Package 'DeLorean'

October 18, 2018

Title Estimates Pseudotimes for Single Cell Expression Data

Author John Reid <johnbaronreid@gmail.com>

Maintainer John Reid <johnbaronreid@gmail.com>

Version 1.5.0

Description Implements the DeLorean model (Reid & Wernisch (2016) <doi:10.1093/bioinformatics/btw372>) to estimate pseudotimes for single cell expression data. The DeLorean model uses a Gaussian process latent variable model to model uncertainty in the capture time of cross-sectional data.

Depends methods, R (>= 3.4.0), Rcpp (>= 0.12.19)

Imports rstan (>= 2.18.1), rstantools (>= 1.5.1), dplyr (>= 0.4.3), reshape2 (>= 1.4), stringr (>= 0.6.2), ggplot2 (>= 1.0.0), MASS (>= 7.3), broom, coda, parallel, functional, kernlab, fastICA, seriation, lattice, memoise

LinkingTo BH (>= 1.66.0-1), Rcpp (>= 0.12.19), RcppEigen (>= 0.3.3.4.0), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

Suggests knitr (>= 1.8), knitcitations, rmarkdown, formatR, extrafont, testthat, svglite, VGAM

Encoding UTF-8

LazyData true

NeedsCompilation yes

SystemRequirements GNU make

RoxygenNote 6.1.0

License MIT + file LICENSE

VignetteBuilder knitr

Repository CRAN

Date/Publication 2018-10-17 22:30:16 UTC

R topics documented:

`DeLorean-package` *The 'DeLorean' package.*

Description

Fits the DeLorean pseudotime model.

`adjust.by.cell.sizes` *Adjust the expression by the estimated cell sizes.*

Description

Adjust the expression by the estimated cell sizes.

Usage

```
adjust.by.cell.sizes(dl)
```

Arguments

`dl`	de.lorean object.

`alpha.for.rug` *Calculate a suitable value for a rug plot given the number of points*

Description

Calculate a suitable value for a rug plot given the number of points

Usage

```
alpha.for.rug(n, scale = 100)
```

Arguments

`n`	Number of points.
`scale`	Scale the value.

`analyse.noise.levels` *Analyse noise levels and assess which genes have the greatest ratio of temporal variance to noise. This are labelled as the 'gene.high.psi' genes.*

Description

Analyse noise levels and assess which genes have the greatest ratio of temporal variance to noise. This are labelled as the 'gene.high.psi' genes.

Usage

```
analyse.noise.levels(dl, num.high.psi = 25)
```

Arguments

`dl`	de.lorean object
`num.high.psi`	How many genes with high variance to examine

`analyse.variance` *Analyse variance of expression between and within capture times.*

Description

Analyse variance of expression between and within capture times.

Usage

```
analyse.variance(dl, adjust.cell.sizes)
```

Arguments

`dl`	de.lorean object
`adjust.cell.sizes`	Choose whether to adjust the expression values by the cell size estimates

---

`anders.huber.cell.sizes` *Estimate the cell sizes according to Anders & Huber Differential expression analysis for sequence count data*

Description

Estimate the cell sizes according to Anders & Huber Differential expression analysis for sequence count data

Usage

```
anders.huber.cell.sizes(expr.l)
```

Arguments

`expr.l`	Melted expression values.

`aov.dl` *Perform an analysis of variance to select genes for the DeLorean model.*

Description

Perform an analysis of variance to select genes for the DeLorean model.

Usage

```
aov.dl(dl)
```

Arguments

`dl`	DeLorean object.

`avg.par.samples` *Average across a parameters samples.*

Description

Average across a parameters samples.

Usage

```
avg.par.samples(s)
```

Arguments

`s`	An array of any dimension in which the first dimensions indexes the samples

`calc.inducing.pseudotimes` *Calculate inducing pseudotimes for sparse approximation*

Description

Calculate inducing pseudotimes for sparse approximation

Usage

```
calc.inducing.pseudotimes(dl, num.inducing, period = 0,
  num.sd.border = 7)
```

Arguments

`dl`	de.lorean object
`num.inducing`	Number of inducing points
`period`	Period of expression patterns
`num.sd.border`	The size of the border of the inducing inputs around the capture times in units of number of standard deviations

`calc.roughness` *Calculate the roughness of the vector. The roughness is the RMS of the differences between consecutive points.*

Description

Calculate the roughness of the vector. The roughness is the RMS of the differences between consecutive points.

Usage

```
calc.roughness(x)
```

Arguments

`x`	Values

`centralise` *Centralises a periodic position into [period/2, period) by shifting by n*period, where n is an integer*

Description

Centralises a periodic position into [period/2, period) by shifting by n*period, where n is an integer

Usage

```
centralise(x, period = 1)
```

Arguments

`x`	Position
`period`	Period

cmp.profiles.plot *Plot a comparison of the profiles from several de.lorean objects*

Description

Plot a comparison of the profiles from several de.lorean objects

Usage

```
cmp.profiles.plot(..., genes = NULL)
```

Arguments

...	Named de.lorean objects
genes	Genes to plot (defaults to genes.high.psi of first de.lorean object)

cov.all.genes.conditioned *Calculate covariances for all genes when conditioned on data at estimated pseudotimes.*

Description

Calculate covariances for all genes when conditioned on data at estimated pseudotimes.

