Package 'ELBOW'

October 9, 2015

Type Package

Title ELBOW - Evaluating foLd change By the lOgit Way

Version 1.4.0

Date 2014-01-30

Author Xiangli Zhang, Natalie Bjorklund, Graham Alvare, Tom Ryzdak, Richard Sparling, Brian Fristensky

Maintainer Graham Alvare <alvare@cc.umanitoba.ca>, Xiangli Zhang <zhangju@cc.umanitoba.ca>

Description Elbow an improved fold change test that uses cluster analysis and pattern recognition to set cut off limits that are derived directly from intrareplicate variance without assuming a normal distribution for as few as 2 biological replicates. Elbow also provides the same consistency as fold testing in cross platform analysis. Elbow has lower false positive and false negative rates than standard fold testing when both are evaluated using T testing and Statistical Analysis of Microarray using 12 replicates (six replicates each for initial and final conditions). Elbow provides a null value based on initial condition replicates and gives error bounds for results to allow better evaluation of significance.

License file LICENSE

License_is_FOSS yes

License_restricts_use no

Suggests DESeq, GEOquery, limma, simpleaffy, affyPLM, RColorBrewer

Depends R (>= 2.15.0)

biocViews Technology, Microarray, RNASeq, Sequencing, Sequencing, Software, MultiChannel, OneChannel, TwoChannel, GeneExpression

NeedsCompilation no

R topics documented:

analyze_elbow . . . 2
do_elbow . . . 3
do_elbow_rnaseq . . . 5
EcoliMutMA . . . 6
EcoliWT . . . 8
ELBOW . . . 10
elbow_variance . . . 12
extract_working_sets . . . 14
get_elbow_limma . . . 15
get_pvalue . . . 16
GSE20986_design . . . 18
GSE20986_eset_exprs . . . 20
null_variance . . . 21
plot_dataset . . . 23
plot_elbow . . . 24
replicates_to_fold . . . 26
yeast_nbinomTest_results . . . 28

Index **29**

`analyze_elbow` *extracts all elbow statistics and plots and elbow curve.*

Description

Analyzes:

- the Elbow cut-offs
- the Elbow curve variance
- the upper and lower error Elbow curves
- $\log \chi^2$ p-value for the Elbow curve model

Usage

```
analyze_elbow(probes, initial_conditions,
  final_conditions, gtitle = "")
```

Arguments

`probes`	the data set of probes (as a data.frame).
`initial_conditions`	a 2D data.fram containing all of the replicate gene expression values corresponding to the experiment's initial conditions.
`final_conditions`	a 2D data.fram containing all of the replicate gene expression values corresponding to the experiment's final conditions.
`gtitle`	the title to display for the graph.

Details

Then plots the data to a curve - AND - prints the statistics to both the terminal and the plot canvas.

Value

a list of all significant probes

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

# Analyze the elbow curve.
sig <- analyze_elbow(probes, initial_conditions, final_conditions)

# write the significant probes to 'signprobes.csv'
write.table(sig,file="signprobes.csv",sep=",",row.names=FALSE)
```

---

do_elbow *calculates the upper and lower elbow cut-off fold-values*

Description

calculates the upper and lower elbow cut-off fold-values

Usage

```
do_elbow(fold_values)
```

Arguments

fold_values	a data.frame containing all of the fold values to calculate the elbow for.

Value

a list containing the following:

- up_limit — the upper (most positive) limit of the elbow curve
- low_limit — the lower (most negative) limit of the elbow curve

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

my_data <- replicates_to_fold(probes, initial_conditions, final_conditions)

# compute the elbow for the dataset
limits <- do_elbow(data.frame(my_data$fold))
```

do_elbow_rnaseq *Calculates fold values from an MArrayLM object.*

Description

Performs the ELBOW fold change test on an CountDataSet from DESeq object. This is a wrapper class to help integrate the ELBOW method into Bioconductor. followed tutorial from: http://cgrlucb.wikispaces.com/Spring+

Usage

```
do_elbow_rnaseq(rnaSeq)
```

Arguments

rnaSeq	is the CountDataSet object to analyze.

Value

a matrix specified as follows

- columns — (1) "up_limit", the upper ELBOW fold-change cut-off value; (2) "low_limit", the lower ELBOW fold-change cut-off value
- rows — one row per sample, specified by the parameter "columns."

Examples

```
# install the DESeq libraries
#source("http://www.bioconductor.org/biocLite.R")
#biocLite("DESeq")

## download the table
library("DESeq")

# the following bam file dataset was obtained from:
# http://cgrlucb.wikispaces.com/file/view/yeast_sample_data.txt
# it has been downloaded into this package for speed convenience.
filename <- system.file("extdata", "yeast_sample_data.txt", package = "ELBOW")

count_table <- read.table(filename, header=TRUE, sep="\t", row.names=1)
expt_design <- data.frame(row.names = colnames(count_table), condition = c("WE","WE","M","M","M"))
conditions = expt_design$condition
data <- newCountDataSet(count_table, conditions)
data <- estimateSizeFactors(data)
data <- as(data, "CountDataSet")
## data <- estimateVarianceFunctions(data)
data <- estimateDispersions(data)

# this next step is essential, but it takes a long time...
# so, just like a good cooking show we will skip this step
```

```
# and load a finished version.
#results <- nbinomTest(data, "M", "WE")

# The below two code lines load a copy of the above dataset
# which has already been processed by:
#     results <- nbinomTest(data, "M", "WE")
# For your own real data, you must use:
#     results <- nbinomTest(data, "M", "WE")'
# Instead of the two lines below:
data(yeast_nbinomTest_results, package="ELBOW")
results <- yeast_nbinomTest_results

# obtain the elbow limit for the dataset
# the final step in the analysis pipeline
do_elbow_rnaseq(results)
```

