifiers (i.e., the values of id) that are used within the package.
in.acc	Either, NULL, or a vector of accessions identifiers (the two letters that identify every one of the 12 accessions; see acc$acc.key). Any subset of acc$acc.key is accepted. If the value of in.acc is NULL (the default), then all 12 accessions are employed.

Details

The function uses the cor.test function to calculate Pearson's correlations between the Standardized Expression Profiles (SEPs) for all possible pairs of genes (id), within the same accession. Also the value of the maximum of the absolute difference (m.a) in every pair will be part of the output. Beware: The function takes time that is square in the number of identifiers in ids. Thus, if more than 500 identifiers are in ids, it could take hours of finish!

Value

A list with 4 components:

ids	Identical to the input ids (vector with the gene identifiers used)
in.acc	The vector of accession identifiers employed.
seps	Data frame with the SEPs (SEP) of the genes and identifiers combinations, except cases where the model was fully uniform (model=="SSSSSS")
res.corr	A data.frame with variables: acc - Accession key; id1 - Identifier of the first gene in the pair; id2 - Identifier of the second gene in the pair; r - Value of the Pearson's r correlation coefficient; p.r - p-value given by the cor.test function; m.a - Estimated value of the maximum absolute and r.pos - (logical) TRUE if r is positive and FALSE otherwise.

Author(s)

O. Martinez

References

A constructive method to estimate Gene Functional Networks demonstrated in chili pepper. (In preparation).

See Also

g2g

Examples

```
# A small example (three genes in two accessions)
temp <- core.g2g(ids=c(3545, 13178, 33679), c("CM", "AS"))
# See the names of the result:
names(temp)
# See the first rows of the main result:
head(temp$res.corr)
# Remove "temp"
rm(temp)
```

DAA *Days After Anthesis*

Description

Time points where all accessions were sampled in all 12 accessions.

Usage

```
data("DAA")
```

Format

The format is: num [1:7] 0 10 20 30 40 50 60

Details

DAA is used for plotting and analysis of data.

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
DAA # Just the time points of sampling.
```

`expected` *Expected value of a contingency table*

Description

Returns the expected value for a contingency table under the hypothesis of independence.

Usage

```
expected(x)
```

Arguments

`x`	A matrix (of observed values)

Details

Mainly for internal use.

Value

A matrix of the same size of `x` with the expected values under independence.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

`get.mat.from.ids`, `analyze.GO`

Examples

```
# An arbitrary matrix
matrix(c(1:6), nrow=2, ncol=3)
# And its expected value under independence
expected(matrix(c(1:6), nrow=2, ncol=3))
# Used internally as
expected(get.mat.from.ids())
```

FPKM.expr *Expression data per accession and time in FPKM units*

Description

Contains mean expression data for each gene (id) and each accession (acc.key) at each one of the seven times sampled (mT00 to mT60) in FPKM (fragments per kilobase of exon model per million reads mapped).

Usage

```
data("FPKM.expr")
```

Format

A data frame with 313919 observations on the following 14 variables.

id a numeric vector; gene numeric identifier.

acc.key a character vector; two letters code for accessions.

acc.type a character vector; D = Domesticated, W = Wild.

model a character vector; six letters code for the expression model (see details).

ExistInAll a logic; If TRUE the gene was expressed in all 12 accessions.

mT00 a numeric vector; FPKM expression at 0 DAA.

mT10 a numeric vector; FPKM expression at 10 DAA.

mT20 a numeric vector; FPKM expression at 20 DAA.

mT30 a numeric vector; FPKM expression at 30 DAA.

mT40 a numeric vector; FPKM expression at 40 DAA.

mT50 a numeric vector; FPKM expression at 50 DAA.

mT60 a numeric vector; FPKM expression at 60 DAA.

mean.over.time a numeric vector; mean over times of gene expression.

coded.expr.level a numeric vector; code for expression levels, see details.

Details

FPKM (fragments per kilobase of exon model per million reads mapped) is a normalised estimation of gene expression based on RNA-seq data. Column model is formed by all the 729 permutations of the letters D, S, I, meaning that the expression in the corresponding interval was Decreasing, Steady or Increasing, respectively. Column coded.expr.level codes mean expression level with the following meaning: 1 expression less or equal to 1 FPKM, 2 more than 1 but less than 10 FPKM, 3 more than 10 but less than 100 FPKM, 4 more than 100 but less than 1000 FPKM and 5 more than 1000 FPKM.

Source

Cinvestav Irapuato, Mexico.

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
# A random row of the data.frame
FPKM.expr[sample(c(1:nrow(FPKM.expr)), size=1), ]
# A gene with high expression level (an average expression larger than 1000 FPKM)
head(FPKM.expr[FPKM.expr$coded.expr.level==5,], 1)
# What is codded by this gene?
gene[gene$id==102, ]
# Plot of the expression of gene with
# id=102 in accession "AS"
plot(DAA, FPKM.expr[(FPKM.expr$id==102)&(FPKM.expr$acc.key=="AS"), 6:12], type="b", ylab="Expression in FPKM
# A line marking the mean expression.
abline(h=2413.108, col="grey")
```

---

g2g *Implements the Gene2Gene algorithm*

Description

From the output of the `core.g2g(ids, in.acc)` function, `g2g()` obtains a set of gene pairs which fulfill given input thresholds for correlations, and are concordantly found in a minimum number of accessions determined by the user. If the original `ids` belong to a given Biological Process (BP), then the output could represent a Gene Functional Network (GFN) for such BP.

Usage

```
g2g(core, min.fdr = 0.05, min.r2 = 0.7, m.a.quan = 0.05, n.min.acc = 6)
```

Arguments

`core`	An object resulting from running the function `core.g2g(ids, in.acc)`.
`min.fdr`	Threshold (in proportion) for the minimum False Discovery Rate (FDR) that the researcher wants for the correlation results.
`min.r2`	Threshold for the minimum `r^2` (squared Pearson's correlation coefficient) that the researcher wants in the relations found.
`m.a.quan`	Threshold for the maximum absolute (`m.a`) quantile that the researcher wants to eliminate from output. This threshold prevents outliers, that even when having large absolute values of correlation, are highly influenced by a single point in time.
`n.min.acc`	Minimum number of independent accessions where the gene pairs must be found (fulfilling `min.fdr`, `min.r2` and `m.a.quan` simultaneously) to appear in the final result.

Details

The core of the Gene2Gene algorithm, implemented in the g2g function, is to find highly alike gene expression profiles (SEP) which are consistently found in a group of independent accessions. Even when the raw results of this function could be confusing at a first sight, because they contain many elements, we include other functions to analyze and plot results.

Value

A list with five components:

ids	Numerical identifiers id of the genes included in the input.
parameters	data.frame with the values of the parameters input to the function.
results.gw	A list with intermediate results, with components win.cand and real.quant. win.cand is a data.frame which contain results for all pair of genes that passed the thresholds imposed by the input. real.quant is a numeric factor with the values of thresholds for the m.a statistics that were employed to let out possible outliers using parameter m.a.quan.
results.cw	A list with two components: winners and paired.seps. winners is a data.frame with all characteristics for the winners, i.e., for all relations that passed the thresholds imposed in the input. paired.seps is a list of paired SEPs for each one of the winners.
results.gn	Is a list of two components: mat4graph and gene.annot. mat4graph is a matrix of the edges of the graph. It could be used in the package igraph to obtain a representation of the estimated network. gene.annot is a data.frame with the annotations of the genes that are included into the winners, i.e., genes that are found to be related in the network.