Usage

```
cov.all.genes.conditioned(dl, cov.fn = NULL, tau = tau.for.sample(dl))
```

Arguments

dl	de.lorean object
cov.fn	Covariance function (defaults to cov.matern.32)
tau	The pseudotimes to use

cov.calc.dists *Calculate distances between vectors of time points*

Description

Calculate distances between vectors of time points

Usage

```
cov.calc.dists(tau.1, tau.2 = tau.1, period = NULL)
```

Arguments

tau.1	First vector of time points
tau.2	Second vector of time points (defaults to first if not given)
period	Period if periodic

cov.calc.dl.dists *Calculate distances over estimated pseudotimes and test inputs.*

Description

Calculate distances over estimated pseudotimes and test inputs.

Usage

```
cov.calc.dl.dists(dl, tau = tau.for.sample(dl), include.test = TRUE)
```

Arguments

dl	de.lorean object
tau	The pseudotimes to use
include.test	Also include the pseudotimes for the test inputs

`cov.calc.gene` *Calculate covariance structure for gene over pseudotimes and test inputs.*

Description

Calculate covariance structure for gene over pseudotimes and test inputs.

Usage

```
cov.calc.gene(dl, gene.idx, cov.fn = cov.matern.32,
  tau = tau.for.sample(dl), include.test = TRUE,
  psi = sampled.gene.param(dl, gene.idx, "psi"),
  omega = sampled.gene.param(dl, gene.idx, "omega"))
```

Arguments

`dl`	de.lorean object
`gene.idx`	Gene index
`cov.fn`	Covariance function (defaults to cov.matern.32)
`tau`	The pseudotimes to use
`include.test`	Also include the pseudotimes for the test inputs
`psi`	Temporal variation
`omega`	Noise

`cov.calc.gene.conditioned` *Calculate covariance for gene over test inputs when conditioned on data at estimated pseudotimes.*

Description

Calculate covariance for gene over test inputs when conditioned on data at estimated pseudotimes.

Usage

```
cov.calc.gene.conditioned(dl, gene.idx, cov.fn = NULL,
  tau = tau.for.sample(dl))
```

Arguments

`dl`	de.lorean object
`gene.idx`	Gene index
`cov.fn`	Covariance function (defaults to cov.matern.32)
`tau`	The pseudotimes to use

cov.matern.32 *Matern 3/2 covariance function*

Description

Matern 3/2 covariance function

Usage

```
cov.matern.32(r, l)
```

Arguments

r	Distance
l	Length scale

cov.periodise *Makes a distance periodic*

Description

Makes a distance periodic

Usage

```
cov.periodise(r, period)
```

Arguments

r	Distance
period	The period

create.ordering.ll.fn *Calculate the covariance structure of evenly spread tau and create a function that calculates the log likelihood of orderings.*

Description

Calculate the covariance structure of evenly spread tau and create a function that calculates the log likelihood of orderings.

Usage

```
create.ordering.ll.fn(dl, cov.fn = cov.fn.for(dl))
```

Arguments

dl	The DeLorean object
cov.fn	The covariance function

de.lorean *Initialise DeLorean object*

Description

Initialise DeLorean object

Usage

```
de.lorean(expr, gene.meta, cell.meta)
```

Arguments

expr	Expression array
gene.meta	Data frame of meta data for genes
cell.meta	Data frame of meta data for cells

Examples

```
data(WindramDeLorean)
dl <- de.lorean(windram.expr, windram.gene.meta, windram.cell.meta)
```

de.lorean.stylesheet *The filename of the R markdown stylesheet*

Description

The filename of the R markdown stylesheet

Usage

```
de.lorean.stylesheet()
```

default.num.cores *Default number of cores to use.*

Description

Default number of cores to use.

Usage

```
default.num.cores()
```

dim.de.lorean *Dimensions of DeLorean object*

Description

Dimensions of DeLorean object

Usage

```
## S3 method for class 'de.lorean'
dim(x)
```

Arguments

x	De lorean object

`estimate.cell.sizes` *Estimate the cell sizes. We only consider genes that are expressed in a certain proportion of cells.*

Description

Estimate the cell sizes. We only consider genes that are expressed in a certain proportion of cells.

Usage

```
estimate.cell.sizes(dl, cell.prop = 0.5, expr.threshold = 0,
  by.capture = TRUE)
```

Arguments

`dl`	de.lorean object.
`cell.prop`	The proportion of cells a gene must be expressed in to be considered for cell size estimation
`expr.threshold`	The threshold above which we consider a gene to be expressed
`by.capture`	Estimate the cell sizes by considering the cells at each capture time separately

`estimate.hyper` *Estimate hyperparameters for model using empirical Bayes.*

Description

Estimate hyperparameters for model using empirical Bayes.