`EcoliMutMA` *Escherichia coli str. K-12 substr. MG1655 - EcoliMutMA (GSM576640-45 mutant – G3.2 mutant of E.coli EcNR1, no isobutanol)*

Description

Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli

Platform organism: Escherichia coli str. K-12 substr. MG1655
Sample organism: Escherichia coli
Experiment type: Expression profiling by array
Summary

BACKGROUND: Isobutanol is a promising next generation biofuel with demonstrated high yield microbial production, but the toxicity of this molecule reduces fermentation volumetric productivity and final titers. Organic solvent tolerance is a complex, multigenic phenotype that has been recalcitrant to rational engineering approaches. We apply experimental evolution followed by genome resequencing and a gene expression study to elucidate genetic bases on adaptation to exogenous isobutanol stress.

RESULTS: The adaptations acquired in our evolved lineages exhibit antagonistic pleiotropy between minimal and rich medium, and appear to be specific to the effects of longer chain alcohols. By examining genotypic adaptation in multiple independent lineages, we find evidence of parallel evolution in hfq, mdh, acrAB, gatYZABCD, and rph genes. Many isobutanol tolerant lineages show reduced rpoS activity, perhaps related to mutations in hfq or acrAB. Consistent with the complex, multigenic nature of solvent tolerance, we observe adaptations in a diversity of cellular processes. Many adaptations appear to involve epistasis between different mutations, implying a rugged fitness landscape for isobutanol tolerance. We observe a trend of evolution targeting post-transcriptional

regulation and high centrality nodes of biochemical networks. Collectively, the genotypic adaptations we observe suggest mechanisms of adaptation to isobutanol stress based on remodelling the cell envelope and surprisingly, stress response attenuation.

CONCLUSIONS: We have discovered a set of genotypic adaptations that confer increased tolerance to exogenous isobutanol stress. Our results are immediately useful to efforts to engineer more isobutanol tolerant host strains of E. coli for isobutanol production. We suggest that rpoS and post-transcriptional regulators, such as hfq, RNA helicases, and sRNAs may be interesting mutagenesis targets for futurue global phenotype engineering.

Overall design
Two strains (WT strain and G3.2 mutant strain), each with two culture conditions (with and without isobutanol in medium). Three biological replicates for each strain/culture condition. Twelve samples in total.

- GSM576634 – WT E.coli EcNR1, no isobutanol
- GSM576635 – WT E.coli EcNR1, no isobutanol
- GSM576636 – WT E.coli EcNR1, no isobutanol
- GSM576637 – WT E.coli EcNR1, 0.5 % isobutanol
- GSM576638 – WT E.coli EcNR1, 0.5 % isobutanol
- GSM576639 – WT E.coli EcNR1, 0.5 % isobutanol
- GSM576640 – G3.2 mutant of E.coli EcNR1, no isobutanol – EcoliMutMA
- GSM576641 – G3.2 mutant of E.coli EcNR1, no isobutanol
- GSM576642 – G3.2 mutant of E.coli EcNR1, no isobutanol
- GSM576643 – G3.2 mutant of E.coli EcNR1, 0.5 % isobutanol
- GSM576644 – G3.2 mutant of E.coli EcNR1, 0.5 % isobutanol
- GSM576645 – G3.2 mutant of E.coli EcNR1, 0.5 % isobutanol

Correponding dataset
EcoliMutMA – GSM576640-45 mutant

Usage

```
EcoliMutMA
```

Format

A 7 column table defined as such:

- the first column containing the names of the probes
- columns 2-4 contain the gene expression values of three replicates for the initial conditions of the experiment
- columns 5-7 contain the gene expression values of three replicates for the final conditions of the experiment

Source

Geo Accession #Series GSE23526

References

Minty, J. J., Lesnefsky, A. A., Lin, F., Chen, Y., Zaroff, T. A., Veloso, A. B., Xie, B., McConnell, C. A.,Ward, R. J., Schwartz, D. R., Rouillard, J. M., Gao, Y., Gulari, E., Lin, X.N. (March 2011) "Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli." *Microb Cell Fact.* Vol. **10**, p. 18. doi: 10.1186/1475-2859-10-18.

