ImpulseDE2 Vignette

June 27, 2016

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

Description

Compute value of impulse function given parameters.

Usage

```
calcImpulse(vecTheta, vecTimepoints)
```

Arguments

`vecTheta`	(vector number of parameters) Numerical vector of impulse parameters with the order beta, h0, h1, h2, t1, t2.
`vecTimepoints`	(vector number vecTimepoints) Observed vecTimepoints, numeric.

Value

vecY (vec number of vecTimepoints) Model expression values of given gene for time points

See Also

Called by `evalLogLikImpulse`,`evalLogLikMean`, `plotDEGenes`.

`compareDEMethods` *Compare ImpulseDE2 output against other differential expression method*

Description

The comparison is performed based on the adjusted p-values of differential expression. This methods allows the user to explore visually how ImpulseDE2 output differs from similar methods.

Usage

```
compareDEMethods(matQval, strMethod1 = "ImpulseDE2", strMethod2 = "",
  Q = 10^(-3), Qdelta = 10^(2), matCountDataProc, matTranslationFactors,
  matSizeFactors, dfAnnotationProc, lsImpulseFits, dfImpulseResults,
  strCaseName = "case", strControlName = NULL, strMode = "batch",
  strDataDescriptionFilename = "")
```

Arguments

matQval:	matrix with p-values by methods

See Also

Auxillary method not called during ImpulseDE2 running. Called separately by user.

---

computeLogLikNull *Fit and compute likelihood of null model for a single gene*

Description

Fits an impulse model and a mean model to a single gene. The optimisation method is set within this function (optim: BFGS). This method is divided into four parts: (I) Prepare data, (II) Fit mean model, (III) Fit impulse model, (IV) Process Fits. Internal function of fitImpulse.

Usage

```
computeLogLikNull(vecCounts, scaDispersionEstimate, vecDropoutRate = NULL,
  vecProbNB = NULL, vecNormConst, vecLongitudinalSeries = NULL,
  vecLongitudinalSeriesAssign = NULL, vecboolObserved, vecboolZero = NULL,
  vecboolNotZeroObserved = NULL, strMode)
```

Arguments

vecLongitudinalSeriesAssign	(numeric vector number samples) Longitudinal sample series assigned to samples. NULL if not operating in strMode="longitudinal".
vecCounts:	(count vector number of samples) Count data.
scaDispersionEstimate:	(scalar) Negative binomial dispersion coefficient for given gene.
vecDropoutRate:	(probability vector number of samples) [Default NULL] Dropout rate/mixing probability of zero inflated negative binomial mixturemodel for each gene and cell.
vecProbNB:	(probability vector number of samples) Probability of observations to come from negative binomial component of mixture model.
vecNormConst:	(numeric vector number of samples) Normalisation constants for each sample.
vecLongitudinalSeries:	(string vector number of longitudinal sample series) Longitudinal sample series.
vecboolObserved:	(bool vector number of samples) Whether sample is observed (not NA).

vecboolZero: (bool vector number of samples) Whether sample has zero count.

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

strMode: (str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

Value

(list length 3)

- scaMu: (scalar) Null model in batch or singlecell mode: One overall mean.
- vecMuLongitudinalSeries: (numerical vector length number of longitudinal series) Null model in longitudinal mode: One mean per longitudinal series. Null if mode is not longitudinal.
- scaLogLikNull: (scalar) Log likelihood of null model.

See Also

Called by fitImpulse_gene.

---

computeNormConst *Compute normalisation constant for each replicate*

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(matCountDataProcFull, matCountDataProc, vecDispersions,
  scaSmallRun = NULL, dfAnnotationProc, strCaseName, strControlName = NULL,
  strMode = "batch", vecSizeFactorsExternal = NULL)
```

Arguments

strCaseName (str) Name of the case condition in dfAnnotationRedFull.

matCountDataProcFull:
(matrix genes x samples) Count data: Reduced version of matCountData. For internal use. This is the entire data set.

matCountDataProc:
(matrix genes x samples) Count data: Reduced version of matCountData. For internal use. This is the data set reduced to the genes which are supposed to be analysed.

scaSmallRun: (integer) [Default NULL] Number of rows on which ImpulseDE2 is supposed to be run, the full data set is only used for size factor estimation.

`dfAnnotationProc`:
(Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and LongitudinalSeries). For internal use.

`strControlName`:
(str) [Default NULL] Name of the control condition in `dfAnnotationRedFull`.

`strMode`: (str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

`vecSizeFactorsExternal`:
(numeric vector number of cells) Model scaling factors for each observation which take sequencing depth into account (size factors). One size factor per cell. These are supplied to ImpulseDE, if this variable is not set, size factors are computed in this function.

Value

(list lsMatTranslationFactors,matSizeFactors)

- matTranslationFactors: (numeric matrix genes x samples) Model scaling factors for each observation which take longitudinal time series mean within a gene into account (translation factors). Computed based based on all samples.
- matSizeFactors: (numeric matrix genes x samples) Model scaling factors for each observation which take sequencing depth into account (size factors). One size factor per sample - rows of this matrix are equal.

See Also

Called by `runImpulseDE2`. Calls `computeTranslationFactors` and `computeSizeFactors`.

`computePval` *Compute p-values for model fit*

Description

Compute p-value of differential expression based on chi-squared distribution of deviance of likli-hoods and report summary of statistics.

Usage

