Package 'ImpulseDE2'

April 4, 2017

Type Package

Title Differential expression analysis of longitudinal count data sets

Version 0.99.5

Date 2017-03-10

Author David S Fischer [aut, cre], Fabian J Theis [ctb], Nir Yosef [ctb]

Maintainer David S Fischer <david.fischer@helmholtz-muenchen.de>

Description
ImpulseDE2 is a differential expression algorithm for longituinal count data sets which arise in sequencing experiments such as RNA-seq, ChIP-seq, ATAC-seq and DNaseI-seq.

License Artistic-2.0

Encoding UTF-8

Imports Biobase, BiocParallel, ComplexHeatmap, circlize, compiler, cowplot, DESeq2, ggplot2, grDevices, knitr, Matrix, methods, S4Vectors, stats, SummarizedExperiment

biocViews Software, StatisticalMethod, TimeCourse, Sequencing, DifferentialExpression, GeneExpression, CellBiology, CellBasedAssays

Collate 'ImpulseDE2_main.R' 'srcImpulseDE2_CostFunctionsFit.R' 'srcImpulseDE2_classImpulseDE2Object.R' 'srcImpulseDE2_computeNormConst.R' 'srcImpulseDE2_evalImpulse.R' 'srcImpulseDE2_evalSigmoid.R' 'srcImpulseDE2_fitImpulse.R' 'srcImpulseDE2_fitSigmoid.R' 'srcImpulseDE2_plotGenes.R' 'srcImpulseDE2_plotHeatmap.R' 'srcImpulseDE2_processData.R' 'srcImpulseDE2_runDEAnalysis.R' 'srcImpulseDE2_runDESeq2.R' 'srcImpulseDE2_simulateDataSet.R'

RoxygenNote 6.0.1

VignetteBuilder knitr

BugReports https://github.com/YosefLab/ImpulseDE2/issues

NeedsCompilation no

R topics documented:

`computeNormConst` *Compute a normalisation constant for each sample*

Description

The normalisation constant is the median of the ratio of gene counts versus the geomtric gene count mean. There is one normalisation constant per replicate. An intuitive alternative would be the sequencing depth, the median ratio is however less sensitive to highly differentially expressed genes with high counts (ref. DESeq). The normalisation constants are used to scale the mean of the negative binomial model inferred during fitting to the sequencing depth of the given sample. The normalisation constants therefore replace normalisation at the count data level, which is not supposed to be done in the framework of ImpulseDE2. There is the option to supply size factors to this function to override its size factor choice.

Usage

```
computeNormConst(matCountDataProc, vecSizeFactorsExternal = NULL)
```

Arguments

`matCountDataProc`
: (matrix genes x samples) Read count data.

`vecSizeFactorsExternal`
: (vector length number of cells in matCountData) [Default NULL] Externally generated list of size factors which override size factor computation in ImpulseDE2.

Value

vecSizeFactors (numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

Author(s)

David Sebastian Fischer

See Also

Called by runImpulseDE2. Calls computeSizeFactors.

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 100,
scaNImp          = 200,
scaNLin          = 100,
scaNSig          = 200)
vecSizeFactors <- computeNormConst(
matCountData = lsSimulatedData$matObservedCounts)
```

---

computeSizeFactors *Compute a size factor for each sample*

Description

This function computes size factors for each sample in the dataset and expands them to a matrix of the size of the dataset. Size factors scale the negative binomial likelihood model of a gene to the sequencing depth of each sample. Note that size factors on bulk and single-cell data are computed differently: Median ratio of data to geometric mean for bul data and normalised relative sequencing depth for single-cell data.

Usage

```
computeSizeFactors(matCountDataProc)
```

Arguments

matCountDataProc	(matrix genes x samples) Read count data.

Value

vecSizeFactors (numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

Author(s)

David Sebastian Fischer

See Also

Called by computeNormConst.

---

estimateImpulseParam *Compute peak and valley impulse model parameter initialisations for data of one gene*

Description

[Model fitting function hierarchy: helper to level 3 out of 4] This is a fitting helper function which computes parameter intialisations and does not wrap or execute numerical optimisation. The peak model models a maximum between start and end time, the valley model models a minimum between start and end time.

Usage

```
estimateImpulseParam(vecCounts, vecTimepoints, vecSizeFactors, lsvecidxBatch)
```

Arguments

vecCounts	(numeric vector number of samples) Read count data.
vecTimepoints	(numeric vector length number of samples) Time coordinates of each sample.
vecSizeFactors	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
lsvecidxBatch	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.

Value

(list length 2)

- peak (numeric vector length 6) {beta, h0, h1, h2, t1, t2} Peak model initialisations of impulse model parameters.
- valley (numeric vector length 6) {beta, h0, h1, h2, t1, t2} Valley model initialisations of impulse model parameters.

Author(s)

David Sebastian Fischer

See Also

Called by fitConstImpulseGene.

`estimateSigmoidParam` *Compute up and down sigmoid model parameter initialisations for data of one gene*

Description

[Model fitting function hierarchy: helper to level 2 out of 3] This is a fitting helper function which computes parameter intialisations and does not wrap or execute numerical optimisation. The up model models a sigmoidal expression increase over time, the down model a sigmoidal decrease over time.

Usage

```
estimateSigmoidParam(vecCounts, vecTimepoints, vecSizeFactors, lsvecidxBatch)
```

Arguments

`vecCounts`	(numeric vector number of samples) Read count data.
`vecTimepoints`	(numeric vector length number of samples) Time coordinates of each sample.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`lsvecidxBatch`	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.

Value

(list length 2)

- peak (numeric vector length 6) {beta, h0, h1, t} Up model initialisations of sigmoidal model parameters.
- valley (numeric vector length 6) {beta, h0, h1, t} Down model initialisations of sigmoidal model parameters.

Author(s)

David Sebastian Fischer

See Also

Called by fitSigmoidGene.

`evalImpulse` *Compute value of impulse function given parameters.*

Description

Compute value of impulse function given parameters. Enforces lower bound on value of function to avoid numerical errors during model fitting.

Usage

```
evalImpulse(vecImpulseParam, vecTimepoints)
```

Arguments

`vecImpulseParam`	(numeric vector number of impulse model parameters) {beta, h0, h1, h2, t1, t2} Vector of impulse model parameters.
`vecTimepoints`	(numeric vector length number of time points) Time points to be evaluated.

Value

vecImpulseValue (vec number of vecTimepoints) Model values for given time points.

Author(s)

David Sebastian Fischer

See Also

Compiled version: evalImpulse_comp

evalImpulse_comp *Compiled function: evalImpulse*

Description

Pre-compile heavily used functions. Refer to evalImpulse.

Usage

```
evalImpulse_comp(vecImpulseParam, vecTimepoints)
```

Arguments

vecImpulseParam	(numeric vector number of impulse model parameters) {beta, h0, h1, h2, t1, t2} Vector of impulse model parameters.
vecTimepoints	(numeric vector length number of time points) Time points to be evaluated.

Value

vecImpulseValue (vec number of vecTimepoints) Model values for given time points.

Author(s)

David Sebastian Fischer

evalLogLikImpulse *Cost function for impulse model*

Description

Log likelihood cost function for numerical optimisation of impulse model. Implements log linker function for the amplitude parameters and the batch correction factors. Implements upper and lower sensitivity bound of likelihood with respect to batch correction factors and lower bound for amplitude paramters.

Usage

```
evalLogLikImpulse(vecTheta, vecCounts, scaDisp, vecSizeFactors,
  vecTimepointsUnique, vecidxTimepoint, lsvecidxBatch, vecboolObserved)
```

Arguments

vecTheta	(numeric vector number of parameters to be estimated) Impulse model parameter and batch correction factor estimates.
vecCounts	(numeric vector number of samples) Read count data.
scaDisp	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
vecSizeFactors	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

`vecTimepointsUnique`
: (numeric vector length number of unique time points) Unique time points of set of time points of given samples.

`vecidxTimepoint`
: (index vector length number of samples) Index of of time point assigned to each sample in vector vecTimepointsUnique.

`lsvecidxBatch`
: (list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.

`vecboolObserved`
: (bool vector number of samples) Whether sample is observed (finite and not NA).

Value

scaLogLik (scalar) Value of cost function (loglikelihood) for given gene.

Author(s)

David Sebastian Fischer

See Also

Compiled version: evalLogLikImpulse_comp

`evalLogLikImpulse_comp`
Compiled function: evalLogLikImpulse

Description

Pre-compile heavily used functions. Refer to evalLogLikImpulse.

Usage