`EcoliWT` *Escherichia coli str. K-12 substr. MG1655 - EcoliWT (GSM576634-39 WT – WT E.coli EcNR1, no isobutanol)*

Description

Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli

Platform organism: Escherichia coli str. K-12 substr. MG1655
Sample organism: Escherichia coli
Experiment type: Expression profiling by array
Summary

BACKGROUND: Isobutanol is a promising next generation biofuel with demonstrated high yield microbial production, but the toxicity of this molecule reduces fermentation volumetric productivity and final titers. Organic solvent tolerance is a complex, multigenic phenotype that has been recalcitrant to rational engineering approaches. We apply experimental evolution followed by genome resequencing and a gene expression study to elucidate genetic bases on adaptation to exogenous isobutanol stress.

RESULTS: The adaptations acquired in our evolved lineages exhibit antagonistic pleiotropy between minimal and rich medium, and appear to be specific to the effects of longer chain alcohols. By examining genotypic adaptation in multiple independent lineages, we find evidence of parallel evolution in hfq, mdh, acrAB, gatYZABCD, and rph genes. Many isobutanol tolerant lineages show reduced rpoS activity, perhaps related to mutations in hfq or acrAB. Consistent with the complex, multigenic nature of solvent tolerance, we observe adaptations in a diversity of cellular processes. Many adaptations appear to involve epistasis between different mutations, implying a rugged fitness landscape for isobutanol tolerance. We observe a trend of evolution targeting post-transcriptional regulation and high centrality nodes of biochemical networks. Collectively, the genotypic adaptations we observe suggest mechanisms of adaptation to isobutanol stress based on remodelling the cell envelope and surprisingly, stress response attenuation.

CONCLUSIONS: We have discovered a set of genotypic adaptations that confer increased tolerance to exogenous isobutanol stress. Our results are immediately useful to efforts to engineer more

isobutanol tolerant host strains of E. coli for isobutanol production. We suggest that rpoS and post-transcriptional regulators, such as hfq, RNA helicases, and sRNAs may be interesting mutagenesis targets for futurue global phenotype engineering.

Overall design
Two strains (WT strain and G3.2 mutant strain), each with two culture conditions (with and without isobutanol in medium). Three biological replicates for each strain/culture condition. Twelve samples in total.

- GSM576634 – WT E.coli EcNR1, no isobutanol – EcoliWT
- GSM576635 – WT E.coli EcNR1, no isobutanol
- GSM576636 – WT E.coli EcNR1, no isobutanol
- GSM576637 – WT E.coli EcNR1, 0.5 % isobutanol
- GSM576638 – WT E.coli EcNR1, 0.5 % isobutanol
- GSM576639 – WT E.coli EcNR1, 0.5 % isobutanol
- GSM576640 – G3.2 mutant of E.coli EcNR1, no isobutanol
- GSM576641 – G3.2 mutant of E.coli EcNR1, no isobutanol
- GSM576642 – G3.2 mutant of E.coli EcNR1, no isobutanol
- GSM576643 – G3.2 mutant of E.coli EcNR1, 0.5 % isobutanol
- GSM576644 – G3.2 mutant of E.coli EcNR1, 0.5 % isobutanol
- GSM576645 – G3.2 mutant of E.coli EcNR1, 0.5 % isobutanol

Correponding dataset
EcoliWT – GSM576634-39 WT

Usage

```
EcoliWT
```

Format

A 7 column table defined as such:

- the first column containing the names of the probes
- columns 2-4 contain the gene expression values of three replicates for the initial conditions of the experiment
- columns 5-7 contain the gene expression values of three replicates for the final conditions of the experiment

Source

Geo Accession #Series GSE23526

References

Minty, J. J., Lesnefsky, A. A., Lin, F., Chen, Y., Zaroff, T. A., Veloso, A. B., Xie, B., McConnell, C. A.,Ward, R. J., Schwartz, D. R., Rouillard, J. M., Gao, Y., Gulari, E., Lin, X.N. (March 2011) "Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli." *Microb Cell Fact.* Vol. **10**, p. 18. doi: 10.1186/1475-2859-1 0-18.

ELBOW *Evaluating foLd change By the lOgit Way*

Description

The "elbow" method an improved fold change test for determining cut off for biologically significant changes in expression levels in transcriptomics.

Details

Package:	ELBOW
Type:	Package
Version:	1.0
Date:	2013-08-08
License:	Creative Commons 3.0 Attribution + ShareAlike
	(see: http://creativecommons.org/licenses/by-sa/3.0/)

Elbow an improved fold change test that uses cluster analysis and pattern recognition to set cut off limits that are derived directly from intrareplicate variance without assuming a normal distribution for as few as 2 biological replicates. Elbow also provides the same consistency as fold testing in cross platform analysis. Elbow has lower false positive and false negative rates than standard fold testing when both are evaluated using T testing and Statistical Analysis of Microarray using 12 replicates (six replicates each for initial and final conditions). Elbow provides a null value based on initial condition replicates and gives error bounds for results to allow better evaluation of significance.

Abstract Reference:
Conference Proceeding: Zhang, X., Bjorklund, N. K., Rydzak, T., Sparling, R., Alvare, G., Fristensky, B. (April 2013) "The Elbow Method for deciding significant fold change cutoffs of differentially expressed genes." *Recomb 2013 17th International Conference on Research in Computational Biology*.