```
computePval(matCountDataProc, vecDispersions, dfAnnotationProc, lsImpulseFits,
  strCaseName = NULL, strControlName = NULL, strMode = "batch",
  NPARAM = 6)
```

Arguments

`vecDispersions` (vector number of genes) Inverse of gene-wise negative binomial dispersion coefficients computed by DESeq2.

`lsImpulseFits` (list length 2 or 6) List of matrices which contain parameter fits and model values for given time course for the case condition (and control and combined if control is present). Each parameter matrix is called parameter_'condition' and has the form (genes x [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0,) where beta to t2 are parameters of the impulse model, mu is the single parameter

of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by `optim` from `stats`. Each value matrix is called value_'condition' and has the form (genes x time points) and contains the counts predicted by the impulse model at the observed time points.

`strCaseName` (str) Name of the case condition in `dfAnnotationRedFull`.

`NPARAM` (scalar) [Default 6] Number of parameters of impulse model.

`matCountDataProc:`
(matrix genes x samples) Count data: Reduced version of `matCountData`. For internal use.

`dfAnnotationProc:`
(Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (categorial) (and LongitudinalSeries). For internal use.

`dfDEAnalysis` (data frame genes x fitting characteristics)

`strControlName:`
(str) [Default NULL] Name of the control condition in `dfAnnotationRedFull`.

`strMode:` (str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

Value

dfDEAnalysis (data frame genes x fitting characteristics) Summary of fitting procedure for each gene.

See Also

Called by `runImpulseDE2`.

`computeSizeFactors` *Compute size factors for a dataset*

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, scaSmallRun = NULL, strMode)
```

Arguments

`matCountDataProc:`
(matrix genes x samples) Count data: Reduced version of `matCountData`. For internal use.

`scaSmallRun:` (integer) [Default NULL] Number of rows on which ImpulseDE2 is supposed to be run, the full data set is only used for size factor estimation.

`strMode:` (str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

Value

matSizeFactors: (numeric matrix genes x samples) Model scaling factors for each observation which take sequencing depth into account (size factors). One size factor per sample - rows of this matrix are equal.

See Also

Called by `computeNormConst`.

`computeTranslationFactors`

Compute translation factors for a dataset

Description

This function computes translation factors for each sample per gene in the dataset. Note that the translation factors are computed with respect to the overall mean, and with respect to the condition-wise means if control samples are given. The rescaling of translation factors for case and control conditions is not necessary for differential expression analysis but gives impulse models which directly represent the data of their conditions, without scaling. Translation factors scale the negative binomial likelihood model of a gene to the mean of a longitudinal series of samples of this gene. This function is only callde if strMode="longitudinal".

Usage

```
computeTranslationFactors(matCountDataProc, matSizeFactors, vecDispersions,
  scaSmallRun = NULL, dfAnnotationProc, strCaseName, strControlName = NULL)
```

Arguments

`strCaseName` (str) Name of the case condition in `dfAnnotationRedFull`.

`matCountDataProc`:
(matrix genes x samples) Count data: Reduced version of `matCountData`. For internal use.

`matSizeFactors`:
(numeric matrix genes x samples) Model scaling factors for each observation which take sequencing depth into account (size factors).

`scaSmallRun`: (integer) [Default NULL] Number of rows on which ImpulseDE2 is supposed to be run, the full data set is only used for size factor estimation.

`dfAnnotationProc`:
(Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and LongitudinalSeries). For internal use.

`strControlName`:
(str) [Default NULL] Name of the control condition in `dfAnnotationRedFull`.

Value

matTranslationFactors: (numeric matrix genes x samples) Model scaling factors for each observation which take longitudinal time series mean within a gene into account (translation factors). Computed based based on all samples.

See Also

Called by computeNormConst.

`estimateImpulseParam` *Estimate impulse model parameter initialisations*

Description

The initialisations reflect intuitive parameter choices corresponding to a peak and to a valley model.

Usage

```
estimateImpulseParam(vecTimepoints, vecCounts, vecDropoutRate = NULL,
  vecProbNB = NULL, strSCMode = "clustered", scaWindowRadius = NULL,
  vecTimepointAssign, vecNormConst, strMode)
```

Arguments

vecTimepoints:	(numeric vector number of timepoints) Time-points at which gene was sampled.
vecCounts:	(count vector number of samples) Count data.
vecTimepointAssign:	(numeric vector number samples) Timepoints assigned to samples.
vecNormConst:	(numeric vector number of samples) Normalisation constants for each sample.
strMode:	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.
vecProbNB:	(probability vector number of samples) Probability of observations to come from negative binomial component of mixture model.