```
evalLogLikImpulse_comp(vecTheta, vecCounts, scaDisp, vecSizeFactors,
  vecTimepointsUnique, vecidxTimepoint, lsvecidxBatch, vecboolObserved)
```

Arguments

`vecTheta`
: (numeric vector number of parameters to be estimated) Impulse model parameter and batch correction factor estimates.

`vecCounts`
: (numeric vector number of samples) Read count data.

`scaDisp`
: (scalar) Gene-wise negative binomial dispersion hyper-parameter.

`vecSizeFactors`
: (numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

`vecTimepointsUnique`
: (numeric vector length number of unique time points) Unique time points of set of time points of given samples.

vecidxTimepoint
: (index vector length number of samples) Index of of time point assigned to each sample in vector vecTimepointsUnique.

lsvecidxBatch
: (list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.

vecboolObserved
: (bool vector number of samples) Whether sample is observed (finite and not NA).

Value

scaLogLik (scalar) Value of cost function (loglikelihood) for given gene.

Author(s)

David Sebastian Fischer

---

evalLogLikMu *Cost function for constant model*

Description

Log likelihood cost function for numerical optimisation of constant model. Implements log linker function for the constant mean parameter and the batch correction factors. Implements lower sensitivity bound of likelihood with respect to constant mean parameter. Implements upper and lower sensitivity bound of likelihood with respect to batch correction factors.

Usage

```
evalLogLikMu(vecTheta, vecCounts, scaDisp, vecSizeFactors, lsvecidxBatch,
  vecboolObserved)
```

Arguments

vecTheta
: (numeric vector number of parameters to be estimated) Constant model parameter and batch correction factor estimates.

vecCounts
: (numeric vector number of samples) Read count data.

scaDisp
: (scalar) Gene-wise negative binomial dispersion hyper-parameter.

vecSizeFactors
: (numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

lsvecidxBatch
: (list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.

vecboolObserved
: (bool vector number of samples) Whether sample is observed (finite and not NA).

Value

scaLogLik (scalar) Value of cost function (loglikelihood) for given gene.

Author(s)

David Sebastian Fischer

See Also

Compiled version: evalLogLikMu_comp

`evalLogLikMu_comp` *Compiled function: evalLogLikMu*

Description

Pre-compile heavily used functions. Refer to evalLogLikMu.

Usage

```
evalLogLikMu_comp(vecTheta, vecCounts, scaDisp, vecSizeFactors, lsvecidxBatch,
  vecboolObserved)
```

Arguments

`vecTheta`	(numeric vector number of parameters to be estimated) Constant model parameter and batch correction factor estimates.
`vecCounts`	(numeric vector number of samples) Read count data.
`scaDisp`	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`lsvecidxBatch`	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.
`vecboolObserved`	(bool vector number of samples) Whether sample is observed (finite and not NA).

Value

scaLogLik (scalar) Value of cost function (loglikelihood) for given gene.

Author(s)

David Sebastian Fischer

`evalLogLikSigmoid` *Cost function for sigmoidal model*

Description

Log likelihood cost function for numerical optimisation of sigmoidal model. Implements log linker function for the amplitude parameters and the batch correction factors. Implements upper and lower sensitivity bound of likelihood with respect to batch correction factors and lower bound for amplitude paramters.

Usage

```
evalLogLikSigmoid(vecTheta, vecCounts, scaDisp, vecSizeFactors,
  vecTimepointsUnique, vecidxTimepoint, lsvecidxBatch, vecboolObserved)
```

Arguments

`vecTheta`	(numeric vector number of parameters to be estimated) Sigmoid model parameter and batch correction factor estimates.
`vecCounts`	(numeric vector number of samples) Read count data.
`scaDisp`	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`vecTimepointsUnique`	(numeric vector length number of unique time points) Unique time points of set of time points of given samples.
`vecidxTimepoint`	(index vector length number of samples) Index of of time point assigned to each sample in vector vecTimepointsUnique.
`lsvecidxBatch`	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.
`vecboolObserved`	(bool vector number of samples) Whether sample is observed (finite and not NA).

Value

scaLogLik (scalar) Value of cost function (loglikelihood) for given gene.

Author(s)

David Sebastian Fischer

See Also

Compiled version: evalLogLikSigmoid_comp

`evalLogLikSigmoid_comp` *Compiled function: evalLogLikSigmoid*

Description

Pre-compile heavily used functions. Refer to evalLogLikSigmoid.

Usage

```
evalLogLikSigmoid_comp(vecTheta, vecCounts, scaDisp, vecSizeFactors,
  vecTimepointsUnique, vecidxTimepoint, lsvecidxBatch, vecboolObserved)
```

Arguments

`vecTheta`	(numeric vector number of parameters to be estimated) Sigmoid model parameter and batch correction factor estimates.
`vecCounts`	(numeric vector number of samples) Read count data.
`scaDisp`	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`vecTimepointsUnique`	(numeric vector length number of unique time points) Unique time points of set of time points of given samples.
`vecidxTimepoint`	(index vector length number of samples) Index of of time point assigned to each sample in vector vecTimepointsUnique.
`lsvecidxBatch`	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.
`vecboolObserved`	(bool vector number of samples) Whether sample is observed (finite and not NA).

Value

scaLogLik (scalar) Value of cost function (loglikelihood) for given gene.

Author(s)

David Sebastian Fischer

evalSigmoid *Compute value of sigmoidal model given parameters.*

Description

Compute value of sigmoidal model given parameters. Enforces lower bound on value of function to avoid numerical errors during model fitting.

Usage

```
evalSigmoid(vecSigmoidParam, vecTimepoints)
```

Arguments

vecSigmoidParam	(numeric vector number of sigmoid model parameters) {beta, h0, h1, t1} Vector of sigmoidal model parameters.
vecTimepoints	(numeric vector length number of time points) Time points to be evaluated.

Value

vecSigmoidValue (numeric vector length of vecTimepoints) Model values for given time points.

Author(s)

David Sebastian Fischer

See Also

Compiled version: evalSigmoid_comp

evalSigmoid_comp *Compiled function: evalSigmoid*

Description

Pre-compile heavily used functions. Refer to evalSigmoid.

Usage

```
evalSigmoid_comp(vecSigmoidParam, vecTimepoints)
```

Arguments

vecSigmoidParam	(numeric vector number of sigmoid model parameters) {beta, h0, h1, t1} Vector of sigmoidal model parameters.
vecTimepoints	(numeric vector length number of time points) Time points to be evaluated.

Value

vecSigmoidValue (numeric vector length of vecTimepoints) Model values for given time points.

Author(s)

David Sebastian Fischer

`fitConstImpulse` *Fits impulse and constant models to all genes on all samples of a condition*

Description

[Model fitting function hierarchy: 2 out of 4] This secondary fitting wrapper performs parralelisation of model fitting across genes.

Usage

```
fitConstImpulse(matCountDataProcCondition, vecDispersions, vecSizeFactors,
  vecTimepoints, lsvecBatches, boolFitConst)
```

Arguments

`matCountDataProcCondition`	(matrix genes x samples) Read count data.
`vecDispersions`	(vector number of genes) Gene-wise negative binomial dispersion hyper-parameters.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`vecTimepoints`	(numeric vector length number of samples) Time coordinates of each sample.
`lsvecBatches`	(list length number of confounding variables) List of vectors. One vector per confounding variable. Each vector has one entry per sample with the name of the batch ID within the given confounding variable of the given sample.
`boolFitConst`	(bool) Whether to fit a constant model.

Value

(list length 5)

- lsFits (list of lists length number of genes) List of model fits for each gene. Each gene entry is a list of model fits to the individual models: Impulse model and constant model (if boolFitConst is TRUE). At this level, the sigmoid model fit can be added later. Each model fit per gene is a list of fitting parameters and results.
  - Gene ID (list length 2) Impulse and constant model fit to gene observations. One entry of this format for all gene IDs.
    - lsImpulseFit (list) List of impulse fit parameters and results.
      - vecImpulseParam (numeric vector length 6) {beta, h0, h1, h2, t1, t2} Maximum likelihood estimators of impulse model parameters.
      - vecImpulseValue (numeric vector length number of time points) Values of impulse model fit at time points used for fit.

        - lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
        - scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
        - scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
        - scaConvergence (scalar) Convergence status of optim on impulse model.
    * lsConstFit (list) List of constant fit parameters and results.
        - scaMu (scalar) Maximum likelihood estimator of negative binomial mean parameter.
        - lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
        - scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
        - scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
        - scaConvergence (scalar) Convergence status of optim on constant model.
  - vecTimepointsUnique (numeric vector length number of unique timepoints) Vector of unique time coordinates observed in this condition.
  - vecidxTimepoint (idx vector length number of samples) Index of the time coordinates of each sample (reference is vecTimepointsUnique).
  - lsvecBatchUnique (list number of confounders) List of string vectors. One vector per confounder: vector of unique batches in this confounder.
  - lsvecidxBatches (idx list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index of the batch ID within the given confounding variable of the given sample. Reference is the list of unique batch ids for each confounding variable.

Author(s)

David Sebastian Fischer

See Also

Called by fitModels to fit constant and impulse model to samples of one condition. Calls fitConstImpulseGene to perform fitting on each gene.

`fitConstImpulseGene` *Fit an impulse and constant model to a single gene*

Description

[Model fitting function hierarchy: 3 out of 4] This tertiary fitting wrapper calls the optimisation wrappers for the individual fitting operations to be performed on the observations of this gene. Structure of this function:

- Fit impulse model
  - Initialise impulse model parameters (peak and valley)
  - Fit impulse model (peak initialisation)
  - Fit impulse model (valley initialisation)
- Select best impulse model fit from initialisations,
- Fit constant model (if constant model is to be fit).

Usage