Note

You could use different sets of thresholds to run g2g. The first step is to have a set of gene identifiers, ids, which are annotated into a biological process of interest. Note that it will be unwise to have many identifiers in this set; usually if you have 500 or less gene identifiers the function core.g2g will run in a reasonable amount of time. After having the output of the core.g2g function with your set of identifiers, you can run g2g with different sets of parameters. It will be advisable to begin by asking relations that are confirmed in all 12 available accessions, by setting the parameter n.min.acc=12 in the input. Depending of the results, you could perform further runs of g2g with lower values of n.min.acc. However, to obtain reasonably robust results, the minimum value that you could try is around n.min.acc=4.

Author(s)

O. Martinez

References

A constructive method to estimate Gene Functional Networks demonstrated in chili pepper. (In preparation).

See Also

core.g2g, analyze.g2g, g2g.plot.winers, pred.error.g2g, g2g.TFcandidates.

Examples

```
# Assume that you are interested in the BP "response to virus"
# We are going to estimate a Gene Functional Network of the genes
# annotated in that BP in Capsicum.
# Note: execute row by row examining results if you want to understand.

# See the description of the GO BP "response to virus"
all.GO[all.GO$GO.desc=="response to virus",]

# We need to obtain the identifiers of the genes annotated in that BP
vir.ids <- GO.annot$id[GO.annot$GO=="GO:0009615"]
# But only the ones that are consistently expressed in all accessions
vir.ids <- intersect(vir.ids, get.SEP(ExistInAll=TRUE)$id)
# How many genes are we going to study?
length(vir.ids)

# Now find the correlations between all pairs of these genes
# within each one of the 12 accessions:
vir.core <- core.g2g(ids=vir.ids)

# In theory we may have 12*(33*(33-1)/2)=6336 rows in the main
# result. However, cases where the expression profiles were
# uniform (model="SSSSSS") were automatically excluded.
# Let's summarize the results in vir.core:
names(vir.core) # A list with those components
length(vir.core$ids) # Number of genes initially included
vir.core$in.acc # Which accessions were studied
nrow(vir.core$seps) # Number of (not null) time profiles studied
names(vir.core$res.corr) # Main results
nrow(vir.core$res.corr) # Number of rows in main results

# Number of cases of r positive and negative
table(vir.core$res.corr$r.pos)

# Summary of correlation coefficient (r)
# classified by "r.pos"
tapply(vir.core$res.corr$r, vir.core$res.corr$r.pos, summary)

# Summary of p-value of correlation coefficient (r)
summary(vir.core$res.corr$p.r)

# Summary of maximum absolute (m.a)
summary(vir.core$res.corr$m.a)

# Now we want to obtain a Gene Functional Network (GFN)
# for genes that will have very simular time expression
# profiles in all 12 accessions.
# We use the previous object (vir.core) and will as for
# thresholds min.fdr=0.1, min.r2=0.7 and n.min.acc=12
vir.g2g.12 <- g2g(vir.core, min.fdr=0.1, min.r2=0.7, n.min.acc=12)

# Let's have a quick look to the results:
names(vir.g2g.12) # Names of the components.
vir.g2g.12$parameters # Parameters employed:

# Third component (intermediate results)
```

```
names(vir.g2g.12$results.gw)
# Number of pairs of genes per accession that passed thresholds
nrow(vir.g2g.12$results.gw$win.can)
# Realized threshold for m.a (based on quantiles)
vir.g2g.12$results.gw$real.quant

# Fourth component ("winners" and "paired.seps")
# (summary of results after selecting winners)
names(vir.g2g.12$results.cw)
# Number of relations (edges) that were confirmed
# in all 12 accessions
nrow(vir.g2g.12$results.cw$winners)
# Names of that data.frame
names(vir.g2g.12$results.cw$winners)
# See this data.frame which contains averages for
# the main parameters.
vir.g2g.12$results.cw$winners
# A summary of the main results in winners
summary(vir.g2g.12$results.cw$winners[,c(7,9,10)])
# List of paired profiles (SEPs) for all winners:
class(vir.g2g.12$results.cw$paired.seps)
length(vir.g2g.12$results.cw$paired.seps)
names(vir.g2g.12$results.cw$paired.seps)
# (those results are used to plot winners)

# Fifth component
# Matrix of the edges of a graph and annotation of genes
names(vir.g2g.12$results.gn)
# Matrix of the edges of the graph
# (suitable to be used by the "igraph" package)
vir.g2g.12$results.gn$mat4graph
# Annotations of the "winner" genes
vir.g2g.12$results.gn$gene.annot

# Analysis of the results
analyze.g2g(vir.g2g.12)

## Remove the objects that we created here
rm(list=ls(patt="vir."))
```

`g2g.plot.winers` *Plots one of the SEP pairs obtained by the Gene2Gene algorithm*

Description

The g2g function implements the Gene2Gene algorithm, which produces a set of relations between genes. The function `g2g.plot.winers` allows the visualization of the plot any of the expression profiles (SEPs) found by the algorithm.

Usage

```
g2g.plot.winers(x, which.winner = NULL, col1 = "red", col2 = "blue", out2pdf = FALSE)
```

Arguments

x	An object produced by the g2g function.
which.winner	Either, NULL (the default), or the tag naming one of the relations found in the input x, which in turn is the result of function g2g.
col1	A valid R color. This will be used in the plot as the color of the first member of the pair of SEPs plot.
col2	A valid R color. This will be used in the plot as the color of the second member of the pair of SEPs plot.
out2pdf	Logic (either, TRUE or FALSE). Does the user wants the function to output a PDF file containing the plot produced?

Details

If in the input the parameter which.winner is NULL the function will ask for the number of the relation that the user wants to plot. If, in the input the user put as value of the parameter which.winner any character that is NOT recognized as a valid character, the function will give an error, but (importantly), it will return the names of all valid values for the parameter which.winner. In this way the user can select the relation for which she/he wants the plot. Then, running again the function, this time with a valid value of which.winner, the user can obtain the desired plot.

Value

No value is returned. The function is run by its side effects, say, to print a summary of the genes in the pair selected and to produce a plot.

Author(s)

O. Martinez

References

A constructive method to estimate Gene Functional Networks demonstrated in chili pepper. (In preparation).

See Also

core.g2g, g2g.

Examples

```
# Obtain an object that could be used by g2g.plot.winers
# (genes annotated in "response to virus" are employed)
vir.ids <- GO.annot$id[GO.annot$GO=="GO:0009615"]
vir.ids <- intersect(vir.ids, get.SEP(ExistInAll=TRUE)$id)
vir.core <- core.g2g(ids=vir.ids)
vir.g2g.12 <- g2g(vir.core, min.fdr=0.1, min.r2=0.7, n.min.acc=12)
# Run g2g.plot.winers with one of the relations
g2g.plot.winers(vir.g2g.12, which.winner="r.pos.13178-33679")
# Examine the summary given in the R window.

## Do not run
# If you want to obtain the PDF file for that winner, run
# g2g.plot.winers(vir.g2g.12, "r.pos.13178-33679", out2pdf=TRUE)
```

```
# After the run, file "1-r.pos.13178-33679.pdf" will be in your
# directory.
#
# If you want to plot a specific winner, run
# g2g.plot.winers(vir.g2g.12)
# The function will ask you for the number
# of the winner that you want to plot.
## Ends do not run.

## Remove the objects that we created here
rm(list=ls(patt="vir."))
```

`g2g.TFcandidates` *Obtains Transcription Factor candidates to be regulating a gene*

Description

Using the time expression profiles (SEPs) of a target gene, the function obtains Transcription Factor (TF) candidates to be regulating the target gene. The function is based in the fact that TF generally have a SEP alike the one of the target gene. This bioinformatic approach is useful to select a small set of TF candidates when no prior information exists about the target gene.