Value

vecParamGuessPeak: (numeric vector number of impulse model parameters) Impulse model parameter initialisation corresponding to a peak.

See Also

Called by fitImpulse_gene.

`evalLogLikImpulseBatch`
Cost function impulse model fit - Batch mode

Description

Log likelihood cost function for impulse model fit based on negative binomial model. This cost function is called in the modes "batch" and "longitudinal". In analogy to generalised linear models, a log linker function is used for the count parameters. The inferred negative binomial model is scaled by the factors in vecNormConst, which represent size factors (and translation factors), for evaluation of the likelihood on the data.

Usage

```
evalLogLikImpulseBatch(vecTheta, vecX, vecY, scaDispEst, vecNormConst,
  vecindTimepointAssign, vecboolObserved)
```

Arguments

vecTheta	(vector number of parameters [6]) Impulse model parameters.
vecX	(numeric vector number of timepoints) Time-points at which gene was sampled.
vecY	(count vector samples) Observed expression values for given gene.
vecindTimepointAssign	(numeric vector number samples) Index of time point assigned to sample in list of sorted time points (vecX).
scaDispEst:	(scalar) Dispersion estimate for given gene.
vecNormConst:	(numeric vector number of samples) Normalisation constants for each sample.
vecboolObserved:	(bool vector number of samples) Stores bool of sample being not NA (observed).

Value

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

See Also

Called by fitImpulse:fitImpulse_gene. Calls calcImpulse. evalLogLikImpulseByTC for dependent residuals (i.e. time course experiments)

---

evalLogLikImpulseSC *Cost function impulse model fit - Single cell mode*

Description

Log likelihood cost function for impulse model fit based on zero inflated negative binomial mode. This cost function is appropriate for sequencing data with high drop out rate, commonly observed in single cell data (e.g. scRNA-seq). In analogy to generalised linear models, a log linker function is used for the count parameters. The impulse model values (negative binomial mean parameters) are normalised by vecNormConst for evaluation of the likelihood on the data.

Usage

```
evalLogLikImpulseSC(vecTheta, vecX, vecY, scaDispEst, vecDropoutRateEst,
  vecNormConst, vecindTimepointAssign, vecboolNotZeroObserved, vecboolZero,
  scaWindowRadius = NULL)
```

Arguments

vecTheta	(vector number of parameters [6]) Impulse model parameters.
vecX	(numeric vector number of timepoints) Time-points at which gene was sampled.
vecY	(count vector samples) Observed expression values for given gene.
vecindTimepointAssign	(numeric vector number samples) Index of time point assigned to sample in list of sorted time points (vecX).
scaDispEst:	(scalar) Negative binomial dispersion parameter for given gene.
vecDropoutRateEst:	(probability vector number of samples) Dropout rate estimate for each cell for given gene.
vecNormConst:	(numeric vector number of samples) Normalisation constants for each sample.
vecboolNotZeroObserved:	(bool vector number of samples) Whether sample is not zero and observed (not NA).
vecboolZero:	(bool vector number of samples) Whether sample has zero count.

Value

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

See Also

Called by fitImpulse::fitImpulse_matrix:: fitImpulse_gene::optimiseImpulseModelFit. Calls calcImpulse and evalLogLikZINB_comp.

---

evalLogLikNBMean *Cost function for negative binomial mean parameter fit*

Description

Log likelihood cost function for numerical optimisation of mean model fit based on negative binomial model. Note that the closed form solution of the negative binomial mean parameter only holds if all normalisation factors are 1. In analogy to generalised linear models, a log linker function is used for the mean. The inferred negative binomial mean model is scaled by the factors in vecNormConst, which represent size factors (and translation factors), for evaluation of the likelihood on the data.

Usage

```
evalLogLikNBMean(scaTheta, vecCounts, scaDispEst, vecNormConst, vecboolObserved)
```

Arguments

scaTheta	(vector number of parameters [6]) Negative binomial mean parameter.
vecCounts	(count vector samples) Observed expression values for given gene.
scaDispEst:	(scalar) Dispersion estimate for given gene.
vecNormConst:	(numeric vector number of samples) Normalisation constants for each sample.
vecWeights:	(probability vector number of samples) Weights for inference on mixture models.
vecboolObserved:	(bool vector number of samples) Stores bool of sample being not NA (observed).

Value

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

---

evalLogLikZINB *Compute log likelihood of zero-inflated negative binomial model*

Description

This liklihood function is appropriate for sequencing data with high drop out rate, commonly observed in single cell data (e.g. scRNA-seq).