```
fitConstImpulseGene(vecCounts, scaDisp, vecSizeFactors, vecTimepointsUnique,
  vecidxTimepoint, lsvecidxBatch, boolFitConst, MAXIT = 1000)
```

Arguments

vecCounts	(numeric vector number of samples) Read count data.
scaDisp	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
vecSizeFactors	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
vecTimepointsUnique	(numeric vector length number of unique timepoints) Vector of unique time coordinates observed in this condition.
vecidxTimepoint	(numeric vector length number of samples) Index of the time coordinates of each sample (reference is vecTimepointsUnique).
lsvecidxBatch	(idx list length number of confounding variables) List of vectors. One vector per confounding variable. Each vector has one entry per sample with the index of the batch ID within the given confounding variable of the given sample. Reference is the list of unique batch ids for each confounding variable.
boolFitConst	(bool) Whether to fit a constant model.
MAXIT	(scalar) [Default 1000] Maximum number of BFGS iterations for model fitting with optim.

Value

(list length 2) Impulse and constant model fit to gene observations.

- lsImpulseFit (list) List of impulse fit parameters and results.
  - vecImpulseParam (numeric vector length 6) {beta, h0, h1, h2, t1, t2} Maximum likelihood estimators of impulse model parameters.
  - vecImpulseValue (numeric vector length number of time points) Values of impulse model fit at time points used for fit.
  - lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
  - scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
  - scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
  - scaConvergence (scalar) Convergence status of optim on impulse model.
- lsConstFit (list) List of constant fit parameters and results.
  - scaMu (scalar) Maximum likelihood estimator of negative binomial mean parameter.
  - lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
  - scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
  - scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
  - scaConvergence (scalar) Convergence status of optim on constant model.

Author(s)

David Sebastian Fischer

See Also

Called by fitConstImpulseGene to fit constant and impulse model to samples of one condition and one gene. Calls impulse parameter initialisation function estimateImpulseParam and optimisation wrappers fitImpulseModel and fitConstModel.

---

fitConstModel *Fit a constant model to data of a gene*

Description

[Model fitting function hierarchy: 4 out of 4] This quarterny fitting wrapper performs constant model fitting: This function executes numerical optimisaiton and error-handling thereof.

Usage

```
fitConstModel(vecCounts, scaDisp, vecSizeFactors, lsvecidxBatch, MAXIT = 1000,
  RELTOL = 10^(-8), trace = 0, REPORT = 10)
```

Arguments

vecCounts	(numeric vector number of samples) Read count data.
scaDisp	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
vecSizeFactors	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
lsvecidxBatch	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.
MAXIT	(scalar) [Default 1000] Maximum number of BFGS iterations for model fitting with optim.
RELTOL	(scalar) [Default 10^(-8)] Maximum relative change in loglikelihood to reach convergence in numerical optimisation by BFGS in optim.
trace	(scalar) [Defaul 0] Reporting parameter of optim.
REPORT	(scalar) [Default 10] Reporting parameter of optim.

Value

(list) List of constant fit parameters and results.

- scaMu (scalar) Maximum likelihood estimator of negative binomial mean parameter.
- lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
- scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
- scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
- scaConvergence (scalar) Convergence status of optim on constant model.

Author(s)

David Sebastian Fischer

See Also

Called by fitConstImpulseGene to fit constant model to samples of one condition and one gene. Calls constant model cost function evalLogLikMu within optim.

`fitImpulseModel` *Fit an impulse model to data of a gene*

Description

[Model fitting function hierarchy: 4 out of 4] This quarterny fitting wrapper performs impulse model fitting: This function executes numerical optimisaiton and error-handling thereof.

Usage

```
fitImpulseModel(vecImpulseParamGuess, vecCounts, scaDisp, vecSizeFactors,
  lsvecidxBatch, vecTimepointsUnique, vecidxTimepoint, MAXIT = 1000,
  RELTOL = 10^(-8), trace = 0, REPORT = 10)
```

Arguments

`vecImpulseParamGuess`	(numeric vector length 6) {beta, h0, h1, h2, t1, t2} Initialisations of impulse model parameters.
`vecCounts`	(numeric vector number of samples) Read count data.
`scaDisp`	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`lsvecidxBatch`	(list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.
`vecTimepointsUnique`	(numeric vector length number of unique time points) Unique time points of set of time points of given samples.
`vecidxTimepoint`	(index vector length number of samples) Index of of time point assigned to each sample in vector vecTimepointsUnique.
`MAXIT`	(scalar) [Default 1000] Maximum number of BFGS iterations for model fitting with optim.
`RELTOL`	(scalar) [Default 10^(-8)] Maximum relative change in loglikelihood to reach convergence in numerical optimisation by BFGS in optim.
`trace`	(scalar) [Defaul 0] Reporting parameter of optim.
`REPORT`	(scalar) [Default 10] Reporting parameter of optim.

Value

(list) List of impulse fit parameters and results.

- vecImpulseParam (numeric vector length 6) {beta, h0, h1, h2, t1, t2} Maximum likelihood estimators of impulse model parameters.
- vecImpulseValue (numeric vector length number of time points) Values of impulse model fit at time points used for fit.
- lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
- scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
- scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
- scaConvergence (scalar) Convergence status of optim on impulse model.

Author(s)

David Sebastian Fischer

See Also

Called by fitConstImpulseGene to fit impulse model to samples of one condition and one gene. Calls impulse model cost function evalLogLikImpulse_comp within optim.

`fitModels` *Fits impulse and constant models to a timecourse dataset*

Description

[Model fitting function hierarchy: 1 out of 4] This primary wrapper coordinates fitting of impulse and constant model to separate conditions according to the differential expression mode (case-only or case-control).

Usage

```
fitModels(objectImpulseDE2, vecConfounders, boolCaseCtrl)
```

Arguments

`objectImpulseDE2`	(object class ImpulseDE2Object) Object to be fit.
`vecConfounders`	(vector of strings number of confounding variables) Factors to correct for during batch correction. Names refer to columns in dfAnnotation.
`boolCaseCtrl`	(bool) Whether to perform case-control analysis. Does case-only analysis if FALSE.

Value

objectImpulseDE2 (object class ImpulseDE2Object) Object with sigmoidal fit added: objectImpulseDE2@lsModelFits is updated to: lsModelFits (list length number of conditions fit (1 or 3) +1) {"case"} or {"case", "control", "combined"} One model fitting object for each condition: In case-only DE analysis, only the condition {"case"} is fit. In case-control DE analysis, the conditions {"case", "control","combined} are fit. Each condition entry is a list of model fits for each gene. Each gene entry is a list of model fits to the individual models: Impulse model and constant model (if boolFitConst is TRUE). At this level, the sigmoid model fit can be added later. Each model fit per gene is a list of fitting parameters and results.

- IdxGroups (list length number of conditions) Samples grouped by time points and by batches and time point vectors. Sample groups are stored in the form of index vectors in which samples of the same time point or batch have the same index.
  - Condition ID (list length 5) List of index vectors and time points. One entry of this format for each condition.
    * vecTimepointsUnique (numeric vector length number of unique timepoints) Vector of unique time coordinates observed in this condition.
    * vecidxTimepoint (idx vector length number of samples) Index of the time coordinates of each sample (reference is vecTimepointsUnique).
    * lsvecBatchUnique (list number of confounders) List of string vectors. One vector per confounder: vector of unique batches in this confounder.
    * lsvecidxBatches (idx list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index of the batch ID within the given confounding variable of the given sample. Reference is the list of unique batch ids for each confounding variable.
    * vecSamples (vector number of samples) Names of samples fit for this condition in same order as index vectors above.
- Condition ID (list length number of genes) List of fits for each gene to the samples of this condition. One entry of this format for all conditions fit.
  - Gene ID (list length 2) Impulse and constant model fit to gene observations. One entry of this format for all gene IDs.
    * lsImpulseFit (list) List of impulse fit parameters and results.
      · vecImpulseParam (numeric vector length 6) {beta, h0, h1, h2, t1, t2} Maximum likelihood estimators of impulse model parameters.
      · vecImpulseValue (numeric vector length number of time points) Values of impulse model fit at time points used for fit.
      · lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
      · scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
      · scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
      · scaConvergence (scalar) Convergence status of optim on impulse model.
    * lsConstFit (list) List of constant fit parameters and results.
      · scaMu (scalar) Maximum likelihood estimator of negative binomial mean parameter.
      · lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
      · scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).

- scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
- scaConvergence (scalar) Convergence status of optim on constant model.

Author(s)

David Sebastian Fischer

See Also

Calls fitConstImpulse once for each condition with the appropriate parameters and samples.

`fitSigmoidGene` *Fit a sigmoidal model to a single gene*

Description

[Model fitting function hierarchy: 2 out of 3] This secondary fitting wrapper calls the optimisation wrappers for the individual fitting operations to be performed on the observations of this gene. Structure of this function:

- Fit sigmoidal model
  - Initialise sigmoidal model parameters (up and down)
  - Fit sigmoidal model (up initialisation)
  - Fit sigmoidal model (down initialisation)
- Select best sigmoidal model fit from initialisations,

Usage