Usage

```
g2g.TFcandidates(tg.id = 32062, in.acc = NULL, min.fdr = 0.1, min.r2 = 0.5, n.min.acc = 6)
```

Arguments

`tg.id`	Numeric identifier of the target gene.
`in.acc`	Either `NULL` or a set of accession identifiers that will be used by the function. If `in.acc = NULL` (the default) the 12 accessions available are employed by the function. Otherwise, `in.acc` must be a subset of `acc$acc.key`.
`min.fdr`	The maximum of the False Discovery Rate (FDR) that the user is going to allow in the correlation test.
`min.r2`	The minimum of the squared value Pearson's correlation coefficient that the user wants the TF candidate to have with the SEP of the target gene to be considered.
`n.min.acc`	The minimum number of accessions in which the correlation must be present, passing all the other thresholds, to be considered in the final list of TF candidates.

Details

When no prior knowledge of the TF that could be regulating a gene of interest exist, running this function can reduce the universe of candidates from the total number of TF that are annotated in Salsa, 1792, to a few ones. Even when there is no guarantee that the TF candidates could in fact be regulating the target gene, the usually small set of TF candidates obtained gives a good clue to continue the search.

Value

A list with components

`winn.summ` A `data.frame` with as many rows as TF candidates were found. That `data.frame` has columns: `id` - Numeric identifier for the TF candidate. `n.acc` - Number of accessions where the relation was found. `r.pos` - Logical: Is the correlation coefficient positive? (`TRUE` or `FALSE`). `acc` - Accession keys where the relation was found. Values of Pearson's correlation coefficients: `r.Min`, `r.Med`, `r.Mea`, `r.Max` (minimum, median, mean and maximum, respectively). Values for the maximum absolute statistic (`m-a`): `m.a.Min`, `m.a.Med`, `m.a.Mea`, `m.a.Max` (minimum, median, mean and maximum, respectively). Values for the q-value for the FDR threshold: `q.Min`, `q.Med`, `q.Mea`, `q.Max` (minimum, median, mean and maximum, respectively) and finally, `est.error` - Estimated error of the selection of the TF candidate.

`winners.c` A `data.frame` containing individual data per accession for each one of the statistics in the TF candidates. The columns in that `data.frame` are: `id` - Numeric identifier for the TF candidate. `acc` - Accession key where the relation was found. `r` - Value of Pearson's correlation coefficient. `p.r` - p-value from the `cor.test` function. `m.a` - Estimated maximum absolute statistic (`m.a`). `r.pos` - Logical; is `r` positive? (`TRUE` or `FALSE`). and `q.r` - Transformation of `p.r` to `q.r` to evaluate FDR.

`par` A one row `data.frame` showing the parameters used for the run.

Note

Using this function we have found TF for genes of interest that had been later confirmed to be regulating the expression of a target gene (manuscript in preparation).

Author(s)

O. Martinez

References

A constructive method to estimate Gene Functional Networks demonstrated in chili pepper. (In preparation).

See Also

`g2g`.

Examples

```
## We are going to present the example to find TF candidates
# for the gene with numeric identifier 19858, which has codes
# for a protein described as:
# glucan endo-1,3-beta-glucosidase

# Run the function asking for the most stringent value
# n.min.acc=12 (other parameters let as defaults)
temp <- g2g.TFcandidates(tg.id=19858, n.min.acc=12)

# Names of the resulting list
names(temp)
```

```
# See first the parameters used to run the function
temp$par

# How many TF candidates do we have?
nrow(temp$winn.summ)

# In this example ONLY TF candidates with positive
# correlation are found; see a brief summary of main statistics:
summary(temp$winn.summ[,c(7,11,15)])

# If you want to see the whole results
# (the 5 rows of the winners)
temp$winn.summ

# Remove the object created
rm(temp)
```

---

gene *Identifiers for genes*

Description

For each one of the expressed genes this data frame contains protein identifiers and descriptions.

Usage

```
data("gene")
```

Format

A data frame with 29946 observations on the following 4 variables.

`id` a numeric vector; numeric gene identifier.

`ProtId` a character vector; protein identifier.

`Prot.Desc` a character vector; protein description

`isTF` a logical vector; Is the coded protein a Transcription Factor?

Details

Descriptors are based in the reference genome CM334 v1.6.

Source

http://passport.pepper.snu.ac.kr/?t=PGENOME

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
# A random row of the data.frame
gene[sample(c(1:nrow(gene)), size=1), ]

# Show data for some Transcription Factors.
head(gene[gene$isTF==TRUE,])
```

`gene.summary` *Graphic and numeric summary of a gene*

Description

Gives a plot as well as a summary of mean standardized expression over time for a gene in the types of accessions where it was expressed (D, W and C).

Usage

```
gene.summary(id, leg.pos, T.col, D.col, W.col, C.col)
```

Arguments

`id`	Identifier of the gene.
`leg.pos`	Possition of the legend in the plot: one of `bottomright`, `bottom`, `bottomleft`, `left`, `topleft`, `top`, `topright`, `right` and `center`.
`T.col`	Color to be used for plotting the expression mean of all accessions.
`D.col`	Color to be used for plotting the expression mean of D accessions.
`W.col`	Color to be used for plotting the expression mean of W accessions.
`C.col`	Color to be used for plotting the expression mean of C accessions.

Details

This function is used mainly for its side effect: plotting of mean expression per group and summary of mean expression. But also it invisibly returns a list with the main results of the summary of mean expression.

Value

Invisibly returns a list with the following components

`IdDesc`	Identification of the gene.
`Acc.x.group`	Number of accessions where the gene was expressed (per group).
`Mean.r.x.group`	Mean expression correlation (r) of SEPs within each group.
`StatMaxExpx.group`	A matrix giving the mean and standard deviation (S) for the time at which the maximum expression was reached for each accession group.

Note

The plot and summary is obtained from the data in SEP.id which in turn summarizes the standardized expression profile for each one of the genes.

Author(s)

Octavio Martinez.

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

SEP.id

Examples

```
# Summary for the gene with highest mean expression in the dataset
# See: FPKM.expr[FPKM.expr$mean.over.time>17000,]
gene.summary(id = 7975, leg.pos = "topleft")

# Obtains a summary of one gene
# selecting it at random
gene.summary(id=sample(SEP.id$id, size=1))
```

---

get.genes.by.desc *Select genes that fulfill a description criteria*

Description

Shows the rows of the gene `data.frame` which include into `Prot.Desc` the character chains pass to the function.

Usage

```
get.genes.by.desc(...)
```

Arguments

`...`	One or more character strings that you want to be part of the `Prot.Desc` for the gene.

Details

All the character strings pass in the input must be present in the `Prot.Desc`, otherwise a `NULL` value is returned by the function.

Value

Either, a `NULL` value (when there are not `Prot.Desc` that fulfill the input) or a subset of the gene `data.frame` which rows in which the `Prot.Desc` column includes all terms in the input.

Note

For general descriptions the `data.frame` returned can be large.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

gene

Examples