Usage

```
evalLogLikZINB(vecY, vecMu, scaDispEst, vecDropoutRateEst,
  vecboolNotZeroObserved, vecboolZero)
```

Arguments

vecY	(count vector number of amples) Observed expression values for given gene.
vecMu	(vector number of samples) Negative binomial mean parameter for each sample.
scaDispEst:	(scalar) Negative binomial dispersion parameter for given gene.
vecDropoutRateEst:	(probability vector number of samples) Dropout rate estimate for each cell for given gene.
vecboolNotZeroObserved:	(bool vector number of samples) Whether sample is not zero and observed (not NA).
vecboolZero:	(bool vector number of samples) Whether sample has zero count.

Value

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

See Also

Called by fitImpulse::fitImpulse_matrix:: fitImpulse_gene::computeLogLikNull and evalLogLikImpulseSC.

`fitImpulse` *Fits impulse model to a timecourse dataset*

Description

This function processes the input matrix and coordinates impulse model fitting through `impulse_fit_matrix`. [Helper `fitImpulse_matrix`] Fit impulse model to matrix of genes. Calls `fitImpulse_gene`. `fitImpulse_matrix` fits impulse models to a matrix of samples from one condition.

Usage

```
fitImpulse(matCountDataProc, dfAnnotationProc, matTranslationFactors,
  matSizeFactors, vecDispersions, matDropoutRate, matProbNB,
  vecClusterAssignments, strCaseName, strControlName = NULL,
  strMode = "batch", strSCMode = "clustered", scaWindowRadius = NULL,
  nProc = 1, NPARAM = 6)
```

Arguments

`strCaseName`	(str) Name of the case condition in `dfAnnotationRedFull`.
`matCountDataProc:`	(matrix genes x samples) Count data: Reduced version of `matCountData`. For internal use.
`dfAnnotationProc:`	(Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and LongitudinalSeries). For internal use.
`matTranslationFactors:`	(numeric matrix genes x samples) Model scaling factors for each observation which take longitudinal time series mean within a gene into account (translation factors). Computed based based on all samples.
`matSizeFactors:`	(numeric matrix genes x samples) Model scaling factors for each observation which take sequencing depth into account (size factors). One size factor per sample - rows of this matrix are equal.
`vecDispersions:`	(vector number of genes) Gene-wise negative binomial dispersion coefficients.
`matDropoutRate:`	(probability matrix genes x samples) Dropout rate/mixing probability of zero inflated negative binomial mixture model for each gene and cell.
`matProbNB:`	(probability matrix genes x samples) Probability of observations to come from negative binomial component of mixture model.
`strControlName:`	(str) [Default NULL] Name of the control condition in `dfAnnotationRedFull`.
`strMode:`	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.
`nProc:`	(scalar) [Default 1] Number of processes for parallelisation.
`NPARAM:`	(scalar) [Default 6] Number of parameters of impulse model.

Value

lsFitResults_all (list length 2 or 6) List of matrices which contain parameter fits and model values for given time course for the case condition (and control and combined if control is present). Each parameter matrix is called parameter_'condition' and has the form (genes x [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0, converge_H0]) where beta to t2 are parameters of the impulse model, mu is the single parameter of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by `optim` from `stats` of either model. Each value matrix is called value_'condition' and has the form (genes x time points) and contains the counts predicted by the impulse model at the observed time points.

See Also

Called by `runImpulseDE2`. Calls `fitImpulse_matrix`.

`fitImpulse_gene` *Fit an impulse model to a single gene*

Description

Fits an impulse model and a mean model to a single gene. The optimisation method is set within `optimiseImpulseModelFit` (`optim`: BFGS). This function is divided into four parts: (I) Prepare data, (II) Fit mean model, (III) Fit impulse model, (IV) Process Fits. The body of this function is broken up into 4 helper functions, which are exclusively called by this function.

Usage

```
fitImpulse_gene(vecCounts, scaDispersionEstimate, vecDropoutRate = NULL,
  vecProbNB = NULL, vecNormConst, vecTimepointAssign,
  vecLongitudinalSeriesAssign, dfAnnotationProc, strMode = "batch",
  strSCMode = "clustered", scaWindowRadius = NULL, NPARAM = 6,
  MAXIT = 1000)
```

Arguments

`vecLongitudinalSeriesAssign`
: (numeric vector number samples) Longitudinal series assigned to samples. NULL if not operating in strMode="longitudinal".

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

`scaDispersionEstimate:`
: (scalar) Inverse of negative binomial dispersion coefficients computed by DESeq2 for given gene.

`vecDropoutRate:`
: (probability vector number of samples) [Default NULL] Dropout rate/mixing probability of zero inflated negative binomial mixturemodel for each gene and cell.