```
fitSigmoidGene(vecCounts, scaDisp, vecSizeFactors, vecTimepointsUnique,
  vecidxTimepoint, lsvecidxBatch, MAXIT = 1000)
```

Arguments

`vecCounts`	(numeric vector number of samples) Read count data.
`scaDisp`	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
`vecTimepointsUnique`	(numeric vector length number of unique timepoints) Vector of unique time coordinates observed in this condition.
`vecidxTimepoint`	(idx vector length number of samples) Index of the time coordinates of each sample (reference is vecTimepointsUnique).
`lsvecidxBatch`	(idx list length number of confounding variables) List of vectors. One vector per confounding variable. Each vector has one entry per sample with the index of the batch ID within the given confounding variable of the given sample. Reference is the list of unique batch ids for each confounding variable.
`MAXIT`	(scalar) [Default 1000] Maximum number of BFGS iterations for model fitting with optim.

Value

(list) List of sigmoidal fit parameters and results.

- vecSigmoidParam (numeric vector length 4) {beta, h0, h1, t} Maximum likelihood estimators of sigmoidal model parameters.
- vecSigmoidValue (numeric vector length number of time points) Values of sigmoid model fit at time points used for fit.
- lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
- scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
- scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
- scaConvergence (scalar) Convergence status of optim on sigmoidal model.

Author(s)

David Sebastian Fischer

See Also

Called by fitSigmoidModels to fit sigmoidal model to samples of one condition and one gene. Calls sigmoidal parameter initialisation function estimateSigmoidParam and optimisation wrapper fitSigmoidModel.

`fitSigmoidModel` *Fit a sigmoidal model to data of a gene*

Description

[Model fitting function hierarchy: 3 out of 3] This tertiary fitting wrapper performs sigmoidal model fitting: This function executes numerical optimisaiton and error-handling thereof.

Usage

```
fitSigmoidModel(vecSigmoidParamGuess, vecCounts, scaDisp, vecSizeFactors,
  lsvecidxBatch, vecTimepointsUnique, vecidxTimepoint, MAXIT = 100,
  RELTOL = 10^(-8), trace = 0, REPORT = 10)
```

Arguments

`vecSigmoidParamGuess`	(numeric vector length 4) {beta, h0, h1, t} Up model initialisations of sigmoidal model parameters.
`vecCounts`	(numeric vector number of samples) Read count data.
`scaDisp`	(scalar) Gene-wise negative binomial dispersion hyper-parameter.
`vecSizeFactors`	(numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

`lsvecidxBatch` (list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index batch within the given confounding variable of the given sample. Batches are enumerated from 1 to number of batches.

`vecTimepointsUnique` (numeric vector length number of unique time points) Unique time points of set of time points of given samples.

`vecidxTimepoint` (index vector length number of samples) Index of of time point assigned to each sample in vector vecTimepointsUnique.

`MAXIT` (scalar) [Default 1000] Maximum number of BFGS iterations for model fitting with optim.

`RELTOL` (scalar) [Default 10^(-8)] Maximum relative change in loglikelihood to reach convergence in numerical optimisation by BFGS in optim.

`trace` (scalar) [Defaul 0] Reporting parameter of optim.

`REPORT` (scalar) [Default 10] Reporting parameter of optim.

Value

(list) List of sigmoid fit parameters and results.

- vecSigmoidParam (numeric vector length 4) {beta, h0, h1, t} Maximum likelihood estimators of sigmoidal model parameters.
- vecSigmoidValue (numeric vector length number of time points) Values of sigmoid model fit at time points used for fit.
- lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
- scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
- scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
- scaConvergence (scalar) Convergence status of optim for sigmoid model.

Author(s)

David Sebastian Fischer

See Also

Called by fitSigmoidGene to fit sigmoidal model to samples of one condition and one gene. Calls sigmoidal model cost function evalLogLikSigmoid_comp within optim.

`fitSigmoidModels` *Fits sigmoidal models to all genes on all all samples of a condition*

Description

[Model fitting function hierarchy: 1 out of 3] This primary fitting wrapper performs parralelisation of model fitting across genes.

Usage

```
fitSigmoidModels(objectImpulseDE2, vecConfounders, strCondition)
```

Arguments

`objectImpulseDE2`	(object class ImpulseDE2Object) Object to be fit with sigmoidal model. Needs to be fitted with impulse model before.
`vecConfounders`	(vector of strings number of confounding variables) Factors to correct for during batch correction. Names refer to columns in dfAnnotation.
`strCondition`	(str) Name of condition entry in lsModelFits for which sigmoidal models are to be fit to each gene.

Value

objectImpulseDE2 (object class ImpulseDE2Object) Object with sigmoidal fit added: objectImpulseDE2@lsModelFits is updated to: lsModelFits (list length number of conditions fit (1 or 3) +1) {"case"} or {"case", "control", "combined"} This is the lsModelFits object handed to this function with additional sigmoid fit entries for every gene for the given condition. One model fitting object for each condition: In case-only DE analysis, only the condition {"case"} is fit. In case-control DE analysis, the conditions {"case", "control","combined} are fit. Each condition entry is a list of model fits for each gene. Each gene entry is a list of model fits to the individual models: Impulse model, constant model and sigmoidal fit. Each model fit per gene is a list of fitting parameters and results.

- IdxGroups (list length number of conditions) Samples grouped by time points and by batches and time point vectors. Sample groups are stored in the form of index vectors in which samples of the same time point or batch have the same index.
  - Condition ID (list length 5) List of index vectors and time points. One entry of this format for each condition.
    - vecTimepointsUnique (numeric vector length number of unique timepoints) Vector of unique time coordinates observed in this condition.
    - vecidxTimepoint (idx vector length number of samples) Index of the time coordinates of each sample (reference is vecTimepointsUnique).
    - lsvecBatchUnique (list number of confounders) List of string vectors. One vector per confounder: vector of unique batches in this confounder.
    - lsvecidxBatches (idx list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index of the batch ID within the given confounding variable of the given sample. Reference is the list of unique batch ids for each confounding variable.

    * vecSamples (vector number of samples) Names of samples fit for this condition in same order as index vectors above.

- Condition ID (list length number of genes) List of fits for each gene to the samples of this condition. One entry of this format for all conditions fit.
  - Gene ID (list length 2) Impulse, constant and sigmoidal model fit to gene observations. One entry of this format for all gene IDs.
    * lsImpulseFit (list) List of impulse fit parameters and results. For details, read the annotation of fitModels.
    * lsConstFit (list) List of constant fit parameters and results. For details, read the annotation of fitModels.
    * ls SigmoidFit (list) List of sigmoidal fit parameters and results.
      · vecSigmoidParam (numeric vector length 4) {beta, h0, h1, t} Maximum likelihood estimators of sigmoidal model parameters.
      · vecSigmoidValue (numeric vector length number of time points) Values of sigmoid model fit at time points used for fit.
      · lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
      · scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyper-parameter).
      · scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
      · scaConvergence (scalar) Convergence status of optim on sigmoidal model.

Author(s)

David Sebastian Fischer

See Also

Calls fitSigmoidGene to perform fitting on each gene.

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 0,
scaNImp          = 20,
scaNLin          = 10,
scaNSig          = 20)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
boolIdentifyTransients = FALSE,
scaNProc        = 1 )
# You could have used boolIdentifyTransients=TRUE
# to avoid the following post wrapper fitting.
objectImpulseDE2 <- fitSigmoidModels(
objectImpulseDE2 = objectImpulseDE2,
vecConfounders   = NULL,
```

```
strCondition     = "case")
objectImpulseDE2 <- updateDEAnalysis(
objectImpulseDE2=objectImpulseDE2,
scaQThresTransients=0.001)
head(objectImpulseDE2$dfImpulseDE2Results)
# dfImpulseDE2Results now contain 'transients-analysis'.
```

`generics_get_accessors` *ImpulseDE2Object accessor method generics*

Description

Generics for methods which operate on ImpulseDE2Object.

Usage

```
get_lsModelFits(object)

get_matCountDataProc(object)

get_dfAnnotationProc(object)

get_vecSizeFactors(object)

get_vecDispersions(object)

get_boolCaseCtrl(object)

get_vecConfounders(object)

get_scaNProc(object)

get_scaQThres(object)

get_strReport(object)
```

Arguments

`object`	(object) Object from which to retrieve data.