Paper Reference:
Zhang, X., Bjorklund, N. K., Alvare, G., Rydzak, T., Sparling, R., Fristensky, B. (2013) "Elbow, an improved fold test method for transcriptomics." Departments of Plant Science and Microbiology, University of Manitoba, Winnipeg, Canada, R3T 2N2

The corresponding author: Brian Fristensky <frist@cc.umanitoba.ca>

Author(s)

Xiangli Zhang, Natalie Bjorklund, Graham Alvare, Tom Ryzdak, Richard Sparling, Brian Fristensky

Maintainers: Graham Alvare <alvare@cc.umanitoba.ca>, Xiangli Zhang <zhangju@cc.umanitoba.ca>

References

Claeskens, G. and Hjort N. L. (2008) *Model Selection and Model Averaging.* Cambridge, England: Cambridge University Press.

Cui X. and Churchill G. A. (2003) "Statistical tests for differential expression in cDNA microarray experiments." *Genome Biol*, Vol. **4**, p. 210.

Dalman, M. R., Deeter, A., Nimishakavi, G. and Duan, Z. H. (2012) "Fold change and p-value cutoffs significantly alter microarray interpretations." *BMC Bioinformatics*, Vol. **13** (Suppl. 2), p. S11

Faraway, J. J. (2006) *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models.* Boca Raton, Florida: Chapman and Hall/CRC.

Guo, L., Lobenhofer, E. K., et al. (2006) "Rat toxicogenomic study reveals analytical consistency across microarray platforms." *Nature Biotechnol*, Vol. **24**, pp. 1162–1169.

Klebanov, L., Qiu, X., Welle, S., Yakovlev, A. (2007) "Statistical methods and microarray data." *Nature Biotechnol*, Vol. **25**, p.1.

MAQC Consortium (2006) "The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements." *Nature Biotechnol*, September, Vol. **24**, No. 9, pp. 1151–1161.

Minty, J. J., Lesnefsky, A. A., et al. (2011) "Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli." *Microb Cell Fact.* Vol. **10**, p. 18.

NIST/SEMATECH e-Handbook of Statistical Methods, April 2012, http://www.itl.nist.gov/div898/handbook/.

Oshlack, A., Robinson, M. D., Young, M. D., (2010) "From RNA-seq reads to differential expression results." *Genome Biol*, Vol. **11**, p. 220.

Shi, L., Tong, W., et al. (2005) "Cross-platform comparability of microarray technology: intra-platform consistency and appropriate data analysis procedures are essential." *BMC Bioinformatics*, Vol. **6** (Suppl. 2), p. S12.

Sjogren, A., Kristiansson, E., Rudemo, M., Nerman, O. (2007) "Weighted analysis of general microarray experiments." *BMC Bioinformatics*, Vol. **8**, p. 387.

Tan, P. K., Downey, T. J., et al. (2003) "Evaluation of gene expression measurements from commercial microarray platforms." *Nucleic Acids Res*, Vol. **31**, No.19, pp. 5676–5684.

Thorndike, R. L. (December 1953) "Who Belong in the Family?" Psychometrika, Vol. **18**, No. 4, pp. 267–276.

Tilstone, C. (2003) "Vital statistics." *Nature*, Vol. **424**, p. 611

Tusher, V. G., Tibshirani, R., Chu, G. (2001) "Significance analysis of microarrays applied to the ionizing radiation response." *Proc Natl Acad Sci*, Aug. 28, Vol. **98** No. 9: pp. 5116–5121.

See Also

See analyze_elbow for doing a full ELBOW analysis and plot.
See do_elbow if you want to extract only the ELBOW cut-off values.

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

# Analyze the elbow curve.
sig <- analyze_elbow(probes, initial_conditions, final_conditions)

# write the significant probes to 'signprobes.csv'
write.table(sig,file="signprobes.csv",sep=",",row.names=FALSE)
```

`elbow_variance` *calculates the variance for the upper and lower limits of an Elbow curve*

Description

The following function determines the variation of cut off limits for significance between individual subsets. The first step is to generate the subsets, then the function determines the highest and lowest

values for the upper limit and the highest and lowest values for the lower limit. These values are used to determine and upper and lower error value reflecting fold dataset variance.

Usage

```
elbow_variance(probes, initial_conditions,
  final_conditions)
```

Arguments

probes	the data set of probes.
initial_conditions	a data set of replicates corresponding to initial conditions.
final_conditions	a data set of replicates corresponding to final conditions.

Value

a list containing the following keys:

- max_upper — the maximum upper elbow limit *(most positive)*
- min_upper — the minimum upper elbow limit
- max_upper — the maximum lower elbow limit
- min_upper — the minimum lower elbow limit *(most negative)*

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
```

```
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

my_data <- replicates_to_fold(probes, initial_conditions, final_conditions)

# compute the elbow for the dataset
limits <- do_elbow(data.frame(my_data$fold))

plus_minus <- elbow_variance(probes, initial_conditions, final_conditions)
```

`extract_working_sets` *extract the initial and final conditions from a table.*

Description

Extract the initial and final conditions from a table.