Usage

```
estimate.hyper(dl, sigma.tau = 0.5, length.scale = NULL,
  model.name = "exact", adjust.cell.sizes = TRUE)
```

Arguments

`dl`	de.lorean object
`sigma.tau`	Noise s.d. in temporal dimension, that is prior s.d. for tau
`length.scale`	Length scale for stationary GP covariance function. Defaults to the range of the observed capture times.
`model.name`	The model's name: • 'exact': The model without a low rank approximation that does not estimate the cell sizes. • 'exactsizes': The model without a low rank approximation that does estimate the cell sizes.

- 'lowrank': Low rank approximation to the 'exact' model.
- 'lowranksizes': Low rank approximation to the 'exactsizes' model.

`adjust.cell.sizes`
: Adjust by the cell sizes for better estimates of the hyperparameters

Examples

```
data(WindramDeLorean)
dl <- de.lorean(windram.expr, windram.gene.meta, windram.cell.meta)
dl <- estimate.hyper(dl)
```

`examine.convergence` *Analyse the samples and gather the convergence statistics. Note this only makes sense if a sampling method was used to fit the model as opposed to variational Bayes.*

Description

Analyse the samples and gather the convergence statistics. Note this only makes sense if a sampling method was used to fit the model as opposed to variational Bayes.

Usage

```
examine.convergence(dl)
```

Arguments

`dl`	de.lorean object

`expected.sample.var` *The expected within sample variance of a Gaussian with the given covariance.*

Description

The expected within sample variance of a Gaussian with the given covariance.

Usage

```
expected.sample.var(K)
```

Arguments

`K`	Covariance

expr.data.plot *Plot the expression data by the capture points*

Description

Plot the expression data by the capture points

Usage

```
expr.data.plot(dl, genes = NULL, num.genes = 12)
```

Arguments

dl	de.lorean object
genes	Genes to plot. If NULL plots some random varying genes
num.genes	Number of genes to plot

filter_cells *Filter cells*

Description

Filter cells

Usage

```
filter_cells(dl, .filter = function(x) x %in% cells, number = NULL,
  cells = sample(colnames(dl$expr), number))
```

Arguments

dl	de.lorean object
.filter	Function that takes a list of cells as input and returns a vector of TRUE/FALSE
number	Number to sample if filter function or cells not supplied.
cells	The cells to keep.

Examples

```
data(WindramDeLorean)
dl <- de.lorean(windram.expr, windram.gene.meta, windram.cell.meta)
dl <- filter_cells(dl, number = 7)
```

filter_genes *Filter genes*

Description

Filter genes

Usage

```
filter_genes(dl, .filter = function(x) x %in% genes, number = NULL,
  genes = sample(rownames(dl$expr), number))
```

Arguments

dl	de.lorean object
.filter	Function that takes a list of genes as input and returns a vector of TRUE/FALSE
number	Number to sample if filter function or genes not supplied.
genes	The genes to keep.

Examples

```
data(WindramDeLorean)
dl <- de.lorean(windram.expr, windram.gene.meta, windram.cell.meta)
dl <- filter_genes(dl, number = 37)
```

find.best.tau *Find best tau to initialise chains with by sampling tau from the prior and using empirical Bayes parameter estimates for the other parameters.*

Description

Find best tau to initialise chains with by sampling tau from the prior and using empirical Bayes parameter estimates for the other parameters.

Usage

```
find.best.tau(dl, num.tau.candidates = 6000,
  num.tau.to.keep = num.cores, num.cores = default.num.cores())
```

Arguments

dl	de.lorean object
num.tau.candidates	How many candidates to examine. Defaults to 6000.
num.tau.to.keep	How many candidates to keep. Defaults to num.cores.
num.cores	Number of cores to run on. Defaults to default.num.cores()

---

find.good.ordering *Run a find good ordering method and append results to existing orderings*

Description

Run a find good ordering method and append results to existing orderings

Usage

```
find.good.ordering(dl, method, ...)
```

Arguments

dl	de.lorean object
method	Function that runs the method
...	Any other arguments for the method

---

find.smooth.tau *Find best order of the samples assuming some smooth GP prior on the expression profiles over this ordering.*

Description

Find best order of the samples assuming some smooth GP prior on the expression profiles over this ordering.

Usage

```
find.smooth.tau(dl, psi = exp(dl$hyper$mu_psi),
  omega = exp(dl$hyper$mu_omega), num.cores = default.num.cores(),
  num.tau.to.try = num.cores, num.tau.to.keep = num.cores,
  method = "metropolis", ...)
```

Arguments

dl	de.lorean object
psi	Temporal variation
omega	Noise
num.cores	Number of cores to run on. Defaults to default.num.cores()
num.tau.to.try	How many initialisations to try
num.tau.to.keep	How many initialisations to keep
method	Method to use "maximise" or "metropolis"
...	Extra arguments to method

`fit.dl` *Perform all the steps necessary to fit the model:*

1. *prepare the data*
2. *find suitable initialisations*
3. *fit the model using the specified method (sampling or variational Bayes)*
4. *process the posterior*

Description

Perform all the steps necessary to fit the model:

1. prepare the data
2. find suitable initialisations
3. fit the model using the specified method (sampling or variational Bayes)
4. process the posterior

Usage

```
fit.dl(dl, method = "sample", ...)
```

Arguments

dl	de.lorean object
method	Fitting method: • 'sample': Use a Stan sampler. See `fit.model.sample`. • 'vb': Use Stan ADVI variational Bayes algorithm. See `fit.model.vb`.
...	Extra arguments for fitting method

`fit.held.out` *Fit held out genes*

Description

Fit held out genes

Usage

```
fit.held.out(dl, expr.held.out, sample.iter = dl$best.sample)
```

Arguments

`dl`	de.lorean object
`expr.held.out`	The expression matrix including the held out genes
`sample.iter`	The sample to use to fit with
	Calculate covariance over pseudotimes and capture times Evaluate the held out gene under the GP model using pseudotimes and a model without.

`fit.model` *Fit the model using specified method (sampling or variational Bayes).*

Description

Fit the model using specified method (sampling or variational Bayes).

Usage

```
fit.model(dl, method = "sample", ...)
```

Arguments

`dl`	de.lorean object
`method`	Fitting method: • 'sample': Use a Stan sampler. See `fit.model.sample`. • 'vb': Use Stan ADVI variational Bayes algorithm. See `fit.model.vb`.
`...`	Extra arguments for method

fit.model.sample *Fit the model using Stan sampler*

Description

Fit the model using Stan sampler

Usage

```
fit.model.sample(dl, num.cores = getOption("DL.num.cores",
  max(parallel::detectCores() - 1, 1)), chains = 1, thin = 50, ...)
```

Arguments

dl	de.lorean object
num.cores	Number of cores to run on. Defaults to getOption("DL.num.cores", max(parallel::detectCores()-1, 1))
chains	Number of chains to run on each core
thin	How many samples to generate before retaining one
...	Extra arguments for rstan::stan() sampling call

fit.model.vb *Fit the model using Stan variational Bayes*

Description

Fit the model using Stan variational Bayes

Usage

```
fit.model.vb(dl, num.cores = default.num.cores(),
  num.inits = num.cores, init.idx = 1, ...)
```

Arguments

dl	de.lorean object
num.cores	Number of cores to run on. Defaults to default.num.cores()
num.inits	Number initialisations to try. Defaults to num.cores
init.idx	Which initialisation to use if only using one
...	Extra arguments for rstan::vb()

gaussian.condition — *Condition a Gaussian on another. See Eqn. A.6 on page 200 of Rasmussen and Williams' book.*

Description

Condition a Gaussian on another. See Eqn. A.6 on page 200 of Rasmussen and Williams' book.

Usage

```
gaussian.condition(y, .A, .B, .C, mu.x = rep(0, nrow(.A)),
  mu.y = rep(0, nrow(.B)), U = chol(.B))
```

Arguments

y	y
.A	Var(X)
.B	Var(Y)
.C	Cov(X, Y)
mu.x	Mean of x
mu.y	Mean of y
U	Cholesky decomposition of .B

gene.covariances — *Calculate the covariance structure of the tau*

Description

Calculate the covariance structure of the tau

Usage

```
gene.covariances(cov.fn, length.scale, period = NULL, tau, psi, omega)
```

Arguments

cov.fn	Covariance function
length.scale	Length scale
period	The period of the covariance function
tau	Pseudotimes
psi	The temporal variation for each gene
omega	The noise level for each gene

`get.posterior.mean` *Get posterior mean of samples*

Description

Get posterior mean of samples

Usage

```
get.posterior.mean(extract)
```

Arguments

`extract`	A named list of samples

`get_model` *Get the Stan model for a DeLorean object.*

Description

Get the Stan model for a DeLorean object.

Usage

```
get_model(dl)
```

Arguments

`dl`	de.lorean object

`gp.log.marg.like` *The log marginal likelihood. See "2.3 Varying the Hyperparameters" on page 19 of Rasmussen and Williams' book.*

Description

The log marginal likelihood. See "2.3 Varying the Hyperparameters" on page 19 of Rasmussen and Williams' book.

Usage

```
gp.log.marg.like(y, K = NULL, U = chol(K))
```

Arguments

y	The targets.
K	The covariance matrix (kernel), not needed if U is provided.
U	Cholesky decomposition of K (chol(K)).

`gp.predict` *Predictive mean, variance and log marginal likelihood of a GP. See "2.3 Varying the Hyperparameters" on page 19 of Rasmussen and Williams' book.*

Description

Predictive mean, variance and log marginal likelihood of a GP. See "2.3 Varying the Hyperparameters" on page 19 of Rasmussen and Williams' book.

Usage

```
gp.predict(y, K = NULL, Kstar, Kstarstar, U = chol(K))
```

Arguments

y	The targets.
K	The covariance matrix (kernel) for input points, not needed if U is provided.
Kstar	The cross covariance matrix (kernel)
Kstarstar	The cross covariance matrix (kernel) for test points
U	Cholesky decomposition of K

`gp.predictions.df` *Convert the output of gp.predict() into a data.frame.*

Description

Convert the output of gp.predict() into a data.frame.