```
# A case where the function cannot
# find a Prot.Desc that fulfills
# the search
get.genes.by.desc("my favorite gene")

# Obtain a subset of genes which includes
# "WRKY" in their description
head(get.genes.by.desc("WRKY"))
# How many of those genes are there?
nrow(get.genes.by.desc("WRKY"))

# Having also ""factor 53"
head(get.genes.by.desc("WRKY", "factor 53"))
# How many of those?
nrow(get.genes.by.desc("WRKY", "factor 53"))

# How many containing "DNA polymerase"
nrow(get.genes.by.desc("DNA polymerase"))

# A more concise search
get.genes.by.desc("DNA polymerase I")
```

`get.genes.by.GO.desc` *Select genes annotated with GO terms*

Description

Obtains a `data.frame` of all genes annotated with one or more Gene Ontology (GO) aspects and terms, returning an informative `data.frame`.

Usage

```
get.genes.by.GO.desc(..., only.aspect = NULL)
```

Arguments

...	One or more character strings that you want to be part of the description of the Gene Ontology term description.
only.aspect	If not NULL it must contain the code for a single GO aspect that you want to look for, say BP = Biological Process, CC = Cell Component or MF = Molecular function.

Details

All the character strings pass to the function must be part of one or more GO term descriptions, otherwise a NULL value is return.

Value

Either, a NULL value (when there are not GO terms that fulfill the input) or a data.frame with the following columns:

id	Gene numeric identifier
aspect	GO aspect
GO	GO term
GO.desc	Description of the GO term
ProtId	Identifier of the protein coded by the gene
Prot.Desc	Description of the protein coded by the gene
isTF	logical; Is the gene annotated as a Transcription Factor

Note

The output data.frame can be redundant, having the same genes various times, depending on how many GO aspects each gene is annotated.

Author(s)

O. MArtinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

get.GO.terms.by.desc, all.GO, analyze.GO, browse.GO, GO.annot.

Examples

```
# A case where the function cannot
# find any gene fulfilling the search
# and thus returns NULL
get.genes.by.GO.desc("my favorite GO term")

# Obtain a set of genes which include
# "mitotic cell cycle" in their GO annotations
# (see only the fist two rows)
head(get.genes.by.GO.desc("mitotic cell cycle"),2)

# How many cases are there?
nrow(get.genes.by.GO.desc("mitotic cell cycle"))
# Note: that one gene can be annotated in
# various of the GO terms:
# How many different genes ("id"s)
length(unique(get.genes.by.GO.desc("mitotic cell cycle")$id))
# How many different GO terms ("id"s)
length(unique(get.genes.by.GO.desc("mitotic cell cycle")$GO))

# Using the only.aspect parameter
nrow(get.genes.by.GO.desc("wall", only.aspect="BP"))
nrow(get.genes.by.GO.desc("wall", only.aspect="CC"))
nrow(get.genes.by.GO.desc("wall", only.aspect="MF"))
```

---

get.GO.terms.by.desc *Finds GO terms that fulfill the input*

Description

Finds all Gene Ontology (GO) terms that include in their descriptions all words pass to the function in the input.

Usage

```
get.GO.terms.by.desc(..., only.aspect = NULL)
```

Arguments

`...`	One or more words (character strings) that must be present in the GO description to be present in the output.
`only.aspect`	If not `NULL` it must contain the code for a single GO aspect that you want to look for, say BP = Biological Process, CC = Cell Component or MF = Molecular function.

Value

Either, `NULL` when the function does not find one or more of the terms pass in the input, of a `data.frame` with columns

`aspect`	GO aspect
`aspect.id`	Numeric identifier of the GO term within aspect

GO	GO identifier
GO.desc	GO descrioption

In this data.frame all cases of GO.desc include all terms pass to the function in the input.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

get.genes.by.GO.desc, all.GO, analyze.GO, browse.GO, GO.annot.

Examples

```
# When the functio does not find
# the term pass in the input:
get.genes.by.GO.desc("My favorite GO term")

# Obtains a set of GO terms which includes
# "mitotic cell cycle" in their description
# Show just the first two of them
head(get.GO.terms.by.desc("mitotic cell cycle"),2)
# How many of those?
nrow(get.GO.terms.by.desc("mitotic cell cycle"))

# A more concise search
# (asking for two terms)
get.GO.terms.by.desc("mitotic cell cycle",
"negative regulation")

# Using a specific aspect
# (only.aspect "BP", "MF" or "CC")
get.GO.terms.by.desc("wall", only.aspect="CC")
get.GO.terms.by.desc("wall", only.aspect="MF")
```

---

get.ids *Obtains a set of gene identifiers*

Description

Obtains a set of gene identifiers that fulfill one or more of the criteria pass in the input.

Usage

```
get.ids(descr = NULL, ExistInAll = NULL, isTF = NULL)
```

Arguments

descr	One or more words (character strings) that must be present in the Prot.Desc of the gene data.frame to be presented in the output of the function.
ExistInAll	A logical variable; if TRUE only genes that were expressed in all 12 accessions will be present in the output.
isTF	A logical variable; if TRUE only genes that are annotated as Transcription Factors will be present in the output.

Details

One or more of the criteria in the input must be not NULL, otherwise the function will produce an error.

Value

Either NULL, when no gene fulfills all criteria pass in the input, or a integer vector including all gene idendifiers of the genes that fulfill all criteria pass to the function in the input.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

gene, get.genes.by.desc, browse.gene.

Examples

```
# Returns NULL
# (because descr is not found in
# any gene description)
get.ids(descr="My favorite gene")

# Returns the ids of all genes expressed
# in the 12 accessions
# get.ids(ExistInAll=TRUE)
# See the first 6 of them
head(get.ids(ExistInAll=TRUE))
# How many?
length(get.ids(ExistInAll=TRUE))

# How many genes are annotated as
# Transcription Factors?
length(get.ids(isTF=TRUE))

# And how many of those include in
# the description the character
```

```
# strings "WRKY"
length(get.ids(descr="WRKY"))

# See some of the Prot.Desc of those genes
gene$Prot.Desc[is.element(gene$id, tail(get.ids(descr="WRKY")))]

# Example using multiple criteria
# of selection
get.ids(descr=c("zipper", "HAT"), ExistInAll=TRUE, isTF=TRUE)
```

`get.mat.from.ids` *2 by 2 Contingency Table*

Description

Returns a 2 by 2 Contingency Table to be used in GO enrichment analysis (mainly for internal use)

Usage

```
get.mat.from.ids(s.t, aspect, aspect.id)
```

Arguments

`s.t`	Vector of integers. The set of gene identifiers in the group which is the target of the analysis.
`aspect`	Character string. The GO aspect (BP, CC or MF)
`aspect.id`	Integer. Numeric identifier of the aspect in `all.GO`.

Details

Used by `analyze.GO`.

Value

A 2 by 2 matrix suitable for GO enrichment analysis.

Note

Used by `analyze.GO`.

Author(s)

Octavio Martinez.

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

analyze.GO, expected

Examples

```
get.mat.from.ids(s.t = SEP$id[1:1000], aspect = "BP", aspect.id = 1)
```

---

get.SEP *Obtains a SEP data frame*

Description

Obtains a Standardized Expression Profile (SEP) data frame with rows that fulfill various criteria determined by the input.

Usage