Usage

```
extract_working_sets(first_my_data, replicate_count1,
  replicate_count2)
```

Arguments

`first_my_data` A table to translate. The columns in the table should be as follows:

- probes — one column containing the names of the probes
- initial conditions — one or more columns containing the initial conditions of the experiment
- final conditions — one or more columns containing the final conditions of the experiment

`replicate_count1`
the number of columns (replicates) corresponding to initial conditions.

`replicate_count2`
the number of columns (replicates) corresponding to final conditions.

Value

a list containing the following:

- probes — the set of probes.
- initial — a data set of replicates corresponding to initial conditions.
- final — a data set of replicates corresponding to final conditions.

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)
```

`get_elbow_limma` *Calculates fold values from an MArrayLM object.*

Description

Performs the ELBOW fold change test on an MArrayLM object. This is a wrapper class to help integrate the ELBOW method into Bioconductor. followed tutorial from: http://bioinformatics.knowledgeblog.org/2011/06/20/an microarray-data-in-bioconductor/

Usage

```
get_elbow_limma(marraylm, columns = NULL)
```

Arguments

`marraylm`	is the MArrayLM object to analyze.
`columns`	is the list of sample columns to obtain the elbow fold cut-off values for. This can be specified as a vector or a single value.

Details

Some of the code in this method is based on how toptable accesses the MArrayLM object to read fold values. Therefore, any MArrayLM object which works with the toptable method should also work with this method.

Value

a matrix specified as follows

- columns — (1) “up_limit”, the upper ELBOW fold-change cut-off value; (2) “low_limit”, the lower ELBOW fold-change cut-off value
- rows — one row per sample, specified by the parameter “columns.”

Examples

```
########
# LOAD DATA INTO LIMMA
########
library("limma")

# load a filtered expression set into R
# NOTE: see the vignette for instructions on preparing
#       a filtered dataset with your own data.
data(GSE20986_eset_exprs, package="ELBOW")
data(GSE20986_design, package="ELBOW")

# fit the linear model to the filtered expression set
fit <- lmFit(GSE20986_eset_exprs, GSE20986_design)

# set up a contrast matrix to compare tissues v cell line
contrast.matrix <- makeContrasts(huvec_choroid = huvec - choroid, huvec_retina = huvec - retina, huvec_iris <- hu

# Now the contrast matrix is combined with the per-probeset linear model fit.
huvec_fits <- contrasts.fit(fit, contrast.matrix)
huvec_ebFit <- eBayes(huvec_fits)

########
# GET THE ELBOW LIMIT (this function)
########
get_elbow_limma(huvec_ebFit)
```

`get_pvalue` *The calculated* $\log \chi$ˆ2 *p-value for the fit of the elbow curve to the model.*

Description

The following function determines a p value from the comparison of the slope of a regression line (i.e. logit). The function calculates a fold value from a set of initial conditions (replicate set 1) substracted from a set of final conditions (replicate set 2). The glm function is used to assess the fit of the dataset to the model, assuming family = binomial (link = "probit"), maxit = 1000. Determining the $\log \chi^2$ p-value for the model, null equals no significance for the set of significant probes to the model, in which case the Elbow method cannot be used to assess the dataset. If significant probes are important to the model, then the $\log \chi^2$ p-value will be below 0.05, in which case the Elbow method can be used to assess the dataset.

Usage

```
get_pvalue(my_data, upper_limit, lower_limit)
```

Arguments

my_data	A table (data.frame) to analyze the elbow variance of. The columns in the table should be as follows: • probes — one column containing the names of the probes • fold — the fold values for the table
upper_limit	the upper limit/cut-off for the elbow.
lower_limit	the lower limit/cut-off for the elbow.

Value

The calculated $\log \chi^2$ p-value for the elbow curve.

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

my_data <- replicates_to_fold(probes, initial_conditions, final_conditions)

# compute the elbow for the dataset
```

```
limits <- do_elbow(data.frame(my_data$fold))

plus_minus <- elbow_variance(probes, initial_conditions, final_conditions)

max_upper_variance <- plus_minus$max_upper
min_upper_variance <- plus_minus$min_upper
max_lower_variance <- plus_minus$max_lower
min_lower_variance <- plus_minus$min_lower

# rounded number for nice appearance graph
upper_limit <- round(limits[[1]],digits=2)

# rounded number nice appearance for graph
lower_limit <- round(limits[[2]],digits=2)

p_limits <- null_variance(my_data, upper_limit, lower_limit, initial_conditions)

prowmin <- p_limits[[1]]
prowmax <- p_limits[[2]]
prowmedian <- p_limits[[3]]

pvalue1 <- get_pvalue(my_data, upper_limit, lower_limit)
```

`GSE20986_design` *Comparative Gene Expression Profiling of HUVEC and Ocular Vascular Endothelial Cells (GEO database entry GSE20986) microarray design data file*

Description

Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli

Platform: GPL570 [HG-U133_Plus_2] Affymetrix Human Genome U133 Plus 2.0 Array
Organism: Homo sapiens
Experiment type: Expression profiling by array
Summary

To compare the gene expression profiles of unpassaged, proliferating HUVEC and human iris, retinal and choroidal microvascular endothelial cells. Gene expression profiling revealed significant differences between HUVEC and ocular microvascular endothelial cells suggesting that HUVE cells may not be a suitable surrogate when studying pathophysiological mechanisms of ocular disorders. There were significant differences in the gene expression of important cell signalling pathways in human retinal and choroidal ECs. These differences may be important in the mechanisms and treatment of choroidal and retinal neovascularisation.