Value

(list) lsModelFits

(numeric matrix size genes x samples) matCountDataProc

(data frame size genes x reported characteristics) dfAnnotationProc

(numeric vector length number of samples) vecSizeFactors

(numeric vector length number of genes) vecDispersions

(bool) boolCaseCtrl

(str vector) vecConfounders

(scalar) scaNProc

(scalar) scaQThres

(str) strReport

`get_accessors` *ImpulseDE2Object accession methods*

Description

Get internal data of ImpulseDE2 output object.

Usage

```
## S4 method for signature 'ImpulseDE2Object'
get_lsModelFits(object)

## S4 method for signature 'ImpulseDE2Object'
get_matCountDataProc(object)

## S4 method for signature 'ImpulseDE2Object'
get_dfAnnotationProc(object)

## S4 method for signature 'ImpulseDE2Object'
get_vecSizeFactors(object)

## S4 method for signature 'ImpulseDE2Object'
get_vecDispersions(object)

## S4 method for signature 'ImpulseDE2Object'
get_boolCaseCtrl(object)

## S4 method for signature 'ImpulseDE2Object'
get_vecConfounders(object)

## S4 method for signature 'ImpulseDE2Object'
get_scaNProc(object)

## S4 method for signature 'ImpulseDE2Object'
get_scaQThres(object)

## S4 method for signature 'ImpulseDE2Object'
get_strReport(object)
```

Arguments

`object`	(objectImpulseDE2) A ImpulseDE2 output object.

Value

The internal data object specified by the function.

Author(s)

David Sebastian Fischer

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 30,
scaNImp          = 10,
scaNLin          = 10,
scaNSig          = 10)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
scaNProc        = 1 )
# Extract hidden auxillary result and processed input objects.
lsModelFits <- get_lsModelFits(objectImpulseDE2)
matCountDataProc <- get_matCountDataProc(objectImpulseDE2)
dfAnnotationProc <- get_dfAnnotationProc(objectImpulseDE2)
vecSizeFactors <- get_vecSizeFactors(objectImpulseDE2)
vecDispersions <- get_vecDispersions(objectImpulseDE2)
boolCaseCtrl <- get_boolCaseCtrl(objectImpulseDE2)
vecConfounders <- get_vecConfounders(objectImpulseDE2)
scaNProc <- get_scaNProc(objectImpulseDE2)
scaQThres <- get_scaQThres(objectImpulseDE2)
strReport <- get_strReport(objectImpulseDE2)
```

ImpulseDE2Object-class

Container class for ImpulseDE2 output

Description

ImpulseDE2 output and intermediate results such as model fits.

Slots

`dfDEAnalysis` (data frame samples x reported characteristics) Summary of fitting procedure and differential expression results for each gene.

- Gene: Gene ID.
- p: P-value for differential expression.
- padj: Benjamini-Hochberg false-discovery rate corrected p-value for differential expression analysis.
- loglik_full: Loglikelihood of full model.
- loglik_red: Loglikelihood of reduced model.

- df_full: Degrees of freedom of full model.
- df_red: Degrees of freedom of reduced model
- mean: Inferred mean parameter of constant model of first batch. From combined samples in case-ctrl.
- allZero (bool) Whether there were no observed non-zero observations of this gene. If TRUE, fitting and DE analsysis were skipped and entry is NA.

Entries only present in case-only DE analysis:

- converge_impulse: Convergence status of optim for impulse model fit (full model).
- converge_const: Convergence status of optim for constant model fit (reduced model).

Entries only present in case-control DE analysis:

- converge_combined: Convergence status of optim for impulse model fit to case and control samples combined (reduced model).
- converge_case: Convergence status of optim for impulse model fit to samples of case condition (full model 1/2).
- converge_control: Convergence status of optim for impulse model fit to samples of control condition (full model 2/2).

Entries only present if boolIdentifyTransients is TRUE:

- converge_sigmoid: Convergence status of optim for sigmoid model fit to samples of case condition.
- impulseTOsigmoid_p: P-value of loglikelihood ratio test impulse model fit versus sigmoidal model on samples of case condition.
- impulseTOsigmoid_padj: Benjamini-Hochberg false-discovery rate corrected p-value of loglikelihood ratio test impulse model fit versus sigmoid model on samples of case condition.
- sigmoidTOconst_p: P-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- sigmoidTOconst_padj: Benjamini-Hochberg false-discovery rate corrected p-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- isTransient (bool) Whether gene is transiently activated or deactivated and differentially expressed.
- isMonotonous (bool) Whether gene is not transiently activated or deactivated and differentially expressed. This scenario corresponds to a montonous expression level increase or decrease.

`vecDEGenes` (list number of genes) Genes IDs identified as differentially expressed by ImpulseDE2 at threshold `scaQThres`.

`lsModelFits` (list length number of conditions fit (1 or 3)) "case" or "case", "control", "combined" One model fitting object for each condition: In case-only DE analysis, only the condition "case" is fit. In case-control DE analysis, the conditions "case", "control","combined are fit. Each condition entry is a list of model fits for each gene. Each gene entry is a list of model fits to the individual models: Impulse model and constant model (if boolFitConst is TRUE). At this level, the sigmoid model fit can be added later. Each model fit per gene is a list of fitting parameters and results.

- IdxGroups (list length number of conditions) Samples grouped by time points and by batches and time point vectors. Sample groups are stored in the form of index vectors in which samples of the same time point or batch have the same index.
  - Condition ID (list length 3) List of index vectors and time points. One entry of this format for each condition.

    * vecTimepointsUnique (numeric vector length number of unique timepoints) Vector of unique time coordinates observed in this condition.
    * vecidxTimepoint (idx vector length number of samples) Index of the time coordinates of each sample (reference is vecTimepointsUnique).
    * lsvecBatchUnique (list number of confounders) List of string vectors. One vector per confounder: vector of unique batches in this confounder.
    * lsvecidxBatches (idx list length number of confounding variables) List of index vectors. One vector per confounding variable. Each vector has one entry per sample with the index of the batch ID within the given confounding variable of the given sample. Reference is the list of unique batch ids for each confounding variable.
- Condition ID (list length number of genes) List of fits for each gene to the samples of this condition. One entry of this format for all conditions fit.
  - Gene ID (list length 2) Impulse and constant model fit to gene observations. One entry of this format for all gene IDs.
    * lsImpulseFit (list) List of impulse fit parameters and results.
      · vecImpulseParam (numeric vector length 6) beta, h0, h1, h2, t1, t2 Maximum likelihood estimators of impulse model parameters.
      · vecImpulseValue (numeric vector length number of time points) Values of impulse model fit at time points used for fit.
      · lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
      · scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyperparameter).
      · scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
      · scaConvergence (scalar) Convergence status of optim on impulse model.
    * lsConstFit (list) List of constant fit parameters and results.
      · scaMu (scalar) Maximum likelihood estimator of negative binomial mean parameter.
      · lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
      · scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyperparameter).
      · scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.
      · scaConvergence (scalar) Convergence status of optim on constant model.
    * ls SigmoidFit (list) List of sigmoidal fit parameters and results. NULL if boolIdentifyTransients is FALSE.
      · vecSigmoidParam (numeric vector length 4) beta, h0, h1, t Maximum likelihood estimators of sigmoidal model parameters.
      · vecSigmoidValue (numeric vector length number of time points) Values of sigmoid model fit at time points used for fit.
      · lsvecBatchFactors (list length number of confounders) List of vectors of scalar batch correction factors for each sample. These are also maximum likelihood estimators. NULL if no confounders given.
      · scaDispParam (scalar) Dispersion parameter estimate used in fitting (hyperparameter).
      · scaLL (scalar) Loglikelihood of data under maximum likelihood estimator model.

· scaConvergence (scalar) Convergence status of optim on sigmoidal model.

`matCountDataProc` (matrix genes x samples) [Default NULL] Read count data, unobserved entries are NA. Processed matrix.

`dfAnnotationProc` (data frame samples x covariates) Sample, Condition, Time (numeric), TimeCateg (str) (and confounding variables if given). Annotation table with covariates for each sample. Processed table.

`vecDispersions` (numeric vector number of samples) Gene-wise negative binomial dispersion hyper-parameters.

`vecSizeFactors` (numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).

`boolCaseCtrl` (bool) Whether to perform case-control analysis. Does case-only analysis if FALSE.

`vecConfounders` (vector of strings number of confounding variables) Factors to correct for during batch correction. Have to supply dispersion factors if more than one is supplied. Names refer to columns in dfAnnotation.

`scaNProc` (scalar) Number of processes for parallelisation.

`scaQThres` (scalar) FDR-corrected p-value cutoff for significance.

`strReport` (str) ImpulseDE2 stdout report.

Author(s)

David Sebastian Fischer

`list_accession` *List-like accessor methods for ImpulseDE2Object*

Description

Allow usage of ImpulseDE2 ouput object like a list with respect to the core output: dfImpulseDE2Results and vecDEGenes.

Usage

```
## S4 method for signature 'ImpulseDE2Object'
names(x)

## S4 method for signature 'ImpulseDE2Object,character,missing'
x[[i, j, ...]]

## S4 method for signature 'ImpulseDE2Object'
x$name
```

Arguments

`x`	(ImpulseDE2Object) ImpulseDE2 output object.
`i, name`	(idx or str) Name or index of core output element of ImpulseDE2Object.
`j`	Not used, only vectors.
`...`	Not used.

Value

Names of core output in ImpulseDE2Object.

Target element from ImpulseDE2Object.

Target element from ImpulseDE2Object.

Author(s)

David Sebastian Fischer

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 30,
scaNImp          = 10,
scaNLin          = 10,
scaNSig          = 10)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
scaNProc        = 1 )
names(objectImpulseDE2) # Display core output
# With respect to this core output, objectImpulseDE2
# can be treated like a list.
head(objectImpulseDE2[["dfImpulseDE2Results"]])
head(objectImpulseDE2$dfImpulseDE2Results)
head(objectImpulseDE2[["vecDEGenes"]])
head(objectImpulseDE2$vecDEGenes)
```

---

plotGenes *Plots the impulse fits and data*

Description

Plots the impulse fits and data to pdf and return a list of gplots. Points are size factor normalised data. Consider using boolSimplePlot=TRUE if the plot seems to crowded.