Usage

```
gp.predictions.df(predictions)
```

Arguments

predictions	The predictions

guo.expr — *Single cell expression data and meta data from Guo et al. (2012). They investigated the expression of 48 genes in 500 mouse embryonic cells.*

Description

Single cell expression data and meta data from Guo et al. (2012). They investigated the expression of 48 genes in 500 mouse embryonic cells.

Usage

```
data(GuoDeLorean)
```

Format

There are three objects in this data:

- guo.expr A matrix of log expression values with no missing data. Rows are named by genes and columns are named by cells/samples.
- guo.gene.meta A data frame containing meta-data about the genes.
- guo.cell.meta A data frame containing meta-data about the cells

Source

http://www.sciencedirect.com/science/article/pii/S1534580710001103

held.out.melt — *Melt held out genes*

Description

Melt held out genes

Usage

```
held.out.melt(dl, expr, held.out.genes)
```

Arguments

dl	de.lorean object
expr	Expression matrix of all genes
held.out.genes	Genes to hold out

`held.out.posterior` *Calculate posterior covariance and estimate parameters for held out genes given pseudotimes estimated by DeLorean model.*

Description

Calculate posterior covariance and estimate parameters for held out genes given pseudotimes estimated by DeLorean model.

Usage

```
held.out.posterior(dl, held.out, posterior.sample = dl$best.sample)
```

Arguments

`dl`	de.lorean object
`held.out`	Held out gene expression levels
`posterior.sample`	Posterior sample to use

`held.out.posterior.by.variation` *Order the genes by the variation of their posterior mean*

Description

Order the genes by the variation of their posterior mean

Usage

```
held.out.posterior.by.variation(posterior)
```

Arguments

`posterior`	The posterior of some held out genes

`held.out.posterior.filter` *Filter the genes*

Description

Filter the genes

Usage

```
held.out.posterior.filter(posterior, genes)
```

Arguments

`posterior`	The posterior of some held out genes
`genes`	Genes to filter

`held.out.posterior.join` *Join with another data frame. Useful for adding gene names etc..*

Description

Join with another data frame. Useful for adding gene names etc..

Usage

```
held.out.posterior.join(posterior, .df)
```

Arguments

`posterior`	The posterior of some held out genes
`.df`	Data frame to join with

held.out.select.genes *Select held out genes by those with highest variance*

Description

Select held out genes by those with highest variance

Usage

```
held.out.select.genes(dl, expr, num.held.out)
```

Arguments

dl	de.lorean object
expr	Expression matrix of all genes
num.held.out	Number to select

inducing.covariance *Calculate the covariance structure of the inducing points*

Description

Calculate the covariance structure of the inducing points

Usage

```
inducing.covariance(cov.fn, length.scale, period = NULL, tau,
  num.inducing, u = seq(min(tau), max(tau), length.out = num.inducing))
```

Arguments

cov.fn	Covariance function
length.scale	Length scale
period	The period of the covariance function
tau	Pseudotimes
num.inducing	How many inducing points to use
u	The inducing points

init.orderings.vs.pseudotimes.plot

Plot the orderings for initialisation against the estimated pseudotime.

Description

Plot the orderings for initialisation against the estimated pseudotime.

Usage

```
init.orderings.vs.pseudotimes.plot(dl, sample.iter = dl$best.sample)
```

Arguments

dl	The DeLorean object
sample.iter	Which sample to take pseudotimes from

is.de.lorean

Is a DeLorean object?

Description

Is a DeLorean object?

Usage

```
is.de.lorean(x)
```

Arguments

x	de.lorean object

`knit.report` *Knit a report, the file inst/Rmd/<report.name>.Rmd must exist in the package directory.*

Description

Knit a report, the file inst/Rmd/<report.name>.Rmd must exist in the package directory.

Usage

```
knit.report(dl, report.name)
```

Arguments

`dl`	de.lorean object
`report.name`	The name of the report. Used to locate the R markdown report file in the package.

`kouno.expr` *Kouno et al. investigated the transcriptional network controlling how THP-1 human myeloid monocytic leukemia cells differentiate into macrophages. They provide expression values for 45 genes in 960 single cells captured across 8 distinct time points.*

Description

Kouno et al. investigated the transcriptional network controlling how THP-1 human myeloid monocytic leukemia cells differentiate into macrophages. They provide expression values for 45 genes in 960 single cells captured across 8 distinct time points.

Usage

```
data(KounoDeLorean)
```

Format

There are three objects in this data:

- kouno.expr A matrix of log expression values with no missing data. Rows are named by genes and columns are named by cells/samples.
- kouno.gene.meta A data frame containing meta-data about the genes.
- kouno.cell.meta A data frame containing meta-data about the cells

Source

http://genomebiology.com/2013/14/10/R118/abstract

make.fit.valid *Make a fit valid by running one iteration of the sampler.*

Description

Make a fit valid by running one iteration of the sampler.

Usage

```
make.fit.valid(dl)
```

Arguments

dl	de.lorean object

make.init.fn *Returns a function that constructs parameter settings with good tau.*

Description

Returns a function that constructs parameter settings with good tau.