Overall design
12 arrays are included. Endothelial cells were derived from 4 tissues: iris, retina, choroid and human umbilical vein. RNA extracts from cells were hybridised to Affymetrix HGU133plus2 arrays in triplicate.

Samples

- GSM524662 — iris-cellline-1
- GSM524663 — retina-cellline-1
- GSM524664 — retina-cellline-2
- GSM524665 — iris-cellline-2
- GSM524666 — retina-cellline-3
- GSM524667 — iris-cellline-3
- GSM524668 — choroid-cellline-1
- GSM524669 — choroid-cellline-2
- GSM524670 — choroid-cellline-3
- GSM524671 — huvec-cellline-1
- GSM524672 — huvec-cellline-2
- GSM524673 — huvec-cellline-3

Usage

```
GSE20986_design
```

Format

A 4 column table relating each sample (see above list) to parts of the body. The columns (parts of the body) are the following:

- choroid
- huvec
- iris
- retina

Source

Geo Accession #Series GSE20986

References

Browning, A. C., Halligan, E. P., Swan, D. C., et al. (Jan 2012) "Comparative Gene Expression Profiling of HUVEC and Ocular Vascular Endothelial Cells." *Br J Ophthalmol.* Vol. **96**, No. 1, p. 128–132. doi: 10.1136/bjophthalmol-2011-300572.

GSE20986_eset_exprs *Comparative Gene Expression Profiling of HUVEC and Ocular Vascular Endothelial Cells (GEO database entry GSE20986) microarray data eset expression values - SUBSET*

Description

The data in this set is a subset of the following experiment:

Evolution combined with genomic study elucidates genetic bases of isobutanol tolerance in Escherichia coli

Platform: GPL570 [HG-U133_Plus_2] Affymetrix Human Genome U133 Plus 2.0 Array
Organism: Homo sapiens
Experiment type: Expression profiling by array
Summary

To compare the gene expression profiles of unpassaged, proliferating HUVEC and human iris, retinal and choroidal microvascular endothelial cells. Gene expression profiling revealed significant differences between HUVEC and ocular microvascular endothelial cells suggesting that HUVE cells may not be a suitable surrogate when studying pathophysiological mechanisms of ocular disorders. There were significant differences in the gene expression of important cell signalling pathways in human retinal and choroidal ECs. These differences may be important in the mechanisms and treatment of choroidal and retinal neovascularisation.

Overall design
12 arrays are included. Endothelial cells were derived from 4 tissues: iris, retina, choroid and human umbilical vein. RNA extracts from cells were hybridised to Affymetrix HGU133plus2 arrays in triplicate.

Samples

- GSM524662 — iris-cellline-1
- GSM524663 — retina-cellline-1
- GSM524664 — retina-cellline-2
- GSM524665 — iris-cellline-2
- GSM524666 — retina-cellline-3
- GSM524667 — iris-cellline-3
- GSM524668 — choroid-cellline-1
- GSM524669 — choroid-cellline-2
- GSM524670 — choroid-cellline-3
- GSM524671 — huvec-cellline-1
- GSM524672 — huvec-cellline-2
- GSM524673 — huvec-cellline-3

Usage

```
GSE20986_design
```

Format

A 13 column table defined as such:

- The row names — these are the microarray probe names used in the analysis.
- GSM524662.CEL — The microarray expression values corresponding to iris-cellline-1.
- GSM524663.CEL — The microarray expression values corresponding to retina-cellline-1.
- GSM524664.CEL — The microarray expression values corresponding to retina-cellline-2.
- GSM524665.CEL — The microarray expression values corresponding to iris-cellline-2.
- GSM524666.CEL — The microarray expression values corresponding to retina-cellline-3.
- GSM524667.CEL — The microarray expression values corresponding to iris-cellline-3.
- GSM524668.CEL — The microarray expression values corresponding to choroid-cellline-1.
- GSM524669.CEL — The microarray expression values corresponding to choroid-cellline-2.
- GSM524670.CEL — The microarray expression values corresponding to choroid-cellline-3.
- GSM524671.CEL — The microarray expression values corresponding to huvec-cellline-1.
- GSM524672.CEL — The microarray expression values corresponding to huvec-cellline-2.
- GSM524673.CEL — The microarray expression values corresponding to huvec-cellline-3.

Source

Geo Accession #Series GSE20986

References

Browning, A. C., Halligan, E. P., Swan, D. C., et al. (Jan 2012) "Comparative Gene Expression Profiling of HUVEC and Ocular Vascular Endothelial Cells." *Br J Ophthalmol.* Vol. **96**, No. 1, p. 128–132. doi: 10.1136/bjophthalmol-2011-300572.