Usage

```
plotGenes(vecGeneIDs = NULL, scaNTopIDs = NULL, objectImpulseDE2,
  boolCaseCtrl, dirOut = NULL, strFileName = "ImpulseDE2_Trajectories.pdf",
  boolMultiplePlotsPerPage = TRUE, boolSimplePlot = FALSE,
  vecRefPval = NULL, strNameRefMethod = NULL)
```

Arguments

vecGeneIDs	(string vector) [Default NULL] Gene names to be plotted. Must be in rownames of objectImpulseDE2@matCountDataProc. Supply either vecGeneIDs or scaNTopIDs.
scaNTopIDs	(int) [Default NULL] Number of top differentially expressed (by q-value) genes to be plotted Supply either vecGeneIDs or scaNTopIDs.
objectImpulseDE2	(ImpulseDE2 object) Object previously fitted to be used for plotting.
boolCaseCtrl	(bool) Whether to create case-ctrl plot.
dirOut	(dir) [Default NULL] Directory into which pdf is printed.
strFileName	(str) [Default "ImpulseDE2_Trajectories.pdf"] File name of pdf with plots.
boolMultiplePlotsPerPage	(bool) [Default TRUE] Whether to create grid with multiple plots on each page of pdf.
boolSimplePlot	(bool) [Default TRUE] Whether to omit batch structure in plotting of model fits and only plot fit to first batch/all data (if no confounders were given). This strongly simplifies plots and is recommended e.g. for case-ctrl data.
vecRefPval	(vector length vecGeneIDs) [Default NULL] P/Q-values to be displayed alongside ImpulseDE2 q-value for differential expression in plot titles.
strNameRefMethod	(str) [Default NULL] Name of reference method used to generate vecRefPval. Mentioned in plot titles.

Value

lsgplotsID (gplot list length vecGeneIDs) List of gplots for IDs in vecGeneIDs. This is secondary output next to the .pdf and can be used to extract single plots or assemble plots differently.

Author(s)

David Sebastian Fischer

See Also

Called by separately by user.

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 0,
scaNImp          = 40,
scaNLin          = 20,
scaNSig          = 40)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
```

```
vecConfounders  = NULL,
boolIdentifyTransients = FALSE,
scaNProc        = 1 )
lsgplotsID <- plotGenes(
scaNTopIDs=5,
objectImpulseDE2=objectImpulseDE2,
boolCaseCtrl=FALSE,
boolMultiplePlotsPerPage=TRUE,
boolSimplePlot=FALSE)
lsgplotsID[[1]]
```

`plotHeatmap` *Plot structured z-value heatmaps of differentially expressed genes*

Description

Creates a complexHeatmap heatmap structured into subsets of genes according to their behaviour and sorted by peak time for raw counts and for the fitted signal.

Usage

```
plotHeatmap(objectImpulseDE2, strCondition, boolIdentifyTransients,
  scaQThres = 0.01)
```

Arguments

`objectImpulseDE2`
: (instance of class ImpulseDE2Object) ImpulseDE2 output object to create heatmap from.

`strCondition`
: (str) "case","control","combined Heatmap is created from samples of this condition.

`boolIdentifyTransients`
: (bool) Whether to structure heatmap into transient and transition trajectories, only possible if sigmoids were fit to the indicated condition.

`scaQThres`
: (scalar) FDR-corrected p-value threshold for calling differentially expressed genes: Only genes below this threshold are included in the heatmap.

Value

(list length 3)

- complexHeatmapRaw (complexHeatmap plot) Heatmap of raw data by time point: Average of the size factor (and batch factor) normalised counts per time point and gene. Plot with draw(complexHeatmapRaw).
- complexHeatmapFit (complexHeatmap plot) Heatmap of impulse-fitted data by time point. Plot with draw(complexHeatmapFit).
- lsvecGeneGroups (list) List of gene ID vectors: One per heatmap group with all gene IDs of the the profiles displayed in the heatmap.

Author(s)

David Sebastian Fischer

See Also

Called seperately by used.

Examples

```
library(ComplexHeatmap)
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 0,
scaNImp          = 50,
scaNLin          = 0,
scaNSig          = 50)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
boolIdentifyTransients = TRUE,
scaNProc        = 1 )
lsHeatmaps <- plotHeatmap(
objectImpulseDE2=objectImpulseDE2,
strCondition="case",
boolIdentifyTransients=TRUE,
scaQThres=0.01)
draw(lsHeatmaps$complexHeatmapRaw)
```

---

processData *Check and process input to runImpulseDE2()*

Description

Check validity of input and process count data matrix and annotation into data structures used later in runImpulseDE2. processData is structure in the following way:

- Subhelper functions:
  - checkNull() Check whether object was supplied (is not NULL).
  - checkDimMatch() Checks whether dimensions of matrices agree.
  - checkElementMatch() Checks whether vectors are identical.
  - checkNumeric() Checks whether elements are numeric.
  - checkProbability() Checks whether elements are probabilities.
  - checkCounts() Checks whether elements are count data.
- Helper functions:
  - checkData() Check format and presence of input data.

  - nameGenes() Name genes if names are not given.
  - procAnnotation() Add categorial time variable to annotation table. Add nested batch column if necessary. Reduce to samples used.
  - reduceCountData() Reduce count data to data which are utilised later.
- Script body

Usage

```
processData(dfAnnotation, matCountData, boolCaseCtrl, vecConfounders,
  vecDispersionsExternal, vecSizeFactorsExternal)
```

Arguments

`dfAnnotation`	(data frame samples x covariates) Sample, Condition, Time (numeric), TimeCateg (str) (and confounding variables if given). Annotation table with covariates for each sample.
`matCountData`	(matrix genes x samples) [Default NULL] Read count data, unobserved entries are NA.
`boolCaseCtrl`	(bool) Whether to perform case-control analysis. Does case-only analysis if FALSE.
`vecConfounders`	(vector of strings number of confounding variables) Factors to correct for during batch correction. Have to supply dispersion factors if more than one is supplied. Names refer to columns in dfAnnotation.
`vecDispersionsExternal`	(vector length number of genes in matCountData) [Default NULL] Externally generated list of gene-wise dispersion factors which overides DESeq2 generated dispersion factors.
`vecSizeFactorsExternal`	(vector length number of cells in matCountData) [Default NULL] Externally generated list of size factors which override size factor computation in ImpulseDE2.

Value

(list length 4)

- matCountDataProc (matrix genes x samples) Read count data.
- dfAnnotationProc (data frame samples x covariates) Sample, Condition, Time (numeric), TimeCateg (str) (and confounding variables if given). Processed annotation table with covariates for each sample.
- vecSizeFactorsExternalProc (numeric vector number of samples) Model scaling factors for each sample which take sequencing depth into account (size factors).
- vecDispersionsExternalProc (vector number of genes) Gene-wise negative binomial dispersion hyper-parameter.
- strReportProcessing (str) String of stdout of processData().

Author(s)

David Sebastian Fischer

See Also

Called by runImpulseDE2.

`runDEAnalysis` *Perform differential expression analysis and identification of transiently activated or deactivated genes.*

Description

Performs model selction based on loglikelihood ratio tests. The primary model selection is the differential expression analysis. The secondary model selection is the selection between a sigmoidal and an impulse fit for differentially expressed genes which is used to define transiently activated or deactivated genes.

Usage