```
get.SEP(ids, descr, acc.key, acc.type, model, ExistInAll,
 TimeMaxExp, isTF, coded.expr.level, previous.sep)
```

Arguments

ids	integer. A vector with the gene identifiers (id) that must be present in the SEP.
descr	One or more words (character strings) that you want to be part of the Prot.Desc for the genes in the output.
acc.key	One or more acc.key (key for the accessions) that you want to be in the output SEP.
acc.type	One or more acc.type (key for the accession types) that you want to be in the output SEP.
model	One or more model (key for SEP models) that you want to be in the output SEP.
ExistInAll	logical. If TRUE only cases of genes expressed in all 12 accession will be present in the output SEP.
TimeMaxExp	numeric. If present must be a vector of times where the maximum expression of the gene was reached, i.e., a time existent in the vector DAA, 0, 10, 20, 30, 40, 50 and or 60.
isTF	logical. If TRUE only genes annotated as Transcription Factors will be present in the output SEP.
coded.expr.level	numeric One or more coded expression levels, i.e., numbers 1, 2, 3, 4 and 5. Only genes with those coded expression levels will be present in the output SEP.
previous.sep	If NULL the SEP used for the selection will be the original SEP present in the package, otherwise this parameter must contain a valid SEP, possibly selected with a previous call to the function. In that case such SEP will be used for the selection.

Details

This function can be used to select a subset of data for further analysis.

Value

A SEP data frame containing rows that fulfill all criteria pass to the function in the input.

Note

This function can be used in the initial data mining steps to select sets of data to be employed in further analysis.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

SEP, SEP.summary, SEPs.plot, analyze.2.SEPs, FPKM.expr.

Examples

```
# Return NULL because there is no gene
# with the description pass to the function
get.SEP(descr="MyFavorite gene")

# Obtain a SEP with all the Transcription Factors that are
# expressed in all 12 accessions.
temp.TF.all <- get.SEP(ExistInAll = TRUE, isTF = TRUE)
# How many rows has that SEP?
nrow(temp.TF.all)

# Some characteristics of that SEP
length(unique(temp.TF.all$id)) # How many different genes?
table(temp.TF.all$acc.key) # Number of cases per accession

# Now, assume that we want the Transcription Factors
# which include in the description the character "WRKY"
# and are present only in wild accessions.
# Using the previous SEP we can obtain:
temp.TF.WRKY.W <- get.SEP(descr="WRKY", acc.type="W",
previous.sep=temp.TF.all)
nrow(temp.TF.WRKY.W)

# Now assume that from that from the previous SEP we want only
# cases where the model is "DSSSSS", i.e., genes that are
# highly expressed in flower but then present an steady state
# along the remaining time intervals:
temp.TF.WRKY.W.DSSSSS <- get.SEP(model="DSSSSS",
previous.sep=temp.TF.WRKY.W)

# See the resulting SEP
```

```
# (note that all cases have the same standardized expression)
temp.TF.WRKY.W.DSSSSS

# An alternative way to obtain "temp.TF.WRKY.W.DSSSSS" with
# a single call of get.SEP is:
temp.TF.WRKY.W.DSSSSS.2 <- get.SEP(descr="WRKY", acc.type="W",
model="DSSSSS", ExistInAll=TRUE)
temp.TF.WRKY.W.DSSSSS.2

# Remove the objects that we created
rm(list=ls(patt='temp.TF'))
```

`GO.annot` *GO annotations*

Description

GO annotations for the genes.

Usage

```
data("GO.annot")
```

Format

A data frame with 784324 observations on the following 4 variables.

`aspect` a character vector where BP = Biological Process, CC = Cell Component and MF = Molecular Function.

`id` a numeric vector; gene numeric identifier.

`GO` a character vector; GO identifier.

`aspect.id` a numeric vector; the identifier of the aspect.

Source

http://geneontology.org/

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
# A random row of the data.frame
GO.annot[sample(c(1:nrow(GO.annot)), size=1), ]

head(GO.annot)
# How many annotations per aspect:
table(GO.annot$aspect)
```

library.desc *Library description*

Description

A data frame with variables describing the RNA-Seq libraries deposited in the NCBI GEO database (GEO accession number GSE165448).

Usage

```
data("library.desc")
```

Format

A data frame with 179 observations on the following 8 variables.

`name` a character vector; a six characters coding the name of the library.

`GEO.sample` a character vector; sample name of the library at GEO.

`acc.key` a character vector; two characters code for accession name.

`tissue` a character vector; source of tissue for the library: fruit or plantlet.

`time` a numeric vector; time of development in DAA.

`replicate` a numeric vector; number of biological replicate (1 or 2).

`in.Salsa` a logical vector; Are the data from this library used in Salsa?

`acc.type` a character vector; type of accession: D - Domesticated, W- Wild, C - Cross.

Details

This data frame give details of all 179 libraries, the majority of which were used to obtain the curated data that are in Salsa.

Source

Cinvestav Irapuato, Mexico. All libraries were deposited in GEO accession GSE165448 (see https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE165448)

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
data(library.desc)
head(library.desc)
# Tabulate the number of libraries per accession
table(library.desc$acc.key)
# Gives the number of libraries used in Salsa
table(library.desc$in.Salsa)
```

`pred.error.g2g` *Calculates predicted or realized error in selecting gene pairs*

Description

Based in the Binomial Distribution, this function calculates the predicted or realized error in finding a correlation between two genes significant at a fixed False Discovery Rate (FDR). When used a priory, i.e., before performing the experiment, it uses the expected FDR. However, after the experiment is realized it calculates the a posteriori or realized error in selecting gene pairs, using as parameter the q-value.

Usage

```
pred.error.g2g(n.acc = 6, tot.acc = 12, q = 0.05)
```

Arguments

`n.acc`	Number of accessions where the relation must be found; a natural number smaller or equal than the total number of accessions, `tot.acc`.
`tot.acc`	The total number of accessions employed. Here this value is always 12, but could be different for other experiments.
`q`	The expected FDR when employing the function a priory. If employed a posteriori, this parameter must be set to the mean value of the q-values found in each one of the correlation tests performed.

Details

This function is a simple application of the Binomial probability function (`dbinom`). It answers the question: What is the chance of repeatedly rejecting the null hypothesis of no correlation `n.acc` times, when performing a total of `tot.acc` tests at a FDR of `q`? Given that the accessions employed are independent, the distribution of the number of successes (rejections) is clearly Binomial, and this is the foundation of this calculation.

Value

The estimated (or realized) error probability.

Note

This function is useful to judge the robustness of a gene pair relation.

Author(s)

O. Martinez

References

A constructive method to estimate Gene Functional Networks demonstrated in chili pepper. (In preparation).

See Also

g2g, g2g.TFcandidates.

Examples

```
## Note: here the total number of accessions is 12.
# Assume that you plan to accept gene pair relations
# when they are judged to be significant at FDR of 5 percent.
# Then, the error probabilities will be equal to
pred.error.g2g(n.acc=6)
# when you will accept only relations found at 5% FDR
# in that case the likelihood of making an error
# in the selection will be approximately of one in
round(1/pred.error.g2g(n.acc=6))

# Now, if you want a FDR of only 10% and want to
# calculate the expected error of selecting relations
# that are present in 5 accessions, that predicted
# error will be given by
pred.error.g2g(n.acc=5, q=0.1)
# and the likelihood of making an error will be
# approximately of 1 in
round(1/pred.error.g2g(n.acc=5, q=0.1))
# etc.