---

null_variance *determines the variance for a null Elbow curve*

Description

determine an upper and lower error limit using all possible subsets of differences between initial conditions and then use the median value for each probe to create the null set. Maximum and minimum values are used to set upper and lower error bounds.

Usage

```
null_variance(my_data, upper_limit, lower_limit,
  initial_conditions)
```

Arguments

my_data	A table to analyze the null variance of. The columns in the table should be as follows: • probes — one column containing the names of the probes • initial conditions — one or more columns containing the initial conditions of the experiment • final conditions — one or more columns containing the final conditions of the experiment
upper_limit	the actual upper limit cut-off for the Elbow curve
lower_limit	the actual lower limit cut-off for the Elbow curve
initial_conditions	a data set of replicates corresponding to initial conditions.

Value

A list containing the following keys:

- prmax — the error limit based on the maximum value for each probe
- prmed — the null (median) value for each probe
- prmin — the error limit based on the minimum value for each probe

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]
```

```
# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

my_data <- replicates_to_fold(probes, initial_conditions, final_conditions)

# compute the elbow for the dataset
limits <- do_elbow(data.frame(my_data$fold))

plus_minus <- elbow_variance(probes, initial_conditions, final_conditions)

max_upper_variance <- plus_minus$max_upper
min_upper_variance <- plus_minus$min_upper
max_lower_variance <- plus_minus$max_lower
min_lower_variance <- plus_minus$min_lower

# rounded number for nice appearance graph
upper_limit <- round(limits[[1]],digits=2)

# rounded number nice appearance for graph
lower_limit <- round(limits[[2]],digits=2)

p_limits <- null_variance(my_data, upper_limit, lower_limit, initial_conditions)

prowmin <- p_limits[[1]]
prowmax <- p_limits[[2]]
prowmedian <- p_limits[[3]]
```

`plot_dataset` *plots data to see if it matches an elbow distribution.*

Description

A debug function for plotting data. This function plots an array or R object, and optionally draws the ELBOW limits on the plot. This function is mainly useful for the development and maintenance of the ELBOW method R package.

Usage

```
plot_dataset(my_data, column = NULL, upper_limit = NULL,
  lower_limit = NULL)
```

Arguments

`my_data`	is the object, or array, to plot.
`column`	the column in the object/array to plot, if applicable.
`upper_limit`	the upper ELBOW limit for the data (optional)
`lower_limit`	the lower ELBOW limit for the data (optional)

Examples

```
# install the DESeq libraries
#source("http://www.bioconductor.org/biocLite.R")
#biocLite("DESeq")

## download the table
library("DESeq")

# the following bam file dataset was obtained from:
# http://cgrlucb.wikispaces.com/file/view/yeast_sample_data.txt
# it has been downloaded into this package for speed convenience.
filename <- system.file("extdata", "yeast_sample_data.txt", package = "ELBOW")

count_table <- read.table(filename, header=TRUE, sep="\t", row.names=1)
expt_design <- data.frame(row.names = colnames(count_table), condition = c("WE","WE","M","M","M"))
conditions = expt_design$condition
data <- newCountDataSet(count_table, conditions)
data <- estimateSizeFactors(data)
data <- as(data, "CountDataSet")
## data <- estimateVarianceFunctions(data)
data <- estimateDispersions(data)


# this next step is essential, but it takes a long time...
# so, just like a good cooking show we will skip this step
# and load a finished version.
#results <- nbinomTest(data, "M", "WE")

# The below two code lines load a copy of the above dataset
# which has already been processed by:
#     results <- nbinomTest(data, "M", "WE")
# For your own real data, you must use:
#     results <- nbinomTest(data, "M", "WE")'
# Instead of the two lines below:
data(yeast_nbinomTest_results, package="ELBOW")
results <- yeast_nbinomTest_results

# plot the results w/o elbow (useful if do_elbow doesn't work)
#plot_dataset(results, "log2FoldChange")

# plot the results w/elbow
limits <- do_elbow_rnaseq(results)
plot_dataset(results, "log2FoldChange", limits$up_limit, limits$low_limit)
```

`plot_elbow` *plots Elbow curve data.*

Description

Plots an elbow curve and its associated data:

- the upper and lower elbow limits for the curve
- the upper, lower, and median initial condition Elbow plots
- the $\log \chi^2$ p-value for the Elbow curve
- the variance for the upper and lower Elbow cut-off values

Usage