```
runDEAnalysis(objectImpulseDE2, boolCaseCtrl, vecConfounders,
  boolIdentifyTransients, scaQThresTransients = 0.001)
```

Arguments

`objectImpulseDE2`
: (object class ImpulseDE2Object) Object containing fits to be evaluated.

`boolCaseCtrl`
: (bool) Whether to perform case-control analysis. Does case-only analysis if FALSE.

`vecConfounders`
: (vector of strings number of confounding variables) Factors to correct for during batch correction. Names refer to columns in dfAnnotation.

`boolIdentifyTransients`
: (bool) [Defaul FALSE] Whether to identify transiently activated or deactivated genes. This involves an additional fitting of sigmoidal models and hypothesis testing between constant, sigmoidal and impulse model.

`scaQThresTransients`
: (scalar) [Default 0.001] FDR-corrected p-value threshold for hypothesis tests between impulse, sigmoidal and constant model used to identify transiently regulated genes.

Value

objectImpulseDE2 (ImpulseDE2Object) Input object with dfDEAnalysis updated to: dfDEAnalysis (data frame samples x reported characteristics) Summary of fitting procedure and differential expression results for each gene.

- Gene: Gene ID.
- p: P-value for differential expression.
- padj: Benjamini-Hochberg false-discovery rate corrected p-value for differential expression analysis.
- loglik_full: Loglikelihood of full model.
- loglik_red: Loglikelihood of reduced model.
- df_full: Degrees of freedom of full model.

- df_red: Degrees of freedom of reduced model
- mean: Inferred mean parameter of constant model of first batch. From combined samples in case-ctrl.
- allZero (bool) Whether there were no observed non-zero observations of this gene. If TRUE, fitting and DE analsysis were skipped and entry is NA.

Entries only present in case-only DE analysis:

- converge_impulse: Convergence status of optim for impulse model fit (full model).
- converge_const: Convergence status of optim for constant model fit (reduced model).

Entries only present in case-control DE analysis:

- converge_combined: Convergence status of optim for impulse model fit to case and control samples combined (reduced model).
- converge_case: Convergence status of optim for impulse model fit to samples of case condition (full model 1/2).
- converge_control: Convergence status of optim for impulse model fit to samples of control condition (full model 2/2).

Entries only present if boolIdentifyTransients is TRUE:

- converge_sigmoid: Convergence status of optim for sigmoid model fit to samples of case condition.
- impulseTOsigmoid_p: P-value of loglikelihood ratio test impulse model fit versus sigmoidal model on samples of case condition.
- impulseTOsigmoid_padj: Benjamini-Hochberg false-discovery rate corrected p-value of log-likelihood ratio test impulse model fit versus sigmoid model on samples of case condition.
- sigmoidTOconst_p: P-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- sigmoidTOconst_padj: Benjamini-Hochberg false-discovery rate corrected p-value of loglike-lihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- isTransient (bool) Whether gene is transiently activated or deactivated and differentially ex-pressed.
- isMonotonous (bool) Whether gene is not transiently activated or deactivated and differentially expressed. This scenario corresponds to a montonous expression level increase or decrease.

Author(s)

David Sebastian Fischer

See Also

Called by runImpulseDE2.

`runDESeq2` *Wrapper function for running DESeq2*

Description

Run DESeq2 and extract dispersion parameter estimates. Catch and remove dispersion outlier exception on samples with zero-count observations.

Usage

```
runDESeq2(dfAnnotationProc, matCountDataProc, boolCaseCtrl, vecConfounders)
```

Arguments

`dfAnnotationProc`	(data frame samples x covariates) Sample, Condition, Time (numeric), TimeCateg (str) (and confounding variables if given). Processed annotation table with covariates for each sample.
`matCountDataProc`	(matrix genes x samples) Read count data.
`boolCaseCtrl`	(bool) Whether to perform case-control analysis. Does case-only analysis if FALSE.
`vecConfounders`	(vector of strings number of confounding variables) Factors to correct for during batch correction. Have to supply dispersion factors if more than one is supplied. Names refer to columns in dfAnnotationProc.

Value

(numeric vector length number of genes) Dispersion parameter estimates for each gene. In format of parameter size of dnbinom which is 1/dispersion factor of DESeq2.

Author(s)

David Sebastian Fischer

See Also

Called by runImpulseDE2.

runImpulseDE2 *ImpulseDE2 wrapper*

Description

Wrapper to run ImpulseDE2 on bulk omics count data. This wrapper can perform the entire analysis pipeline of ImpulseDE2 on its own if the right parameters are supplied. To run ImpulseDE2 on bulk omics count data, use the minimal parameter set:

- matCountData
- dfAnnotation
- boolCaseCtrl
- vecConfounders

Additionally, you can provide:

- scaNProc to set the number of processes for parallelisation.
- scaQThres to set the cut off for your DE gene list.
- vecDispersionsExternal to supply external dispersion parameters which may be necessary depending on your confounding factors (runImpulseDE2 will tell you if it is necessary).
- vecSizeFactorsExternal to supply external size factors.
- boolVerbose to control stdout output.

Usage

```
runImpulseDE2(matCountData = NULL, dfAnnotation = NULL,
  boolCaseCtrl = FALSE, vecConfounders = NULL, scaNProc = 1,
  scaQThres = NULL, vecDispersionsExternal = NULL,
  vecSizeFactorsExternal = NULL, boolIdentifyTransients = FALSE,
  boolVerbose = TRUE)
```

Arguments

matCountData	(matrix genes x samples) [Default NULL] Read count data, unobserved entries are NA. Can be SummarizedExperiment object.
dfAnnotation	(data frame samples x covariates) Sample, Condition, Time (numeric), TimeCateg (str) (and confounding variables if given). Annotation table with covariates for each sample.
boolCaseCtrl	(bool) [Default FALSE] Whether to perform case-control analysis. Does case-only analysis if FALSE.
vecConfounders	(vector of strings number of confounding variables) Factors to correct for during batch correction. Have to supply dispersion factors if more than one is supplied. Names refer to columns in dfAnnotation.
scaNProc	(scalar) [Default 1] Number of processes for parallelisation.
scaQThres	(scalar) [Default NULL] FDR-corrected p-value cutoff for significance.
vecDispersionsExternal	(vector length number of genes in matCountData) [Default NULL] Externally generated list of gene-wise dispersion factors which overides DESeq2 generated dispersion factors.

`vecSizeFactorsExternal`
: (vector length number of cells in matCountData) [Default NULL] Externally generated list of size factors which override size factor computation in ImpulseDE2.

`boolIdentifyTransients`
: (bool) [Defaul FALSE] Whether to identify transiently activated or deactivated genes. This involves an additional fitting of sigmoidal models and hypothesis testing between constant, sigmoidal and impulse model.

`boolVerbose`
: (bool) [Default TRUE] Whether to print progress to stdout.

Details

ImpulseDE2 is based on the impulse model proposed by Chechik and Koller (Chechik and Koller, 2009). The computational complexity of ImpulseDE2 is linear in the number of genes and linear in the number of samples.

Value

(object of class ImpulseDE2Object) This object can be treated as a list with 2 elements: (list length 2)

- vecDEGenes (list number of genes) Genes IDs identified as differentially expressed by ImpulseDE2 at threshold `scaQThres`.
- dfDEAnalysis (data frame samples x reported characteristics) Summary of fitting procedure and differential expression results for each gene.
  - Gene: Gene ID.
  - p: P-value for differential expression.
  - padj: Benjamini-Hochberg false-discovery rate corrected p-value for differential expression analysis.
  - loglik_full: Loglikelihood of full model.
  - loglik_red: Loglikelihood of reduced model.
  - df_full: Degrees of freedom of full model.
  - df_red: Degrees of freedom of reduced model
  - mean: Inferred mean parameter of constant model of first batch. From combined samples in case-ctrl.
  - allZero (bool) Whether there were no observed non-zero observations of this gene. If TRUE, fitting and DE analsysis were skipped and entry is NA.

  Entries only present in case-only DE analysis:

  - converge_impulse: Convergence status of optim for impulse model fit (full model).
  - converge_const: Convergence status of optim for constant model fit (reduced model).

  Entries only present in case-control DE analysis:

  - converge_combined: Convergence status of optim for impulse model fit to case and control samples combined (reduced model).
  - converge_case: Convergence status of optim for impulse model fit to samples of case condition (full model 1/2).
  - converge_control: Convergence status of optim for impulse model fit to samples of control condition (full model 2/2).

  Entries only present if boolIdentifyTransients is TRUE:

- converge_sigmoid: Convergence status of optim for sigmoid model fit to samples of case condition.
- impulseTOsigmoid_p: P-value of loglikelihood ratio test impulse model fit versus sigmoidal model on samples of case condition.
- impulseTOsigmoid_padj: Benjamini-Hochberg false-discovery rate corrected p-value of loglikelihood ratio test impulse model fit versus sigmoid model on samples of case condition.
- sigmoidTOconst_p: P-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- sigmoidTOconst_padj: Benjamini-Hochberg false-discovery rate corrected p-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- isTransient (bool) Whether gene is transiently activated or deactivated and differentially expressed.
- isMonotonous (bool) Whether gene is not transiently activated or deactivated and differentially expressed. This scenario corresponds to a montonous expression level increase or decrease.

Author(s)

David Sebastian Fischer

See Also

Calls the following functions: processData, runDESeq2, computeNormConst, fitModels, fitSigmoidModels, runDEAnalysis. The following functions are additionally available to the user: fitSigmoidModels, plotGenes, plotHeatmap, runDEAnalysis, simulateDataSetImpulseDE2.

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 30,
scaNImp          = 10,
scaNLin          = 10,
scaNSig          = 10)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
scaNProc        = 1 )
head(objectImpulseDE2$dfImpulseDE2Results)
```

simulateDataSetImpulseDE2

Simulate a data set for ImpulseDE2

Description

Simulates a data set with genes with constant and impulse expression traces. Expression strength and variation in impulse like traces are parameterised and random. All temporary files are saved into dirOutSimulation and only the objects necessary for running ImpulseDE2 (the count matrix and the annotation table are returned). The remaining objects representing hidden parameters can be used to evaluate parameter estimates.

