Package: RPPASPACE (via r-universe)

July 20, 2024

Version 1.0.10

Date 2023-10-19

Title Reverse-Phase Protein Array Super Position and Concentration Evaluation

Maintainer James M. Melott <jmmelott@mdanderson.org>

Description Provides tools for the analysis of reverse-phase protein arrays (RPPAs), which are also known as 'tissue lysate arrays' or simply 'lysate arrays'. The package's primary purpose is to input a set of quantification files representing dilution series of samples and control points taken from scanned RPPA slides and determine a relative log concentration value for each valid dilution series present in each slide and provide graphical visualization of the input and output data and their relationships. Other optional features include generation of quality control scores for judging the quality of the input data, spatial adjustment of sample points based on controls added to the slides, and various types of normalization of calculated values across a set of slides. The package was derived from a previous package named SuperCurve. For a detailed description of data inputs and outputs, usage information, and a list of related papers describing methods used in the package please review the vignette 'Guide_to_RPPASPACE'. 'RPPA SPACE: an R package for normalization and quantitation of Reverse-Phase Protein Array data'. Bioinformatics Nov 15;38(22):5131-5133. <doi:10.1093/bioinformatics/btac665>.

Depends R (>= 3.5.0), methods, doParallel, foreach, parallel, iterators

Imports utils, grDevices, graphics, stats, bmp, jpeg, tiff, png, imager, cobs, splines, nlme, robustbase, mgcv, SparseM, quantreg, timeDate

Suggests boot, knitr, rmarkdown

NeedsCompilation no

VignetteBuilder knitr, rmarkdown

URL https://pubmed.ncbi.nlm.nih.gov/36205581,
https://github.com/MD-Anderson-Bioinformatics/rppaspace,
https://bioinformatics.mdanderson.org/public-software/rppaspace/

License Artistic-2.0

Copyright file inst/COPYRIGHTS

LazyLoad yes

Encoding UTF-8

RoxygenNote 7.2.3

Repository https://md-anderson-bioinformatics.r-universe.dev

RemoteUrl https://github.com/MD-Anderson-Bioinformatics/rppaspace

RemoteRef HEAD

RemoteSha b2d8972ec0801df5ddda3227ec66d2dcd80877f0

Contents

RPPASPACE-package *Reverse phase protein lysate array analysis*

Description

A package for analyzing reverse phase protein lysate arrays (RPPA).

Details

Package:	RPPASPACE
Type:	Package
Version:	1.0.10
Phase:	
Date:	2023-10-19
License:	Artistic-2.0

For a complete list of functions, use `library(help="RPPASPACE")`.
For a high-level summary of the changes for each revision, use
`file.show(system.file("NEWS", package="RPPASPACE"))`.

Author(s)

Kevin R. Coombes <coombes.3@osu.edu>, P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

CobsFitClass-class *Class "CobsFitClass"*

Description

The `CobsFitClass` class represents models that were fit with the nonparametric model.

Usage

```
## S4 method for signature 'CobsFitClass'
fitSeries(object,
          diln,
          intensity,
          est.conc,
          method="nls",
          silent=TRUE,
          trace=FALSE,
          ...)
## S4 method for signature 'CobsFitClass'
```

```
fitSlide(object,
         conc,
         intensity,
         ...)
## S4 method for signature 'CobsFitClass'
fitted(object, conc, ...)
## S4 method for signature 'CobsFitClass'
trimConc(object,
         conc,
         intensity,
         design,
         trimLevel,
         ...)
```

Arguments

object	object of class CobsFitClass
diln	numeric vector of dilutions for series to be fit
intensity	numeric vector of observed intensities for series to be fit
est.conc	numeric estimated concentration for EC50 dilution
method	character string specifying regression method to use to fit the series
silent	logical scalar. If TRUE, report of error messages will be suppressed in try(nlsmeth(...))
trace	logical scalar. Used in nls method.
conc	numeric vector containing estimates of the log concentration for each dilution series
design	object of class RPPADesignParams describing options for processing the array
trimLevel	numeric scalar multiplied to Median Absolute Deviation MAD
...	extra arguments for generic routines

Value

The fitted method returns a numeric vector.

Objects from the Class

Objects are created internally by calls to the methods fitSlide or RPPAFit.