# On the other hand, assume that you planed to
# accept all pair of relations found in 5 accessions
# at a FDR of 0.05, but the mean value found for
# the q-values was in fact 0.0035, then the
# estimated probability of that happening by chance
# will be approximately
pred.error.g2g(n.acc=5, q=0.0035)
```

---

readcounts *Raw number of reads for each gene at each RNA-Seq library.*

Description

This data frame contains the raw number of reads map to each one of the genes in the Capsicum genome. The first variable (id) is the unique gene identifier employed in Salsa, and all other 179 columns are the names of the RNA-Seq libraries. See library.desc for a description of the libraries.

Usage

```
data("readcounts")
```

Format

A data frame with 35883 observations on the following 180 variables.

`id` a numeric vector; unique gene identifier in Salsa.

`CM00R1` a numeric vector

`CM00R2` a numeric vector

`CM10R1` a numeric vector

`CM10R2` a numeric vector

`CM20R1` a numeric vector

`CM20R2` a numeric vector

`CM30R1` a numeric vector

`CM30R2` a numeric vector

`CM40R1` a numeric vector

`CM40R2` a numeric vector

`CM50R1` a numeric vector

`CM50R2` a numeric vector

`CM60R1` a numeric vector

`CM60R2` a numeric vector

`CO00R1` a numeric vector

`CO00R2` a numeric vector

`CO10R1` a numeric vector

`CO10R2` a numeric vector

`CO20R1` a numeric vector

`CO20R2` a numeric vector

`CO30R1` a numeric vector

`CO30R2` a numeric vector

`CO40R1` a numeric vector

`CO40R2` a numeric vector

`CO50R1` a numeric vector

`CO50R2` a numeric vector

`CO60R1` a numeric vector

`CO60R2` a numeric vector

`CQ00R1` a numeric vector

`CQ00R2` a numeric vector

`CQ10R1` a numeric vector

`CQ10R2` a numeric vector

`CQ20R1` a numeric vector

`CQ20R2` a numeric vector

`CQ30R1` a numeric vector

`CQ30R2` a numeric vector

`CQ40R1` a numeric vector

`CQ40R2` a numeric vector

`CQ50R1` a numeric vector

`CQ50R2` a numeric vector

`CQ60R1` a numeric vector

`CQ60R2` a numeric vector

CW00R1 a numeric vector

CW00R2 a numeric vector

CW10R1 a numeric vector

CW10R2 a numeric vector

CW20R1 a numeric vector

CW20R2 a numeric vector

CW30R1 a numeric vector

CW30R2 a numeric vector

CW40R1 a numeric vector

CW40R2 a numeric vector

CW50R1 a numeric vector

CW50R2 a numeric vector

CW60R1 a numeric vector

CW60R2 a numeric vector

CW70R1 a numeric vector

CW70R2 a numeric vector

QC00R1 a numeric vector

QC00R2 a numeric vector

QC10R1 a numeric vector

QC20R1 a numeric vector

QC20R2 a numeric vector

QC30R1 a numeric vector

QC30R2 a numeric vector

QC40R1 a numeric vector

QC40R2 a numeric vector

QC50R1 a numeric vector

QC50R2 a numeric vector

QC60R1 a numeric vector

QC60R2 a numeric vector

QU00R1 a numeric vector

QU00R2 a numeric vector

QU10R1 a numeric vector

QU10R2 a numeric vector

QU20R1 a numeric vector

QU20R2 a numeric vector

QU30R1 a numeric vector

QU30R2 a numeric vector

QU40R1 a numeric vector

QU40R2 a numeric vector

QU50R1 a numeric vector

QU50R2 a numeric vector
QU60R1 a numeric vector
QU60R2 a numeric vector
ST00R1 a numeric vector
ST00R2 a numeric vector
ST10R1 a numeric vector
ST10R2 a numeric vector
ST20R1 a numeric vector
ST20R2 a numeric vector
ST30R1 a numeric vector
ST30R2 a numeric vector
ST40R1 a numeric vector
ST40R2 a numeric vector
ST50R1 a numeric vector
ST50R2 a numeric vector
ST60R1 a numeric vector
ST60R2 a numeric vector
ZU00R1 a numeric vector
ZU00R2 a numeric vector
ZU10R1 a numeric vector
ZU10R2 a numeric vector
ZU20R1 a numeric vector
ZU20R2 a numeric vector
ZU30R1 a numeric vector
ZU30R2 a numeric vector
ZU40R1 a numeric vector
ZU40R2 a numeric vector
ZU50R1 a numeric vector
ZU50R2 a numeric vector
ZU60R1 a numeric vector
ZU60R2 a numeric vector
AS00R1 a numeric vector
AS00R2 a numeric vector
AS10R1 a numeric vector
AS10R2 a numeric vector
AS20R1 a numeric vector
AS20R2 a numeric vector
AS30R1 a numeric vector
AS30R2 a numeric vector
AS40R1 a numeric vector

AS40R2 a numeric vector

AS50R1 a numeric vector

AS50R2 a numeric vector

AS60R1 a numeric vector

AS60R2 a numeric vector

AS70R1 a numeric vector

AS70R2 a numeric vector

AS80R1 a numeric vector

AS80R2 a numeric vector

JE00R1 a numeric vector

JE00R2 a numeric vector

JE10R1 a numeric vector

JE10R2 a numeric vector

JE20R1 a numeric vector

JE20R2 a numeric vector

JE30R1 a numeric vector

JE30R2 a numeric vector

JE40R1 a numeric vector

JE40R2 a numeric vector

JE50R1 a numeric vector

JE50R2 a numeric vector

JE60R1 a numeric vector

JE60R2 a numeric vector

JE70R1 a numeric vector

JE70R2 a numeric vector

PLQUR1 a numeric vector

PLQUR2 a numeric vector

PLSTR1 a numeric vector

PLSTR2 a numeric vector

SR00R1 a numeric vector

SR00R2 a numeric vector

SR10R1 a numeric vector

SR10R2 a numeric vector

SR20R1 a numeric vector

SR20R2 a numeric vector

SR30R1 a numeric vector

SR30R2 a numeric vector

SR40R1 a numeric vector

SR40R2 a numeric vector

SR50R1 a numeric vector

SR50R2 a numeric vector

SR60R1 a numeric vector

SR60R2 a numeric vector

SY00R1 a numeric vector

SY00R2 a numeric vector

SY10R1 a numeric vector

SY10R2 a numeric vector

SY20R1 a numeric vector

SY20R2 a numeric vector

SY30R1 a numeric vector

SY30R2 a numeric vector

SY40R1 a numeric vector

SY40R2 a numeric vector

SY50R1 a numeric vector

SY50R2 a numeric vector

SY60R1 a numeric vector

SY60R2 a numeric vector

Details

This set of raw reads can be used to perform gene expression analyses.

Source

Cinvestav Irapuato, Mexico. All libraries were deposited in GEO accession GSE165448 (see https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE165448)

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
data(readcounts)
# Summary of the number of reads in the first
# four RNA-Seq libraries
summary(readcounts[,2:5])
# See library.desc for descriptions of the libraries.
```

SEP *Standardized Expression Profile (SEP)*

Description

For each gene expressed in each one of the 12 accessions this data frame gives the Standardized Expression Profiles (SEP) as well as auxiliary variables.