`vecProbNB:`
: (probability vector number of samples) [Default NULL] Probability of observations to come from negative binomial component of mixture model.

vecNormConst: (numeric vector number of samples) Model scaling factors for each observation: Take sequencing depth and longitudinal time series mean within a gene into account (size and translation factors).

vecTimepointAssign: (numeric vector number samples) Timepoints assigned to samples.

dfAnnotationProc: (Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and LongitudinalSeries). For internal use.

strMode: (str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

NPARAM: (scalar) [Default 6] Number of impulse model parameters

MAXIT: (scalar) [Default 100] Number of iterations, which are performed to fit the impulse model to the clusters.

Value

vecBestFitSummary: (vector [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0]) Beta to t2 are parameters of the impulse model, mu is the single parameter of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by `optim` from `stats`.

See Also

Called by `fitImpulse_matrix`. This function calls `computeLogLikNull`, `estimateImpulseParam` and `optimiseImpulseModelFit`.

---

fitImpulse_matrix *Fits impulse models to all genes of a dataset*

Description

Fits impulse models to all genes of a dataset. Performs parallelisation. Internal function of `fitImpulse`.

Usage

```
fitImpulse_matrix(matCountDataProcCondition, vecDispersions,
  matDropoutRate = NULL, matProbNB = NULL, matNormConst, vecTimepointAssign,
  vecLongitudinalSeriesAssign, dfAnnotationProc, strCaseName,
  strControlName = NULL, strMode = "batch", strSCMode = "clustered",
  scaWindowRadius = NULL, nProc = 1, NPARAM = 6)
```

Arguments

matCountDataProc: (count matrix genes x samples) Count data.

vecDispersions: (vector number of genes) Gene-wise negative binomial dispersion coefficients.

matDropoutRate: (probability matrix genes x samples) Dropout rate/mixing probability of zero inflated negative binomial mixture model for each gene and cell.

matProbNB: (probability vector genes x samples) Probability of observations to come from negative binomial component of mixture model.

matNormConst: (numeric matrix genes x samples) Model scaling factors for each observation: Take sequencing depth and longitudinal time series mean within a gene into account (size and translation factors).

vecTimepointAssign: (numeric vector number samples) Timepoints assigned to samples.

vecLongitudinalSeriesAssign: (numeric vector number samples) Time courses assigned to samples. NULL if not operating in strMode="longitudinal".

dfAnnotationProc: (Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and LongitudinalSeries). For internal use.

strCaseName: (str) Name of the case condition in dfAnnotationRedFull.

strControlName: (str) [Default NULL] Name of the control condition in dfAnnotationRedFull.

strMode: (str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

nProc: (scalar) [Default 3] Number of processes for parallelisation.

NPARAM: (scalar) [Default 6] Number of parameters of impulse model.

Details

Maximum number of iterations for optimisation is set within this function as MAXIT. In the case of single condition differential expression over time, this function is called once for the case condition. In the case of case and control condition, this function is called three times: data from case, control and combined conditions. Calls fitImpulse_gene.

Value

lsFitResults_matrix (list length 2) List of two matrices which contain parameter fits and model values for given time course for the given condition.

- parameter(genes x [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0, converge_H0]) Beta to t2 are parameters of the impulse model, mu is the single parameter of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by optim from stats of either model.
- values (genes x time points) Contains the counts predicted by the impulse model at the observed time points.

See Also

Called by fitImpulse. Calls fitImpulse_gene.

`fitNBMean` *Fit negative binomial mean parameter*

Description

Numerical optimisation of negative binomial mean model fit. Note that the closed form solution of the maximum likelihood estimator of the negative binomial mean parameter (the weighted average) only holds if all normalisation factors are 1. Catches numerical errors.

Usage

```
fitNBMean(vecCounts, scaDispEst, vecNormConst)
```

Arguments

`vecCounts`	(count vector samples) Observed expression values for given gene.
`scaDispEst:`	(scalar) Dispersion estimate for given gene.
`vecNormConst:`	(numeric vector number of samples) Normalisation constants for each sample.
`vecWeights:`	(probability vector number of samples) Weights for inference on mixture models.
`vecboolObserved:`	(bool vector number of samples) Stores bool of sample being not NA (observed).

Value

scaMu: (scalar) Maximum likelihood estimator of negative binomial mean parameter.

See Also

Called by `computeTranslationFactors()` and `computeLogLikNull()`.

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

Description

Given a parameter initialisation, this function performs numerical optimisation using BFGS of the likelihood function given the impulse model and returns the fitted (maximum likelihood) model.