Slots

model: object of class cobs summarizing nonparametric fit

lambda: numeric

Extends

Class FitClass, directly.

Methods

fitSeries `signature(object = "CobsFitClass")`:
Finds the concentration for an individual dilution series given the curve fit for the slide.

fitSlide `signature(object = "CobsFitClass")`:
Uses the concentration and intensity series for an entire slide to fit a curve for the slide of intensity = f(conc).

fitted `signature(object = "CobsFitClass")`:
Extracts fitted values of the model.

trimConc `signature(object = "CobsFitClass")`:
Returns concentration and intensity cutoffs for the model.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

FitClass

Directory-class *Class "Directory"*

Description

The Directory class represents a file system directory.

Usage

```
Directory(path)
is.Directory(x)
## S4 method for signature 'character,Directory'
coerce(from, to, strict=TRUE)
## S4 method for signature 'Directory,character'
coerce(from, to, strict=TRUE)
```

Arguments

path	character string specifying a directory
x	object of class Directory
from	object of class Directory or character string specifying pathname of directory
to	object of class Directory or character string specifying pathname of directory
strict	logical scalar. If TRUE, the returned object must be strictly from the target class; otherwise, any simple extension of the target class will be returned, without further change.

Value

The `Directory` generator returns an object of class `Directory`.

The `is.Directory` method returns `TRUE` if its argument is an object of class `Directory`.

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the `Directory` generator function.

Slots

`path`**:** character string specifying a directory

Methods

coerce `signature(from = "Directory", to = "character")`:
Coerce an object of class `Directory` to its character string pathname equivalent.

coerce `signature(from = "character", to = "Directory")`:
Coerce a character string specifying directory pathname to an equivalent object of class `Directory`.

Note

The coercion methods should not be called explicitly; instead, use an explicit call to the `as` method.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

`as`

`DS5RPPAPreFitQC-class` *Class "DS5RPPAPreFitQC"*

Description

The DS5RPPAPreFitQC class represents the inputs necessary to determine the quality control rating of a reverse-phase protein array slide with 5 dilution series.

Usage

```
## S4 method for signature 'DS5RPPAPreFitQC'
qcprob(object, ...)
## S4 method for signature 'DS5RPPAPreFitQC'
summary(object, ...)
```

Arguments

object	object of class DS5RPPAPreFitQC
...	extra arguments for generic routines

Details

The prediction model used multiple training datasets from the RPPA Core Facility by fitting a logistic regression model using an expert rating of a slide's quality (good, fair, or poor) as the response variable and a host of metrics about the raw positive control data as predicting variables.

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the RPPAPreFitQC factory generator function.

Slots

antibody: character string specifying name of antibody

slopediff: numeric scalar specifying the difference from perfect slope

cvs: numeric vector containing the coefficient of variance for each positive control dilution series

slopes: numeric vector containing the slopes for each positive control dilution series

drdiffs: numeric vector containing the difference in dynamic range of each positive control dilution series

percentgood: numeric scalar specifying percentage of "good" sample spots on the slide

adjusted: logical scalar specifying if adjusted measures were used

Extends

Class RPPAPreFitQC, directly.

Methods

qcprob `signature(object = "DS5RPPAPreFitQC")`:
Calculates the probability of good slide, returned as numeric scalar.

summary `signature(object = "DS5RPPAPreFitQC")`:
Prints a summary of the underlying data frame.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

References

Ju Z, Liu W, Roebuck PL, Siwak DR, Zhang N, Lu Y, Davies MA, Akbani R, Weinstein JN, Mills GB, Coombes KR
Development of a Robust Classifier for Quality Control of Reverse Phase Protein Arrays.
Bioinformatics (2015) 31(6): 912-918.
https://pubmed.ncbi.nlm.nih.gov/25380958/

FitClass-class *Class "FitClass"*

Description

The FitClass class is a virtual class representing the model that was fit in the RPPAFit routine. Functions for use with FitClass are only to be used internally.