Usage

```
make.init.fn(dl)
```

Arguments

dl	de.lorean object

make.predictions *Make predictions*

Description

Make predictions

Usage

```
make.predictions(dl)
```

Arguments

dl	de.lorean object

marg.like.plot — *Plot posterior for marginal log likelihoods of individual gene's expression profiles*

Description

Plot posterior for marginal log likelihoods of individual gene's expression profiles

Usage

```
marg.like.plot(dl)
```

Arguments

dl	de.lorean object

melt.expr — *Melt an expression matrix.*

Description

Melt an expression matrix.

Usage

```
## S3 method for class 'expr'
melt(dl, expr = dl$expr)
```

Arguments

dl	The de.lorean object.
expr	Matrix of expression values.

mutate.profile.data *Mutate the profile data into shape compatible with GP plot function*

Description

Mutate the profile data into shape compatible with GP plot function

Usage

```
## S3 method for class 'profile.data'
mutate(.data)
```

Arguments

.data	The data

optimise.best.sample *Optimise the best sample and update the best.sample index.*

Description

Optimise the best sample and update the best.sample index.

Usage

```
optimise.best.sample(dl, sample.to.opt = dl$best.sample,
  new.best.sample = -1)
```

Arguments

dl	de.lorean object
sample.to.opt	Sample to optimise
new.best.sample	Update to best sample index

ordering.block.move *Move a block in an ordering and shift the other items.*

Description

Move a block in an ordering and shift the other items.

Usage

```
ordering.block.move(ordering, from, width, to, reverse = FALSE)
```

Arguments

ordering	The ordering.
from	The start index of the block to move.
width	The width of the block to move.
to	The index it should be moved to.
reverse	Reverse the block?

ordering.improve *Improve the ordering in the sense that some function is maximised.*

Description

Improve the ordering in the sense that some function is maximised.

Usage

```
ordering.improve(fn, ordering)
```

Arguments

fn	A function to maximise.
ordering	The permutation (ordering) to start from.

`ordering.invert` *Invert the ordering*

Description

Invert the ordering

Usage

```
ordering.invert(ordering)
```

Arguments

`ordering`	The permutation (ordering) to invert.

`ordering.is.valid` *Check that it is a valid ordering*

Description

Check that it is a valid ordering

Usage

```
ordering.is.valid(ordering, full.check = FALSE)
```

Arguments

`ordering`	The ordering
`full.check`	Perform a full check

`ordering.maximise` *Find a good ordering in the sense that some function is locally maximised.*

Description

Find a good ordering in the sense that some function is locally maximised.

Usage

```
ordering.maximise(ordering, fn)
```

Arguments

`ordering`	The permutation (ordering) to start from.
`fn`	A function to maximise.

ordering.metropolis.hastings *Metropolis-Hastings on orderings.*

Description

Metropolis-Hastings on orderings.

Usage

```
ordering.metropolis.hastings(ordering, log.likelihood,
  proposal.fn = ordering.random.move, iterations = 1000, thin = 1)
```

Arguments

ordering	Initial ordering
log.likelihood	Log likelihood function
proposal.fn	Proposal function
iterations	Number of iterations
thin	Thinning parameter

ordering.move *Move one item in an ordering and shift the other items.*

Description

Move one item in an ordering and shift the other items.

Usage

```
ordering.move(ordering, from, to)
```

Arguments

ordering	The ordering.
from	The index of the item to move.
to	The index it should be moved to.

`ordering.random.block.move` *Randomly move a block in an ordering to another location*

Description

Randomly move a block in an ordering to another location

Usage

```
ordering.random.block.move(ordering, max.width = 4)
```

Arguments

`ordering`	The ordering
`max.width`	The maximum width of the block

`ordering.random.move` *Randomly move one item in an ordering to another location*

Description

Randomly move one item in an ordering to another location

Usage

```
ordering.random.move(ordering)
```

Arguments

`ordering`	The ordering.

`ordering.test.score` *Test ordering score: sum every time consecutive items are in order.*

Description

Test ordering score: sum every time consecutive items are in order.

Usage

```
ordering.test.score(ordering)
```

Arguments

`ordering`	The permutation (ordering) to test.

`orderings.plot` *Plot likelihoods of orderings against elapsed times taken to generate them*

Description

Plot likelihoods of orderings against elapsed times taken to generate them

Usage

```
orderings.plot(dl)
```

Arguments

`dl`	The DeLorean object

`partition.de.lorean` *Partition de.lorean object by cells*

Description

Partition de.lorean object by cells

Usage

```
partition.de.lorean(dl, pieces = 2)
```

Arguments

`dl`	de.lorean object
`pieces`	How many pieces to partition into

`permute.df` *Permute a data frame, x. If group.col is given it should name an ordered factor that the order of the permutation should respect.*

Description

Permute a data frame, x. If group.col is given it should name an ordered factor that the order of the permutation should respect.

Usage

```
permute.df(.df, group.col = NULL)
```

Arguments

`.df`	Data frame
`group.col`	Name of an ordered factor that the permutation should respect.