Usage

```
optimiseImpulseModelFit(vecParamGuess, vecTimepoints, vecCounts,
  scaDispersionEstimate, vecDropoutRate = NULL, vecNormConst,
  vecindTimepointAssign, vecboolObserved, vecboolZero = NULL,
  vecboolNotZeroObserved = NULL, scaWindowRadius = NULL,
  strMode = "batch", MAXIT = 100)
```

Arguments

vecTimepoints	(numeric vector number of timepoints) Time-points at which gene was sampled.
vecCounts	(count vector numer of samples)
vecindTimepointAssign	(numeric vector number samples) Index of time point assigned to sample in list of sorted time points (vecX).
vecParamGuessPeak	(vector number of parameters [6]) Impulse model parameters.
scaDispersionEstimate:	(scalar) Dispersion estimate for given gene.
vecDropoutRate:	(probability vector number of samples) Dropout rate estimate for each cell for given gene.
vecNormConst:	(numeric vector number of samples) Normalisation constants for each sample.
vecboolObserved:	(bool vector number of samples) Whether sample is observed (not NA).
vecboolZero:	(bool vector number of samples) Whether sample has zero count.
vecboolNotZeroObserved:	(bool vector number of samples) Whether sample is not zero and observed (not NA).
strMode:	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.
MAXIT:	(scalar) [Default 100] Number of iterations, which are performed to fit the impulse model to the clusters.

Value

vecFit: (vector [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1) Impulse model parameters, likelihood of data under fitted impulse model with given initialiation and convergence status of numerical optimisation.

See Also

Called by `fitImpulse_gene`. This function calls the cost function `evalLogLikImpulseBatch_comp`, `evalLogLikImpulseTC_comp` or `evalLogLikImpulseSC_comp` depending on the mode.

---

plotDEGenes *Plots the impulse fits and data to pdf*

Description

Plots the impulse fits and data to pdf.

Usage

```
plotDEGenes(vecGeneIDs, matCountDataProc, matTranslationFactors = NULL,
  matSizeFactors, dfAnnotationProc, lsImpulseFits, matMuCluster = NULL,
  vecCentroids = NULL, vecClusterAssignments = NULL, dfImpulseResults,
  vecRefPval = NULL, strCaseName, strControlName = NULL,
  strMode = "batch", strSCMode = "clustered", strFileNameSuffix = "",
  strPlotTitleSuffix = "", strPlotSubtitle = "",
  strNameMethod1 = "ImpulseDE2", strNameMethod2 = NULL,
  boolSimplePlot = FALSE, boolLogPlot = FALSE, NPARAM = 6)
```

Arguments

vecGeneIDs	(string vector) Gene names to be plotted,
lsImpulseFits	(list length 2 or 6) List of matrices which contain parameter fits and model values for given time course for the case condition (and control and combined if control is present). Each parameter matrix is called parameter_'condition' and has the form (genes x [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0,) where beta to t2 are parameters of the impulse model, mu is the single parameter of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by optim from stats. Each value matrix is called value_'condition' and has the form (genes x time points) and contains the counts predicted by the impulse model at the observed time points.
vecRefPval	(vec length genes) Method 2 (DESeq2) adjusted p-values
strCaseName	(str) Name of the case condition in dfAnnotationProcFull.
strFileNameSuffix	(character string) [Default ""] File extention.
strPlotSubtitle	(character string) [Default ""] Subtitle for each plot.
NPARAM	(scalar) [Default 6] Number of parameters of impulse model.
matCountDataProc:	(matrix genes x samples) Count data: Reduced version of matCountData. For internal use.
matTranslationFactors:	(numeric matrix genes x samples) Model scaling factors for each observation which take longitudinal time series mean within a gene into account (translation factors). Computed based based on all samples.
matSizeFactors:	(numeric matrix genes x samples) Model scaling factors for each observation which take sequencing depth into account (size factors). One size factor per sample - rows of this matrix are equal.
dfAnnotationProc:	(Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and LongitudinalSeries). For internal use.
dfDEAnalysis	(data frame genes x fitting characteristics) Summary of fitting procedure for each gene.
strControlName:	(str) [Default NULL] Name of the control condition in dfAnnotationProcFull.

strMode:	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.
title_string	(character string) [Default ""] Title for each plot.
NPARAM	(scalar) [Default 6] Number of parameters of impulse model.

See Also

Called by runImpulseDE2.

---

processData *Process annotation and count data*

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: (I) 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. (II) 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. reduceCountData() Reduce count data to data which are utilised later. (III) Script body

Usage

```
processData(dfAnnotation = NULL, matCountData = NULL, scaSmallRun = NULL,
  strCaseName = NULL, strControlName = NULL, strMode = NULL,
  strSCMode = NULL, scaWindowRadius = NULL, lsPseudoDE = NULL,
  vecDispersionsExternal = NULL, vecSizeFactorsExternal = NULL,
  boolRunDESeq2 = NULL)
```

Arguments

strCaseName	(str) [Default NULL] Name of the case condition in dfAnnotation.
matCountData:	(matrix genes x samples) [Default NULL] Count data of all conditions, unobserved entries are NA.
dfAnnotation:	(Table) [Default NULL] Annotation table. Lists co-variables of samples: Sample, Condition, Time (and LongitudinalSeries). Time must be numeric.
strControlName:	(str) [Default NULL] Name of the control condition in dfAnnotation.
strMode:	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

Value

(list length 3) with the following elements:

- matCountDataProc: (count matrix genes x samples) Count data: Reduced version of matCountData.
- matProbNB (probability matrix genes x samples) Probability of each observation to originate from the negative binomial component in the zero inflated negative binomial mixture model. All entries are 1 if not operating in strMode=="singlecell".

