Package ‘NPARC’

May 18, 2024

Type Package

Title Non-parametric analysis of response curves for thermal proteome profiling experiments

Version 1.16.0

Author Dorothee Childs, Nils Kurzawa

Maintainer Nils Kurzawa <nilskurzawa@gmail.com>


License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Depends R (>= 4.0.0)

Imports dplyr, tidyr, BiocParallel, broom, MASS, rlang, magrittr, stats, methods

Suggests testthat, devtools, knitr, rprojroot, rmarkdown, ggplot2, BiocStyle

VignetteBuilder knitr

biocViews Software, Proteomics

git_url https://git.bioconductor.org/packages/NPARC

git_branch RELEASE_3_19

git_last_commit c3850f8

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-17
### fitSingleSigmoid

**Description**
Fit sigmoid model

**Usage**

```r
fitSingleSigmoid(x, y, start = c(Pl = 0, a = 550, b = 10))
```

**Arguments**
- `x`: numeric vector of the independent variables (typically temperature)
- `y`: numeric vector of the dependent variables (typically relative abundance measurements)
- `start`: numeric vector of start parameters for the melting curve equation

**Details**
Fits the following function to the data:

\[
y = \frac{1 - Pl}{1 + exp((b - a/x))} + Pl
\]

**Value**
model summary of type "nls"

**Examples**

```r
data(stauro_TPP_data_tidy)
stk4 <- dplyr::filter(stauro_TPP_data_tidy, grepl("STK4", uniqueID))
fitSingleSigmoid(stk4$temperature, stk4$relAbundance)
```
getParams

Description

Control parameters for model fitting

Usage

getParams(start = c(P1 = 0, a = 550, b = 10), maxAttempts = 100)

Arguments

start Numeric vector of start parameters for the melting curve equation
maxAttempts Number of resampling steps in case of unsuccessful model fits

Value

list of two elements: 1) "start" listing the starting parameters for melting curve fitting, 2) "maxAttempts" listing the maximal number of attempts the fit should be allowed

Examples

data(stauro_TPP_data_tidy)
df <- dplyr::filter(stauro_TPP_data_tidy, grepl("MAPK|ATP|CDK|GTP|CRK", uniqueID))
testResults <- runNPARC(x = df$temperature,
y = df$relAbundance,
id = df$uniqueID,
groupsAlt = df$compoundConcentration,
dfType = "empirical",
control = getParams(maxAttempts = 50))

NPARC

NPARC package

Description

Non-parametric analysis of response curves

Details

See the preprint on Childs, Bach, Franken et al. (2019): Non-parametric analysis of thermal proteome profiles reveals novel drug-binding proteins
NPARCfit | Fit null and alternative models for Non-parametric analysis of response curves

Description
Fit melting curve and return model metrics as well as predictions for the null and alternative models.

Usage
```r
NPARCfit(x, y, id, control = getParams(), groupsNull = NULL, groupsAlt, BPPARAM = BiocParallel::SerialParam(progressbar = TRUE), returnModels = FALSE)
```

Arguments
- **x**: numeric vector of the independent variables (typically temperature)
- **y**: numeric vector of the dependent variables (typically relative abundance measurements)
- **id**: character vector with the protein ID to which each data point belongs.
- **control**: list of parameters used to control specific parts of the analysis
- **groupsNull**: one or more vectors with grouping variables for the null models. See details.
- **groupsAlt**: one or more vectors with grouping variables for the alternative models. See details.
- **BPPARAM**: BiocParallel parameter object to invoke curve fitting in parallel. Default: BiocParallel::SerialParam()
- **returnModels**: boolean value. If true, the fitted models are returned together with the test results

Details
- **groupsNull** or **groupsAlt** can either be a single vector each, or data.frames of the same length as **x** and **y** with one column per factor

Value
data frame with fitted model parameters and additional columns listing e.g. residuals sum of squares