`permuted.roughness` *Permute cells and test roughness of expression.*

Description

Permute cells and test roughness of expression.

Usage

```
permuted.roughness(dl, expr.held.out = dl$expr.held.out)
```

Arguments

`dl`	de.lorean object
`expr.held.out`	The expression matrix of the held out genes

plot.add.expr *Add expression data to a plot*

Description

Add expression data to a plot

Usage

```
## S3 method for class 'add.expr'
plot(gp, .data = NULL)
```

Arguments

gp	Plot object
.data	Expression data to add

plot.add.mean.and.variance *Add posterior representation to a plot.*

Description

Add posterior representation to a plot.

Usage

```
## S3 method for class 'add.mean.and.variance'
plot(gp, .data = NULL, color = "black",
  line.alpha = 0.3, ribbon.alpha = 0.1)
```

Arguments

gp	Plot object
.data	Data frame containing variables to plot (mean, var) phi, predictedvar)
color	Color to use
line.alpha	Alpha to use for mean line
ribbon.alpha	Alpha to use for variance ribbon

`plot.de.lorean` *Various DeLorean object plots*

Description

Various DeLorean object plots

Usage

```
## S3 method for class 'de.lorean'
plot(x, type = "profiles", ...)
```

Arguments

`x`	de.lorean object
`type`	Type of plot: • 'expr.data': The expression data plotted by capture time. See `expr.data.plot`. • 'Rhat': $hatR$ convergence statistics See `Rhat.plot`. • 'pseudotime': Pseudotimes in best posterior sample See `pseudotime.plot`. • 'profiles': Gene expression profiles for best posterior sample See `profiles.plot`. • 'tau.offsets': Offsets of pseudotimes to assess the prior See `tau.offsets.plot`. • 'marg.like': Plot the posterior of the marginal likelihoods for individual genes. See `marg.like.plot`. • 'roughnesses': Roughnesses of the pseudotime posterior See `roughnesses.plot`. • 'init.vs.pseudotimes': Plot the initialisations used against the pseudotimes estimated See `init.orderings.vs.pseudotimes.plot`.
`...`	Extra arguments to plot function

`plot.held.out.posterior`

Plot the posterior of held out genes

Description

Plot the posterior of held out genes

Usage

```
## S3 method for class 'held.out.posterior'
plot(dl, posterior, facets = ~gene)
```

Arguments

`dl`	de.lorean object
`posterior`	The posterior of some held out genes
`facets`	Variables to wrap facets on

prepare.for.stan *Prepare for Stan*

Description

Prepare for Stan

Usage

```
prepare.for.stan(dl, num.test = 101, num.inducing = 30, period = 0,
  hold.out = 0, num.sd.border = 7)
```

Arguments

dl	de.lorean object
num.test	Number of test points to consider
num.inducing	Number of inducing points
period	Period of expression patterns
hold.out	Number genes to hold out for generalisation tests
num.sd.border	The size of the border of the inducing inputs around the capture times in units of number of standard deviations

print.de.lorean *Print details of DeLorean object*

Description

Print details of DeLorean object

Usage

```
## S3 method for class 'de.lorean'
print(x, ...)
```

Arguments

x	de.lorean object
...	Extra arguments

`process.posterior` *Process the posterior, that is extract and reformat the samples from Stan. We also determine which sample has the highest likelihood, this is labelled as the 'best' sample.*

Description

Process the posterior, that is extract and reformat the samples from Stan. We also determine which sample has the highest likelihood, this is labelled as the 'best' sample.

Usage

```
process.posterior(dl)
```

Arguments

`dl`	de.lorean object

`profiles.plot` *Plot best sample predicted expression.*

Description

Plot best sample predicted expression.

Usage

```
profiles.plot(dl, genes = dl$genes.high.psi, profile.color = "black",
  add.data = T, sample.iter = dl$best.sample,
  ignore.cell.sizes = FALSE, ...)
```

Arguments

`dl`	de.lorean object
`genes`	Genes to plot (defaults to genes.high.psi)
`profile.color`	Colour for the profile
`add.data`	Add actual expression data to plot
`sample.iter`	Which sample to plot
`ignore.cell.sizes`	Ignore cell sizes if the model has estimated them
`...`	Extra arguments

`pseudotime.plot` *Plot pseudotime (tau) against observed capture time.*

Description

Plot pseudotime (tau) against observed capture time.

Usage

```
pseudotime.plot(dl, sample.iter = dl$best.sample)
```

Arguments

`dl`	de.lorean object
`sample.iter`	Which sample to take pseudotimes from

`pseudotimes.from.orderings` *Convert best orderings into initialisations*

Description

Convert best orderings into initialisations

Usage

```
pseudotimes.from.orderings(dl, num.to.keep = default.num.cores())
```

Arguments

`dl`	The DeLorean object
`num.to.keep`	The number to keep (defaults to default.num.cores())

`pseudotimes.pair.plot` *Plot two sets of pseudotimes against each other.*

Description

Plot two sets of pseudotimes against each other.