See Also

Called by runImpulseDE2.

---

runDESeq2 *Wrapper function for running DESeq2*

Description

Run DESeq2 and extract differential expression analysis results and overdispersion coefficients.

Usage

```
runDESeq2(dfAnnotationProc, matCountDataProc, nProc = 1, strCaseName = NULL,
  strControlName = NULL, strMode = "batch")
```

Arguments

matCountDataProc:	(matrix genes x samples) Count data: Reduced version of matCountData. For internal use.
dfAnnotationProc:	(Table) Processed annotation table. Lists co-variables of samples: Sample, Condition, Time (numeric), TimeCateg (str) (and Timecourse). For internal use.
nProc:	(scalar) [Default 1] Number of processes for parallelisation.
strCaseName:	(str) [Default NULL] Name of the case condition in dfAnnotation.
strControlName:	(str) [Default NULL] Name of the control condition in dfAnnotation.
strMode:	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.

Value

(list length 2) with the following elements:

- dds_dispersions (vector number of genes) Inverse of gene-wise negative binomial dispersion coefficients computed by DESeq2.
- dds_resultsTable (data frame) DESeq2 results.

See Also

Called by runImpulseDE2.

`runImpulseDE2` *Differential expression analysis using impulse models*

Description

Fits an impulse model to time course data and uses this model as a basis to detect differentially expressed genes. Differential expression is either differential expression of a gene over time within one condition or differential expression of a gene over time between two conditions (case and control). With a single condition, the alternative model is the impulse fit to the time course data and the null model is the mean fit. The mean fit models no differential behaviour over time. With two conditions, the alternative model is separate impulse fits to case and control data and the null model is an impulse fit to the combined data. Here, the impulse fit to the combined data models no differential expression between the conditions.

Usage

```
runImpulseDE2(matCountData = NULL, dfAnnotation = NULL,
  strCaseName = NULL, strControlName = NULL, strMode = "batch",
  strSCMode = "clustered", scaWindowRadius = NULL, nProc = 1,
  Q_value = 0.01, scaSmallRun = NULL, boolPlotting = TRUE,
  lsPseudoDE = NULL, vecDispersionsExternal = NULL,
  vecSizeFactorsExternal = NULL, boolRunDESeq2 = TRUE,
  boolSimplePlot = FALSE, boolLogPlot = FALSE)
```

Arguments

`matCountData:`	(matrix genes x replicates) [Default NULL] Count data of all conditions, unobserved entries are NA.
`dfAnnotation:`	(Table) [Default NULL] Lists co-variables of samples: Sample, Condition, Time (numeric), (and Timecourse).
`strCaseName:`	(str) [Default NULL] Name of the case condition in `dfAnnotation`.
`strControlName:`	(str) [Default NULL] Name of the control condition in `dfAnnotation`.
`strMode:`	(str) [Default "batch"] "batch","longitudinal","singlecell" Mode of model fitting.
`nProc:`	(scalar) [Default 1] Number of processes for parallelisation.
`Q_value:`	(scalar) [Default 0.01] FDR-corrected p-value cutoff for significance.
`scaSmallRun:`	(integer) [Default NULL] Number of rows on which ImpulseDE2 is supposed to be run, the full data set is only used for size factor estimation.
`boolPlotting:`	(bool) [TRUE] Whether to plot significant DE genes into output pdf. Consider setting FALSE for large data sets with many hits.
`lsPseudoDE:`	(list) [Defaul NULL]
`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.

`boolRunDESeq2`: (bool) [Default TRUE] Whether to run DESeq2.

`boolSimplePlot`: (bool) [Default FALSE] Whether to reduce plot to data points and impulse trace.

`boolLogPlot`: (bool) [Default FALSE] Whether to plot in counts in log space.

Details

`ImpulseDE2` is based on the impulse model proposed by Chechik and Koller (Chechik and Koller, 2009). The impulse model models the response of gene activity read outs (such as RNAseq counts) to environmental or developmental stimuli as the product of two simgoids. This model can capture simple time course patterns, such as plateaus, increase and decrease. ImpulseDE2 uses the impulse model to identify differential activity over time on any type of count data which follows the negative binomial distribution (as frequently encountered in sequencing data). ImpulseDE2 performs fitting of the impulse model and a mean model to data and evaluates the fit. The computational complexity of ImpulseDE2 is linear in the number of genes and linear in the number of samples.