Usage

```
data("SEP")
```

Format

A data frame with 313919 observations on the following 13 variables.

`id` a numeric vector; gene identifier.

`acc.key` a character vector; accession identifier.

`acc.type` a character vector; accession type identifier.

`model` a character vector; each value consist of six concatenated letters which for each one of the 6 consecutive intervals are D if the expression decremented, S if the expression was steady, I if the expression incremented.

`ExistInAll` a logic vector; TRUE if the gene (id) was expressed in all 12 accessions.

`seT0` a numeric vector; mean expression at time 0.

`seT10` a numeric vector; mean expression at time 10.

`seT20` a numeric vector; mean expression at time 20.

`seT30` a numeric vector; mean expression at time 30.

`seT40` a numeric vector; mean expression at time 40.

`seT50` a numeric vector; mean expression at time 50.

`seT60` a numeric vector; mean expression at time 60.

`TimeMaxExp` a numeric vector; time at which the maximum expression over time was reached.

Details

SEPs are standardized over time, thus the mean value of each vector is equal to zero while its standard deviation (S) is equal to one.

Source

Cinvestav Irapuato, Mexico.

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
# A random row of the data.frame
SEP[sample(c(1:nrow(SEP)), size=1), ]

# Structure of the dataset
str(SEP)

# Number of SEPs that exist in only some or all accessions.
table(SEP$ExistInAll)

# Summary of numeric variables in SEP
summary(SEP[,6:13])
```

SEP.id *Summaries of SEP for each gene expressed*

Description

For each gene expressed this data.frame presents the number of SEPs per group of accessions where the gene was expressed, general mean SEPs as well as mean SEPs per group of accessions (D = Domesticated, W = Wild and C = Crosses). Includes mean correlation coefficients for SEPs in the whole set of SEPs for each gene, and in the SEPs of each group. Finally, it includes the mean and standard deviation (S) for the time at which the maximum expression of the gene was reached for the total, D, W and C groups. Function gene.summary gives a summary of the data in this data.frame per gene id.

Usage

```
data("SEP.id")
```

Format

A data frame with 29946 observations on the following 45 variables.

`id` a numeric vector of gene identifiers.

`n.t` a numeric vector with the total number of accessions where the gene was expressed.

`n.D` a numeric vector with the number of D (domesticated) accessions where the gene was expressed.

`n.W` a numeric vector with the number of W (wild) accessions where the gene was expressed.

`n.C` a numeric vector with the number of C (crosses) where the gene was expressed.

`t.mT0` a numeric vector; mean at time 0 in the whole set of SEPs.

`t.mT10` a numeric vector; mean at time 10 in the whole set of SEPs.

`t.mT20` a numeric vector; mean at time 20 in the whole set of SEPs.

`t.mT30` a numeric vector; mean at time 30 in the whole set of SEPs.

`t.mT40` a numeric vector; mean at time 40 in the whole set of SEPs.

`t.mT50` a numeric vector; mean at time 50 in the whole set of SEPs.

`t.mT60` a numeric vector; mean at time 60 in the whole set of SEPs.

`D.mT0` a numeric vector; mean at time 0 in the D set of SEPs.

D.mT10 a numeric vector; mean at time 10 in the D set of SEPs.

D.mT20 a numeric vector; mean at time 20 in the D set of SEPs.

D.mT30 a numeric vector; mean at time 30 in the D set of SEPs.

D.mT40 a numeric vector; mean at time 40 in the D set of SEPs.

D.mT50 a numeric vector; mean at time 50 in the D set of SEPs.

D.mT60 a numeric vector; mean at time 60 in the D set of SEPs.

W.mT0 a numeric vector; mean at time 0 in the W set of SEPs.

W.mT10 a numeric vector; mean at time 10 in the W set of SEPs.

W.mT20 a numeric vector; mean at time 20 in the W set of SEPs.

W.mT30 a numeric vector; mean at time 30 in the W set of SEPs.

W.mT40 a numeric vector; mean at time 40 in the W set of SEPs.

W.mT50 a numeric vector; mean at time 50 in the W set of SEPs.

W.mT60 a numeric vector; mean at time 60 in the W set of SEPs.

C.mT0 a numeric vector; mean at time 0 in the C set of SEPs.

C.mT10 a numeric vector; mean at time 10 in the C set of SEPs.

C.mT20 a numeric vector; mean at time 20 in the C set of SEPs.

C.mT30 a numeric vector; mean at time 30 in the C set of SEPs.

C.mT40 a numeric vector; mean at time 40 in the C set of SEPs.

C.mT50 a numeric vector; mean at time 50 in the C set of SEPs.

C.mT60 a numeric vector; mean at time 60 in the C set of SEPs.

m.r.t a numeric vector; mean correlation between SEPs.

m.r.D a numeric vector; mean correlation between SEPs in D.

m.r.W a numeric vector; mean correlation between SEPs in W.

m.r.C a numeric vector; mean correlation between SEPs in C.

m.TME.t a numeric vector; mean of maximum expression time in the whole set.

S.TME.t a numeric vector; Stantard deviation of maximum expression time in the whole set.

m.TME.D a numeric vector; mean of maximum expression time in the D set.

S.TME.D a numeric vector; Stantard deviation of maximum expression time in the D set.

m.TME.W a numeric vector; mean of maximum expression time in the D set.

S.TME.W a numeric vector; Stantard deviation of maximum expression time in the W set.

m.TME.C a numeric vector; mean of maximum expression time in the C set.

S.TME.C a numeric vector; Stantard deviation of maximum expression time in the C set.

Details

NA values are present where the corresponding value could not be estimated.

Source

Cinvestav Irapuato, Mexico

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

Examples

```
# A random row of the data.frame
SEP.id[sample(c(1:nrow(SEP.id)), size=1), ]

# Table with the number of SEPs in the whole (total) set.
table(SEP.id$n.t)

# Plot of mean SEPs per group.
plot(DAA, apply(SEP.id[SEP.id$n.t==12, 6:12], 2, mean), ylim=c(-0.43, 0.25), type="b", ylab="Mean Standaridz
points(DAA, apply(SEP.id[SEP.id$n.D==6, 13:19], 2, mean), type="b", col="red")
points(DAA, apply(SEP.id[SEP.id$n.W==4, 20:26], 2, mean), type="b", col="blue")
points(DAA, apply(SEP.id[SEP.id$n.C==2, 27:33], 2, mean), type="b", col="violet")
legend("topright", legend=c("Total", "D", "W", "C"), pch=1, lty=1, col=c("black", "red",  "blue", "violet"))
```

`SEP.summary` *Summary of one or many SEPs*

Description

Gives a comprehensive summary for the content of one or more Standardized Expression Profile (SEPs) `data.frames`. The output is a list with four components containing general statistics, means per expression time, lower limit of the confidence interval for the means and upper limit of the confidence interval for the means, respectively. For each one of the SEPs in the input, one row is produced in each one of the rows of the output `data.frame`.

Usage