Usage

```
is.FitClass(x)
## S4 method for signature 'FitClass'
coef(object, ...)
## S4 method for signature 'FitClass'
coefficients(object, ...)
## S4 method for signature 'FitClass'
fitSeries(object,
          diln,
          intensity,
          est.conc,
          method="nls",
          silent=TRUE,
          trace=FALSE,
          ...)
## S4 method for signature 'FitClass'
fitSlide(object,
         conc,
         intensity,
         ...)
## S4 method for signature 'FitClass'
fitted(object,
       conc,
       ...)
## S4 method for signature 'FitClass'
trimConc(object,
         conc,
         intensity,
         design,
         trimLevel,
         ...)
```

Arguments

x	object of (sub)class FitClass
object	object of (sub)class FitClass
diln	numeric vector of dilutions for series to be fit

intensity	numeric vector of observed intensities for series to be fit
est.conc	numeric estimated concentration for EC50 dilution
method	character string specifying regression method to use to fit the series
silent	logical scalar. If TRUE, report of error messages will be suppressed in try(nlsmeth(...))
trace	logical scalar. Used in nls method.
conc	numeric vector containing current estimates of concentration for each series
design	object of class RPPADesignParams describing options for processing the array
trimLevel	numeric scalar multiplied to Median Absolute Deviation MAD
...	extra arguments for generic routines

Value

The is.FitClass method returns TRUE if its argument is an object of subclass of class FitClass.

The coef and coefficients methods return NULL.

Objects from the Class

This class should not be instantiated directly; extend this class instead.

Methods

coef signature(object = "FitClass"):
Placeholder method which should be implemented by subclass if appropriate for the particular model.

coefficients signature(object = "FitClass"):
An alias for coef.

fitSeries signature(object = "FitClass"):
Placeholder method which must be implemented by subclass.

fitSlide signature(object = "FitClass"):
Placeholder method which must be implemented by subclass.

fitted signature(object = "FitClass"):
Placeholder method which must be implemented by subclass.

trimConc signature(object = "FitClass"):
Placeholder method which must be implemented by subclass.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

`getConfidenceInterval` *Compute Confidence Intervals for a Model Fit to Dilution Series*

Description

This function computes confidence intervals for the estimated concentrations in a four-parameter logistic model fit to a set of dilution series in a reverse-phase protein array experiment.

Usage

```
getConfidenceInterval(result,
                      alpha=0.1,
                      nSim=50,
                      progmethod=NULL)
```

Arguments

`result`	object of class `RPPAFit` representing the result of fitting a four-parameter logistic model
`alpha`	numeric scalar specifying desired significance of the confidence interval; the width of the resulting interval is 1 - alpha.
`nSim`	numeric scalar specifying number of times to resample the data in order to estimate the confidence intervals.
`progmethod`	optional function that can be used to report progress.

Details

In order to compute the confidence intervals, the function assumes that the errors in the observed Y intensities are independent normal values, with mean centered on the estimated curve and standard deviation that varies smoothly as a function of the (log) concentration. The smooth function is estimated using `loess`. The residuals are resampled from this estimate and the model is refit; the confidence intervals are computed empirically as symmetrically defined quantiles of the refit parameter sets.

Value

An object of class `RPPAFit`, containing updated values for the slots `lower`, `upper`, and `conf.width` that describe the confidence interval.

Author(s)

Kevin R. Coombes <coombes.3@osu.edu>, P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

`RPPAFit-class`, `RPPAFit`

LoessFitClass-class *Class "LoessFitClass"*

Description

The LoessFitClass class represents models that were fit with the nonparametric model.

Usage

```
## S4 method for signature 'LoessFitClass'
fitSeries(object,
          diln,
          intensity,
          est.conc,
          method="nls",
          silent=TRUE,
          trace=FALSE,
          ...)
## S4 method for signature 'LoessFitClass'
fitSlide(object,
         conc,
         intensity,
         ...)
## S4 method for signature 'LoessFitClass'
fitted(object,
       conc,
       ...)
## S4 method for signature 'LoessFitClass'
trimConc(object,
         conc,
         intensity,
         design,
         trimLevel,
         ...)
```

Arguments

object	object of class LoessFitClass
diln	numeric vector of dilutions for series to be fit
intensity	numeric vector of observed intensities for series to be fit
est.conc	numeric estimated concentration for EC50 dilution
method	character string specifying regression method to use to fit the series
silent	logical scalar. If TRUE, report of error messages will be suppressed in try(nlsmeth(...))
trace	logical scalar. Used in nls method.

conc	numeric vector containing estimates of the log concentration for each dilution series
design	object of class RPPADesignParams describing options for processing the array
trimLevel	numeric scalar multiplied to Median Absolute Deviation MAD
...	extra arguments for generic routines

Value

The fitted method returns a numeric vector.