1. Impulse fitting: The impulse model is fitted based on the assumption that the input count data follow a negative binomial distribution with dispersion as identified by DESeq2. The impulse model does not have a closed form maximum likelihood parameter estimate and must therefore be inferred from numerical optimisation.
   (a) Initialisation: Initialisation is performed twice for each gene, based on a peak and a valley model. The parameters representing these models reflect the form that these two models would have given the count data of each gene and are specific to each gene.
   (b) Optimisation: The cost function for the fit is the log likelihood of the data which is evaluated based on negative binomial likelihoods at each observed time point, with the value of the impulse model as the mean and the gene dispersion inferred using DESeq2 as the dispersion. Numerical optimisation is performed using the BFGS algorithm. In analogy to generalised linear models for count data, the fitting of the parameters representing limit behaviours of the two sigmoids (which are counts) are fitted in log space so that they cannot adopt negative values.
   (c) Fit selection: The fit with the higher log likelihood of the two initialisation is selected and kept as a maximum likelihood estimate.
2. Mean fitting: The mean model is a single negative binomial and serves as the null model in the case of differential expression over time within a single condition. The maximum likelihood estimate of the mean of a negative binomial distribution given the data and the dispersion is the sample average. This closed form solution is used here.
3. Fit evaluation: The model comparison statistic is the deviance 2 * (loglikelihood(H1) - loglikelihood(H0)). The deviance is chi-squared distributed with the difference in degrees of freedom of both models as degrees of freedom if the null model is contained in the alternative model, which is given in both modes of differential expression analysis with ImpulseDE (with and without control). Therefore p-values for differential expression are computed based on the chi-squared distribution. The p-values are the FDR corrected (Benjamini and Hochberg, 1995).

Value

(list length 4)

- vecDEGenes: (list number of genes) Genes IDs identified as differentially expressed by ImpulseDE2 at threshold `Q_value`.
- dfImpulseResults: (data frame) ImpulseDE2 results.

- lsImpulseFits: (list) List of matrices which contain parameter fits and model values for given time course for the case condition (and control and combined if control is present). Each parameter matrix is called parameter_'condition' and has the form (genes x [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0, converge_H0]) where beta to t2 are parameters of the impulse model, mu is the single parameter of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by `optim` from `stats` of either model. Each value matrix is called value_'condition' and has the form (genes x time points) and contains the counts predicted by the impulse model at the observed time points.
- dfDESeq2Results: (data frame) DESeq2 results. NULL if DESeq2 is not run.

Additionally, `ImpulseDE2` saves the following objects and tables into the working directory:

- `ImpulseDE2_matCountDataProc.RData` (2D array genes x replicates) Count data: Reduced version of `matCountData`. For internal use.
- `ImpulseDE2_dfAnnotationProc.RData` (data frame) Annotation table.
- `ImpulseDE2_matSizeFactors.RData` (numeric matrix genes x samples) Model scaling factors for each observation which take sequencing depth into account (size factors). One size factor per sample - rows of this matrix are equal.
- `ImpulseDE2_matTranslationFactors.RData` (numeric matrix genes x samples) Model scaling factors for each observation which take longitudinal time series mean within a gene into account (translation factors). Computed based based on all samples.
- `ImpulseDE2_vecDispersions.RData` (vector number of genes) Inverse of gene-wise negative binomial dispersion coefficients computed by DESeq2.
- `ImpulseDE2_dfDESeq2Results.RData` (data frame) DESeq2 results.
- `ImpulseDE2_lsImpulseFits.RData` (list) List of matrices which contain parameter fits and model values for given time course for the case condition (and control and combined if control is present). Each parameter matrix is called parameter_'condition' and has the form (genes x [beta, h0, h1, h2, t1, t2, logL_H1, converge_H1, mu, logL_H0, converge_H0]) where beta to t2 are parameters of the impulse model, mu is the single parameter of the mean model, logL are log likelihoods of full (H1) and reduced model (H0) respectively, converge is convergence status of numerical optimisation of model fitting by `optim` from `stats` of either model. Each value matrix is called value_'condition' and has the form (genes x time points) and contains the counts predicted by the impulse model at the observed time points.
- `ImpulseDE2_dfImpulseResults.RData` (data frame) ImpulseDE2 results.
- `ImpulseDE2_vecDEGenes.RData` (list number of genes) Genes IDs identified as differentially expressed by ImpulseDE2 at threshold `Q_value`.
- `ImpulseDE2_ClusterOut.txt` Text-file with stdout and stderr from cluster created in `fitImpulse`.

Author(s)

David Sebastian Fischer

See Also

Calls the following functions: `processData`, `runDESeq2`, `fitImpulse`, `computePval`, `plotDEGenes`.

Index