Usage

```
pseudotimes.pair.plot(dl, fits = NULL)
```

Arguments

`dl`	The DeLorean object
`fits`	Fit indexes

`report.file` *The filename of the R markdown report.*

Description

The filename of the R markdown report.

Usage

```
report.file(report.name)
```

Arguments

`report.name`	The report name

`Rhat.plot` *Plot the Rhat convergence statistics.* `examine.convergence` *must be called before this plot can be made.*

Description

Plot the Rhat convergence statistics. `examine.convergence` must be called before this plot can be made.

Usage

```
Rhat.plot(dl)
```

Arguments

`dl`	de.lorean object

roughness.of.permutations

Apply permutation based roughness test to held out genes

Description

Apply permutation based roughness test to held out genes

Usage

```
roughness.of.permutations(dl, expr.held.out = dl$expr.held.out,
  num.perms = 1000)
```

Arguments

dl	de.lorean object
expr.held.out	The expression matrix including the held out genes
num.perms	Number of permutations to test

roughness.of.sample

Calculate the roughness of the held out genes given the sample.

Description

Calculate the roughness of the held out genes given the sample.

Usage

```
roughness.of.sample(dl, expr.held.out = dl$expr.held.out,
  sample.iter = dl$best.sample)
```

Arguments

dl	de.lorean object
expr.held.out	The expression matrix including the held out genes
sample.iter	Which sample to use

`roughness.test` *Calculate roughnesses under fit samples and also under random permutations*

Description

Calculate roughnesses under fit samples and also under random permutations

Usage

```
roughness.test(dl, expr.held.out = dl$expr.held.out, num.perms = 1000)
```

Arguments

`dl`	de.lorean object
`expr.held.out`	The expression matrix including the held out genes
`num.perms`	Number of permutations to test

`roughnesses.plot` *Plot results of roughness test*

Description

Plot results of roughness test

Usage

```
roughnesses.plot(dl)
```

Arguments

`dl`	de.lorean object

seriation.find.orderings *Use seriation package to find good orderings*

Description

Use seriation package to find good orderings

Usage

```
seriation.find.orderings(dl, .methods = c("TSP", "R2E", "HC", "GW",
  "OLO"), scaled = c("scaled", "unscaled"), dim.red = c("none", "pca",
  "kfa", "ica", "mds"), dims = geom.series(base = 2, max = min(8,
  nrow(dl$expr) - 1)), num.cores = default.num.cores(),
  num.tau.to.keep = default.num.cores())
```

Arguments

dl	de.lorean object
.methods	The seriation methods to apply
scaled	Whether to use the scaled and/or unscaled expression data
dim.red	Dimension reduction methods to apply
dims	Number of dimensions to reduce to
num.cores	Number of cores to use in parallel
num.tau.to.keep	How many initialisations to keep

tau.offsets.plot *Plot the tau offsets, that is how much the pseudotimes (tau) differ from their prior means over the full posterior.*

Description

Plot the tau offsets, that is how much the pseudotimes (tau) differ from their prior means over the full posterior.

Usage

```
tau.offsets.plot(dl, rug.alpha = 0.3)
```

Arguments

dl	de.lorean object
rug.alpha	Alpha parameter for rug geom

test.fit *Test fit for log normal and gamma*

Description

Test fit for log normal and gamma

Usage

```
test.fit(vars)
```

Arguments

vars	Data to fit

test.mh *Test ordering Metropolis-Hastings sampler.*

Description

Test ordering Metropolis-Hastings sampler.

Usage

```
test.mh(dl, psi = mean(dl$gene.map$psi.hat),
  omega = mean(dl$gene.map$omega.hat),
  num.cores = getOption("DL.num.cores", max(parallel::detectCores() - 1,
  1)), iterations = 1000, thin = 15)
```

Arguments

dl	de.lorean object
psi	Temporal variation
omega	Noise
num.cores	Number of cores to run on. Defaults to getOption("DL.num.cores", max(parallel::detectCores()-1, 1))
iterations	Number of iterations
thin	Thin the samples

test.robustness.de.lorean

Test robustness of pseudotime estimation on subsets of de.lorean object

Description

Test robustness of pseudotime estimation on subsets of de.lorean object

Usage

```
test.robustness.de.lorean(dl, pieces = 2)
```

Arguments

dl	de.lorean object
pieces	How many pieces to partition into

windram.expr

Windram et al. investigated the defense response in Arabidopsis thaliana to the necrotrophic fungal pathogen Botrytis cinerea. They collected data at 24 time points in two conditions for 30336 genes.

Description

Windram et al. investigated the defense response in Arabidopsis thaliana to the necrotrophic fungal pathogen Botrytis cinerea. They collected data at 24 time points in two conditions for 30336 genes.

Usage

```
data(WindramDeLorean)
```

Format

There are three objects in this data:

- windram.expr A matrix of log expression values with no missing data. Rows are named by genes and columns are named by cells/samples.
- windram.gene.meta A data frame containing meta-data about the genes.
- windram.cell.meta A data frame containing meta-data about the cells

Source

http://www.plantcell.org/content/24/9/3530.long

Index