Objects from the Class

Objects are created internally by calls to the methods fitSlide or RPPAFit.

Slots

model: object of class loess summarizing loess fit

Extends

Class FitClass, directly.

Methods

fitSeries signature(object = "LoessFitClass"):
Finds the concentration for an individual dilution series given the curve fit for the slide.

fitSlide signature(object = "LoessFitClass"):
Uses the concentration and intensity series for an entire slide to fit a curve for the slide of intensity = f(conc).

fitted signature(object = "LoessFitClass"):
Extracts fitted values of the model.

trimConc signature(object = "LoessFitClass"):
Returns concentration and intensity cutoffs for the model.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

FitClass

LogisticFitClass-class
Class "LogisticFitClass"

Description

The LogisticFitClass class represents models that were fit with the logistic model.

Usage

```
## S4 method for signature 'LogisticFitClass'
coef(object, ...)
## S4 method for signature 'LogisticFitClass'
coefficients(object, ...)
## S4 method for signature 'LogisticFitClass'
fitSeries(object,
          diln,
          intensity,
          est.conc,
          method="nls",
          silent=TRUE,
          trace=FALSE,
          ...)
## S4 method for signature 'LogisticFitClass'
fitSlide(object,
         conc,
         intensity,
         ...)
## S4 method for signature 'LogisticFitClass'
fitted(object,
       conc,
       ...)
## S4 method for signature 'LogisticFitClass'
trimConc(object,
         conc,
         intensity,
         design,
         trimLevel,
         ...)
```

Arguments

object	object of class LogisticFitClass
diln	numeric vector of dilutions for series to be fit
intensity	numeric vector of observed intensities for series to be fit
est.conc	numeric estimated concentration for EC50 dilution

method	character string specifying regression method to use to fit the series
silent	logical scalar. If TRUE, report of error messages will be suppressed in try(nlsmeth(...))
trace	logical scalar. Used in nls method.
conc	numeric vector containing estimates of the log concentration for each dilution series
design	object of class RPPADesignParams describing options for processing the array
trimLevel	numeric scalar multiplied to Median Absolute Deviation MAD
...	extra arguments for generic routines

Value

The coef and coefficients methods return a named vector of length three with logistic curve coefficients.

The fitted method returns a numeric vector.

Objects from the Class

Objects are created internally by calls to the methods fitSlide or RPPAFit.

Slots

coefficients: numeric vector of length 3, representing alpha, beta, and gamma respectively.

Extends

Class FitClass, directly.

Methods

coef signature(object = "LogisticFitClass"):
Extracts model coefficients from objects returned by modeling functions.

coefficients signature(object = "LogisticFitClass"):
An alias for coef

fitSeries signature(object = "LogisticFitClass"):
Finds the concentration for an individual dilution series given the curve fit for the slide.

fitSlide signature(object = "LogisticFitClass"):
Uses the concentration and intensity series for an entire slide to fit a curve for the slide of intensity = f(conc).

fitted signature(object = "LogisticFitClass"):
Extracts fitted values of the model.

trimConc signature(object = "LogisticFitClass"):
Returns concentration and intensity cutoffs for the model.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

FitClass

`normalize` *Normalization*

Description

This function performs normalization for sample loading after quantification. It is typically invoked as part of the process of creating summary information from an `RPPASet` object.

Usage

```
## S4 method for signature 'MatrixLike'
normalize(object,
          method=getRegisteredNormalizationMethodKeys(),
          calc.medians=TRUE,
          sweep.cols=calc.medians,
          ...)
```

Arguments

`object`	data frame or matrix to be normalized
`method`	character string specifying name of method of sample loading normalization (see section 'Details' below)
`calc.medians`	logical scalar. If `TRUE`, calculate row and column median values from the data to be normalized.
`sweep.cols`	logical scalar. If `TRUE`, subtract column medians from data values prior to invoking the normalization method.
`...`	extra arguments for normalization routines

Details

By default, column medians are subtracted from the input data values; these adjusted data values are then passed to the requested normalization routine for further processing.