```
SEP.summary(..., conf.level = 0.95)
```

Arguments

`...`	One or more SEPs.
`conf.level`	Confidence Level for the calculation of Confidence Intervals for the mean expression.

Details

The order of the rows in the results is the order at which each SEP was set in the input. The `t.test` function is used to obtain the Confidence Intervals for the mean expression at each one of the 7 times sampled. If the expression at one or more time points are uniform, the Confidence Interval limits for the mean expression at those points are set equal to the mean at those points, avoiding errors. For SEPs with only one row, CIs (`LL.time.means` and `UL.time.means`) are set equal to the value of the corresponding mean and a warning is shown in the output.

Value

A list with four components: general time.means LL.time.means and UL.time.means. For each SEP in the input, one row is produced in each one of the four components. The order of the rows in the four components of the output corresponds to the order of the SEPs in the input.

general	data.frame with columns n.rows Number of rows in the SEP, n.ids Number of different gene identifiers in the SEP, n.acc Number of different accession keys in the SEP, n.type Number of different accession types in the SEP, n.mod Number of different models in the SEP, mean.TimeMaxExp Mean time of maximum expression for SEPs, LL.TimeMaxExp Lower Limit for the Mean time of maximum expression for SEPs, UL.TimeMaxExp Upper Limit for the Mean time of maximum expression for SEPs.
time.means	data.frame with seven columns corresponding to the mean expression at each one of the times sampled, from m.T0 up to m.T60.
LL.time.means	data.frame with seven columns corresponding to the lower limits for the mean expression at each one of the times sampled, from LL.m.T0 up to LL.m.T60.
UL.time.means	data.frame with seven columns corresponding to the upper limits for the mean expression at each one of the times sampled, from UL.m.T0 up to UL.m.T60.

Each one of the rows in each one of the components corresponds to one of the SEPs in the input, in the same order; i.e., the first row corresponds to to first SEP in the input, the second row corresponds to the second SEP in the input, etc.

Note

The output gives a summary of the content of each SEP and is used to plot the mean expression of SEPs with their confidence levels at each time, allowing the judgment of significance of differences per time.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

SEP, get.SEP, SEP.id, SEPs.plot, analyze.2.SEPs.

Examples

```
# Summary of a SEP with a single gene (id)
SEP.summary(get.SEP(ids=3))

# Summary of a SEP with many ids
# (all genes that are Transcription Factors)
SEP.summary(get.SEP(isTF=TRUE))
```

```
# Summary of three different SEPs
# (each one containing all Transcription Factors
# in each one of the three acc.type)
SEP.summary(get.SEP(acc.type="D", isTF=TRUE),
get.SEP(acc.type="W", isTF=TRUE),
get.SEP(acc.type="C", isTF=TRUE))

# Summary in a case where all expression values
# at each time are the same (and thus confidence
# intervals are set to be equal to the means)
SEP.summary(get.SEP(model="DSSSSS"))
```

`SEPs.plot` *Plot mean expression times in SEPs*

Description

For one or more SEPs produces a plot of the means of standardized expression at each one of the times sampled, including their confidence intervals.

Usage

```
SEPs.plot(seps, colors, lwd = 2, conf.level = 0.95, CIs.sep = 0.5)
```

Arguments

`seps`	Either, a single SEP or a `list` of two or more SEPs.
`colors`	A single color in the case when `seps` is a single SEP, or a vector of colors with the same number of colors than the number of `seps` passed on that `list`. Any valid code for `col` (color) is accepted.
`lwd`	Line width.
`conf.level`	Confidence Level for the estimation of the Confidence Intervals for the means at each one of the times.
`CIs.sep`	A value to avoid the overlapping of Confidence Intervals in the X axis. The default of 0.5 days works well in most cases.

Details

It is very important that if two or more SEPs are pass to this function, the groups of SEPs must be defined as a `list`. Otherwise an unexpected error will be produced. If the expression at one or more time points are uniform, the Confidence Interval limits for the mean expression at those points are set equal to the mean at those points, avoiding errors. In cases of SEPs with only one row CIs are not shown (it is not posible to estimate such CIs).

Value

`NULL` (the function is used by its side effect, i.e., to produce a plot of SEPs).

Note

Confidence Intervals for the means are shown as thin vertical lines. It is not wise to produce plots with more than a few SEPs, say a maximum of five. Plots with more SEPs will be difficult to interpret.

Author(s)

O. Martinez

References

Escoto-Sandoval C, Flores-Diaz A, Reyes-Valdes MH, Ochoa-Alejo N and Martinez O. "Salsa: An R database to analyze gene expression profiles in chili pepper fruit". (Submitted, 2021).

Martinez O, et al. "Transcriptome Analyses Throughout Chili Pepper Fruit Development Reveal Novel Insights into the Domestication Process". Plants. 2021; 10(3):585. doi.org/10.3390/plants10030585

See Also

SEP.summary, get.SEP, SEP.id, SEPs.plot, analyze.2.SEPs.

Examples

```
# Plot of a single SEP
temp.sep.3 <- get.SEP(ids=3)
SEPs.plot(seps=temp.sep.3, colors="grey")

# Obtaining SEPs per acc.type
temp.sep.3.D <- get.SEP(ids=3, acc.type="D")
temp.sep.3.W <- get.SEP(ids=3, acc.type="W")
temp.sep.3.C <- get.SEP(ids=3, acc.type="C")

# and plotting them together
# Note: in that case argument seps
# MUST be a list!
SEPs.plot(seps=list(temp.sep.3, temp.sep.3.D,
temp.sep.3.W, temp.sep.3.C),
colors=c("grey", "red", "blue", "violet"))

# Clean temporary objects
rm(list=ls(patt="temp.sep.3"))

# Plot SEPs for two different genes
SEPs.plot(seps=list(get.SEP(ids=3), get.SEP(ids=19)), colors=c("red", "blue"))

# Plot of SEPs with all transcription
# factors group by acc.type
SEPs.plot(seps=list(get.SEP(acc.type="D", isTF=TRUE),
get.SEP(acc.type="W", isTF=TRUE),
get.SEP(acc.type="C", isTF=TRUE)),
colors=c("red", "blue", "violet"))
```

Index

∗Topic **datasets**
acc, 4
all.GO, 5
DAA, 15
FPKM.expr, 17
gene, 25
GO.annot, 36
library.desc, 37
readcounts, 39
SEP, 45
SEP.id, 46
∗Topic **package**
Salsa-package, 2

acc, 4
all.GO, 5, *29*, *31*
analyze.2.SEPs, 6, *35*, *49*, *51*
analyze.all.GO, 7, *11*
analyze.g2g, 9, *19*
analyze.GO, *8*, 10, *16*, *29*, *31*, *34*

browse.gene, *12*, 12, *32*
browse.GO, *8*, *11*, 13, *29*, *31*

core.g2g, *9*, 14, *19*, *22*

DAA, 15

expected, 16, *34*

FPKM.expr, 17, *35*

g2g, *9*, *15*, 18, *22*, *24*, *39*
g2g.plot.winers, *19*, 21
g2g.TFcandidates, *19*, 23, *39*
gene, *12*, 25, *28*, *32*
gene.summary, 26
get.genes.by.desc, *12*, 27, *32*
get.genes.by.GO.desc, 28, *31*
get.GO.terms.by.desc, *8*, *11*, *13*, *29*, 30
get.ids, *8*, *11*, 31
get.mat.from.ids, *16*, 33
get.SEP, *7*, 34, *49*, *51*
GO.annot, *29*, *31*, 36

library.desc, 37

pred.error.g2g, *19*, 38

readcounts, 39

Salsa *(*Salsa-package*)*, 2
Salsa-package, 2
SEP, *35*, 45, *49*
SEP.id, *27*, 46, *49*, *51*
SEP.summary, *7*, *35*, 48, *51*
SEPs.plot, *35*, *49*, 50, *51*