Usage

```
simulateDataSetImpulseDE2(vecTimePointsA, vecTimePointsB, vecBatchesA,
  vecBatchesB, scaNConst, scaNImp, scaNLin, scaNSig, scaNRand = 0,
  scaSeedInit = 1, scaMumax = 1000, boolOneConstMu = FALSE,
  scaSDExpressionChange = 1, scaSDRand = NULL, scaMuSizeEffect = 1,
  scaSDSizeEffect = 0.1, scaMuBatchEffect = NULL, scaSDBatchEffect = NULL,
  dirOutSimulation = NULL)
```

Arguments

vecTimePointsA	(numeric vector number of time points) Number of time points in batch A.
vecTimePointsB	(numeric vector number of time points) Number of time points in batch B.
vecBatchesA	(str vector number of samples in vecTimePointsA) [Default NULL] Batch IDs of each sample in condition A. Set to NULL if simulating without batch effects.
vecBatchesB	(str vector number of samples in vecTimePointsB) [Default NULL] Batch IDs of each sample in condition B. Set to NULL if simulating without batch effects.
scaNConst	(scalar) Number of constant genes in data set.
scaNImp	(scalar) Number of impulse distributed genes in data set.
scaNLin	(scalar) Number of linear distributed genes in data set.
scaNSig	(scalar) Number of sigmoid distributed genes in data set.
scaNRand	(scalar) [Default NULL] Number of random distributed genes in data set.
scaSeedInit	(scalar) [Default 1] Scalar based on which seeds are chosen. One vlaue correspond sto a unique set of seeds for all random number generations.
scaMumax	(scalar) [Default 1000] Maximum expression mean parameter to be used.
boolOneConstMu	(bool) [Default False] Don't sample constant trajectories from uniform [0,scaMumax] but set all to scaMumax
scaSDExpressionChange	(scalar) [Default 1] Standard deviation of normal distribution from which the amplitude change within an impulse trace is drawn.
scaSDRand	(scalar) [Default 0] Standard deviation of normal distribution from which the random deviations are drawn.
scaMuSizeEffect	(numeric vector number of genes) [Default NULL] Mean of normal distribution of which scaNLing factor for size effects per sample are drawn.

`scaSDSizeEffect`
: (numeric vector number of genes) [Default NULL] Standard deviation of normal distribution of which scaling factor for size effects per sample are drawn.

`scaMuBatchEffect`
: (numeric vector number of genes) [Default NULL] Mean of normal distribution of which scaling factor for batch effects per gene are drawn (reference is batch A).

`scaSDBatchEffect`
: (numeric vector number of genes) [Default NULL] Standard deviation of normal distribution of which scaling factor for batch effects per gene are drawn (reference is batch A).

`dirOutSimulation`
: (directory) [Default NULL] Directory to which simulated parameter objects are saved to.

Value

list (length 2)

- dfAnnotation (data frame samples x covariates) Sample, Condition, Time (numeric), TimeCateg (str) (and confounding variables if given). Annotation table with covariates for each sample.
- matSampledCountsObserved (matrix genes x cells) Sampled count data of all cells after dropout.

Author(s)

David Sebastian Fischer

See Also

Called by separately by user.

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 30,
scaNImp          = 10,
scaNLin          = 10,
scaNSig          = 10)
```

`updateDEAnalysis` *Update dfImpulseDE2Results after sigmoids have been fit through external call*

Description

This is a userfriendly wrapper of runDEAnalysis for this update scenario.

Usage

```
updateDEAnalysis(objectImpulseDE2, scaQThresTransients = 0.001)
```

Arguments

`objectImpulseDE2`
: (object class ImpulseDE2Object) Object containing fits to be evaluated.

`scaQThresTransients`
: (scalar) [Default 0.001] FDR-corrected p-value threshold for hypothesis tests between impulse, sigmoidal and constant model used to identify transiently regulated genes.

Value

objectImpulseDE2 (ImpulseDE2Object) Input object with dfDEAnalysis updated to: dfDEAnalysis (data frame samples x reported characteristics) Summary of fitting procedure and differential expression results for each gene.

- Gene: Gene ID.
- p: P-value for differential expression.
- padj: Benjamini-Hochberg false-discovery rate corrected p-value for differential expression analysis.
- loglik_full: Loglikelihood of full model.
- loglik_red: Loglikelihood of reduced model.
- df_full: Degrees of freedom of full model.
- df_red: Degrees of freedom of reduced model
- mean: Inferred mean parameter of constant model of first batch. From combined samples in case-ctrl.
- allZero (bool) Whether there were no observed non-zero observations of this gene. If TRUE, fitting and DE analsysis were skipped and entry is NA.

Entries only present in case-only DE analysis:

- converge_impulse: Convergence status of optim for impulse model fit (full model).
- converge_const: Convergence status of optim for constant model fit (reduced model).

Entries only present in case-control DE analysis:

- converge_combined: Convergence status of optim for impulse model fit to case and control samples combined (reduced model).

- converge_case: Convergence status of optim for impulse model fit to samples of case condition (full model 1/2).
- converge_control: Convergence status of optim for impulse model fit to samples of control condition (full model 2/2).

Entries only present if boolIdentifyTransients is TRUE:

- converge_sigmoid: Convergence status of optim for sigmoid model fit to samples of case condition.
- impulseTOsigmoid_p: P-value of loglikelihood ratio test impulse model fit versus sigmoidal model on samples of case condition.
- impulseTOsigmoid_padj: Benjamini-Hochberg false-discovery rate corrected p-value of log-likelihood ratio test impulse model fit versus sigmoid model on samples of case condition.
- sigmoidTOconst_p: P-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- sigmoidTOconst_padj: Benjamini-Hochberg false-discovery rate corrected p-value of loglikelihood ratio test sigmoidal model fit versus constant model on samples of case condition.
- isTransient (bool) Whether gene is transiently activated or deactivated and differentially expressed.
- isMonotonous (bool) Whether gene is not transiently activated or deactivated and differentially expressed. This scenario corresponds to a montonous expression level increase or decrease.

Author(s)

David Sebastian Fischer

See Also

Called by separately by user.

Examples

```
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 0,
scaNImp          = 50,
scaNLin          = 0,
scaNSig          = 50)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
boolIdentifyTransients = FALSE,
scaNProc        = 1 )
# You could have used boolIdentifyTransients=TRUE
# to avoid the following post wrapper fitting.
objectImpulseDE2 <- fitSigmoidModels(
objectImpulseDE2 = objectImpulseDE2,
vecConfounders   = NULL,
```

```
strCondition     = "case")
objectImpulseDE2 <- updateDEAnalysis(
objectImpulseDE2=objectImpulseDE2,
scaQThresTransients=0.001)
head(objectImpulseDE2$dfImpulseDE2Results)
# dfImpulseDE2Results now contain 'transients-analysis'.
```

`writeReportToFile` *Print ImpulseDE2 report to .txt file*

Description

Print ImpulseDE2 report to .txt file.

Usage

```
writeReportToFile(object, fileReport)
```

Arguments

`object`	(ImpulseDE2Object) Output object of ImpulseDE2.
`fileReport`	(file) File to print report to.

Value

No return.

Author(s)

David Sebastian Fischer

Examples

```
dirPWD <- getwd() # Will save into current working directory.
lsSimulatedData <- simulateDataSetImpulseDE2(
vecTimePointsA   = rep(seq(1,8),3),
vecTimePointsB   = NULL,
vecBatchesA      = NULL,
vecBatchesB      = NULL,
scaNConst        = 30,
scaNImp          = 10,
scaNLin          = 10,
scaNSig          = 10)
objectImpulseDE2 <- runImpulseDE2(
matCountData    = lsSimulatedData$matObservedCounts,
dfAnnotation    = lsSimulatedData$dfAnnotation,
boolCaseCtrl    = FALSE,
vecConfounders  = NULL,
scaNProc        = 1 )
# Uncomment to run:
#writeReportToFile(
#object=objectImpulseDE2,
#file=paste0(dirPWD, "ImpulseDE2Report.txt")
```

```
#)
```

Index