```
plot_elbow(my_data, upper_limit, lower_limit, pvalue1,
  prowmin, prowmax, prowmedian, max_upper_variance,
  min_upper_variance, max_lower_variance,
  min_lower_variance, gtitle = "")
```

Arguments

`my_data`	A table (data.frame) to plot. The columns in the table should be as follows: • probes — one column containing the names of the probes • fold — the fold values for the table
`upper_limit`	the upper limit/cut-off for the elbow.
`lower_limit`	the lower limit/cut-off for the elbow.
`pvalue1`	the $\log \chi^2$ p-value for the elbow curve.
`prowmin`	the error limit based on the maximum value for each probe.
`prowmax`	the error limit based on the minimum value for each probe.
`prowmedian`	the null (median) value for each probe.
`max_upper_variance`	the maximum upper elbow limit *(most positive)*.
`min_upper_variance`	the minimum upper elbow limit.
`max_lower_variance`	the maximum lower elbow limit.
`min_lower_variance`	the minimum lower elbow limit *(most negative)*.
`gtitle`	the title to display for the graph.

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3
```

```
# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")

my_data <- replicates_to_fold(probes, initial_conditions, final_conditions)

# compute the elbow for the dataset
limits <- do_elbow(data.frame(my_data$fold))

plus_minus <- elbow_variance(probes, initial_conditions, final_conditions)

max_upper_variance <- plus_minus$max_upper
min_upper_variance <- plus_minus$min_upper
max_lower_variance <- plus_minus$max_lower
min_lower_variance <- plus_minus$min_lower

# rounded number for nice appearance graph
upper_limit <- round(limits[[1]],digits=2)

# rounded number nice appearance for graph
lower_limit <- round(limits[[2]],digits=2)

p_limits <- null_variance(my_data, upper_limit, lower_limit, initial_conditions)

prowmin <- p_limits[[1]]
prowmax <- p_limits[[2]]
prowmedian <- p_limits[[3]]

pvalue1 <- get_pvalue(my_data, upper_limit, lower_limit)
plot_elbow(my_data, upper_limit, lower_limit, pvalue1, prowmin, prowmax, prowmedian, max_upper_variance, min_upp
```

`replicates_to_fold` *calculates fold values from a table of experimental data*

Description

Converts a data.frame of probes with initial and final experimental conditions to a table with probes and fold change values.

Usage

```
replicates_to_fold(probes, initial_conditions,
  final_conditions)
```

Arguments

probes	the data set of probes.
initial_conditions	a data set of replicates corresponding to initial conditions.
final_conditions	a data set of replicates corresponding to final conditions.

Value

a data.frame comprised of probe names and fold-change values

Examples

```
# read in the EcoliMutMA sample data from the package
data(EcoliMutMA, package="ELBOW")
csv_data <- EcoliMutMA
# - OR - Read in a CSV file (uncomment - remove the #'s
#        - from the line below and replace 'filename' with
#        the CSV file's filename)
# csv_data <- read.csv(filename)

# set the number of initial and final condition replicates both to three
init_count  <- 3
final_count <- 3

# Parse the probes, intial conditions and final conditions
# out of the CSV file.  Please see: extract_working_sets
# for more information.
#
# init_count should be the number of columns associated with
#       the initial conditions of the experiment.
# final_count should be the number of columns associated with
#       the final conditions of the experiment.
working_sets <- extract_working_sets(csv_data, init_count, final_count)

probes <- working_sets[[1]]
initial_conditions <- working_sets[[2]]
final_conditions <- working_sets[[3]]

# Uncomment to output the plot to a PNG file (optional)
# png(file="output_plot.png")
```

```
my_data <- replicates_to_fold(probes, initial_conditions, final_conditions)
```

`yeast_nbinomTest_results`

Pre-processed Yeast sample RNA-seq data

Description

Pre-processed Yeast sample RNA-seq data, obtained from the website: http://cgrlucb.wikispaces.com/Spring+2012+DESeq+Tutorial

For more information, please contact the author of the webpage.

Usage

```
yeast_nbinomTest_results
```

Format

An 8 column table defined as such:

- id — the Gene ID reprented by the row
- baseMean — mean estimated from both conditions (with normalization by size factors)
- baseMeanA — mean estimated from the first condition specified as an argument to nbinomTest (in this case, "M")
- baseMeanB — mean estimated from the second condition (in this case, "WE")
- foldChange — ratio of the estimated means
- log2FoldChange — log2 of the ratio
- pval — uncorrected p-value from the negative binomial test
- padj — p-value adjusted for multiple testing using Benjamini-Hochberg to estimate the false discovery rate
- resVarA — dispersion estimated from the first condition
- resVarB — dispersion estimated from the second condition

Source

http://cgrlucb.wikispaces.com/Spring+2012+DESeq+Tutorial

References

Lee, H. "Spring 2012 DESeq Tutorial." http://cgrlucb.wikispaces.com/Spring+2012+DESeq+Tutorial

Index

∗Topic **ELBOW**
ELBOW, 10
∗Topic **datasets**
EcoliMutMA, 6
EcoliWT, 8
GSE20986_design, 18
GSE20986_eset_exprs, 20
yeast_nbinomTest_results, 28
∗Topic **package**
ELBOW, 10

analyze_elbow, 2, *12*

do_elbow, 3, *12*
do_elbow_rnaseq, 5

EcoliMutMA, 6
EcoliWT, 8
ELBOW, 10
elbow_variance, 12
elbowlib *(*ELBOW*)*, 10
extract_working_sets, 14

get_elbow_limma, 15
get_pvalue, 16
GSE20986_design, 18
GSE20986_eset_exprs, 20

null_variance, 21

pkg_elbowlib *(*ELBOW*)*, 10
pkg_elbowlib-package *(*ELBOW*)*, 10
plot_dataset, 23
plot_elbow, 24

replicates_to_fold, 26

yeast_nbinomTest_results, 28