Examples
```r
data(stauro_TPP_data_tidy)
df <- dplyr::filter(stauro_TPP_data_tidy, grepl("CDK\|GTP\|CRK", uniqueID))
testResults <- NPARCfit(x = df$temperature,
                         y = df$relAbundance,
                         id = df$uniqueID,
                         groupsAlt = df$compoundConcentration)
```
**NPARCtest**

*Perform F-test*

**Description**

Perform F-test

**Usage**

```
NPARCtest(modelMetrics, dfType = c("empirical", "theoretical"))
```

**Arguments**

- **modelMetrics**: data.frame with results of the model fit in long format.
- **dfType**: character value indicating the method for degrees of freedom computation for the F-test. Theoretical yields the text-book solution. Empirical yields estimates derived from the distribution moments of the RSS.

**Value**

data frame with fitted model parameters and additional columns listing e.g. residuals sum of squares of null and alternative model and raw and adjusted p values retrieved from testing

**Examples**

```r
data(stauro_TPP_data_tidy)
df <- dplyr::filter(stauro_TPP_data_tidy, grepl("CDK\|GTP\|CRK", uniqueID))
fits <- NPARCfit(x = df$temperature,
                 y = df$relAbundance,
                 id = df$uniqueID,
                 groupsNull = NULL,
                 groupsAlt = df$compoundConcentration,
                 returnModels = FALSE)
modelMetrics <- fits$metrics
testRes <- NPARCtest(modelMetrics, dfType = "theoretical")
```

**runNPARC**

*Non-parametric analysis of response curves*

**Description**

Wrapper function for melting curve fitting and hypothesis testing.

**Usage**

```
runNPARC(x, y, id, groupsNull = NULL, groupsAlt,
         BPPARAM = BiocParallel::SerialParam(progressbar = TRUE),
         dfType = c("theoretical", "empirical"), control = getParams()))
```
Arguments

- **x**
  - numeric vector of the independent variables (typically temperature)
- **y**
  - numeric vector of the dependent variables (typically relative abundance measurements)
- **id**
  - character vector with the protein ID to which each data point belongs.
- **groupsNull**
  - one or more vectors with grouping variables for the null models. See details.
- **groupsAlt**
  - one or more vectors with grouping variables for the alternative models. See details.
- **BPPARAM**
  - BiocParallel parameter object to invoke curve fitting in parallel. Default: BiocParallel::SerialParam()
- **dfType**
  - character value indicating the method for degrees of freedom computation for the F-test. Theoretical yields the text-book solution. Empirical yields estimates derived from the distribution moments of the RSS.
- **control**
  - list of parameters used to control specific parts of the analyse

Details

groupsNull or groupsAlt can either be a single vector each, or data.frames of the same length as x and y with one column per factor

Value

data frame with fitted model parameters and additional columns listing e.g. residuals sum of squares of null and alternative model

Examples

data(stauro_TPP_data_tidy)
df <- dplyr::filter(stauro_TPP_data_tidy, grepl("CDK|GTP|CRK", uniqueID))
testResults <- runNPARC(x = df$temperature,
y = df$relAbundance,
id = df$uniqueID,
groupsAlt = df$compoundConcentration,
dfType = "empirical")

stauro_TPP_data_tidy  TPP dataset of staurosporine treated cells.

Description

Data from a thermal proteome profiling (TPP) experiment investigating the ATP-competitive pan-kinase inhibitor staurosporine on K562 cells. The data has been downloaded the data from the supplement of the respective publication and converted into tidy format.
Usage

data(stauro_TPP_data_tidy)

Format

An object of class "data.frame"

References

Index

* datasets
  stauro_TPP_data_tidy, 6

fitSingleSigmoid, 2

getParams, 3

NPARC, 3
NPARC-package (NPARC), 3
NPARCFit, 4
NPARCTest, 5

runNPARC, 5

stauro_TPP_data_tidy, 6