The `method` argument may be augmented with user-provided normalization methods. Package-provided values are:

medpolish	Tukey's median polish normalization
median	sample median normalization
house	housekeeping normalization
vs	variable slope normalization
none	no normalization done

Specifying "median" as the method argument causes the row median to be subtracted from each sample. Specifying "house" causes the median of one or more housekeeping antibodies to be used. The names of the antibodies to be used must be supplied as a named argument to this method. Specifying "vs" causes the sample median to be used along with a multiplicative gamma (see reference below).

Value

Returns normalized concentrations as matrix appropriately annotated.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, E. Shannon Neeley <sneeley@stat.byu.edu>, James M. Melott <jmmelott@mdanderson.org>

See Also

RPPASet

---

normalize-method *Method "normalize"*

Description

normalize is a generic function used to normalize the data based on the input object. The method invokes particular methods which depend on the class of the first argument.

Usage

```
## S4 method for signature 'ANY'
normalize(object, ...)
## S4 method for signature 'NULL'
normalize(object, ...)
```

Arguments

object	an object to be normalized
...	additional arguments affecting the normalization process

Value

The form of the value returned by normalize depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

If the object is NULL, NA is returned.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>

qcprob-method *Method "qcprob"*

Description

qcprob is a generic function used to produce a quality control probability based on the input object. The method invokes particular methods which depend on the class of the first argument.

Usage

```
## S4 method for signature 'ANY'
qcprob(object, ...)
## S4 method for signature 'NULL'
qcprob(object, ...)
```

Arguments

object	an object for which a QC probability is desired
...	additional arguments affecting the QC probability produced

Value

The form of the value returned by qcprob depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

If the object is NULL, NA is returned.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

registerModel *Model Registration Methods*

Description

These routines represent the high-level access for model registration, which enables data-driven access by other routines. This represents the initial implementation and may change in the future.

Usage

```
getRegisteredModel(key)
getRegisteredModelLabel(key)
getRegisteredModelKeys()
registerModel(key, classname, ui.label=names(key))
```

Arguments

key	character string representing a registered model
classname	character string specifying Model class name to register
ui.label	character string specifying label to display by UI

Value

getRegisteredModel returns the classname associated with key.

getRegisteredModelLabel returns the ui.label associated with key.

getRegisteredModelKeys returns vector of keys for all registered models.

registerModel is invoked for its side effect, which is registering classname and ui.label by association to key.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

getRegisteredObject, getRegisteredObjectKeys, registerClassname

Examples

```
## Create new (but nonfunctional) fit model
## Not run due to lack of capability to unregister class
  ## Not run:
setClass("TestFitClass",
         representation("FitClass",
                        testfit="character"),
         prototype(testfit="TestFitClass"))

## Register fit model to enable its use by package
registerModel("testfit", "TestFitClass", "Registered Test Fit Class")

## Show all registered fit models
sapply(getRegisteredModelKeys(),
       function(key) {
           c(model=getRegisteredModel(key),
             label=getRegisteredModelLabel(key))
       })

## End(Not run)
```

registerNormalizationMethod
Normalization Method Registration Methods

Description

These routines represent the high-level access for normalization method registration, which enables data-driven access by other routines. This represents the initial implementation and may change in the future.

Usage

```
getRegisteredNormalizationMethod(key)
getRegisteredNormalizationMethodLabel(key)
getRegisteredNormalizationMethodKeys()
registerNormalizationMethod(key, method, ui.label=names(key))
```

Arguments

key	character string representing a registered normalization method
method	function to invoke for normalization
ui.label	character string specifying label to display by UI

Value

getRegisteredNormalizationMethod returns the method associated with key.

getRegisteredNormalizationMethodLabel returns the ui.label associated with key.

getRegisteredNormalizationMethodKeys returns vector of keys for all registered normalization methods.

registerNormalizationMethod is invoked for its side effect, which is registering method and ui.label by association to key.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

getRegisteredObject, getRegisteredObjectKeys, registerMethod

Examples

```
  ## Not run:
## Not run due to lack of capability to unregister methods
## Create new normalization method
normalize.testNorm <- function(concs, bar) {
    return(normconcs <- concs - bar)
}

## Register normalization method to enable its use by package
registerNormalizationMethod("testNorm", normalize.testNorm, "Registered Test Normalization Class")

## Use it...
concs <- matrix(runif(500), nrow=10)
normalize(concs, method="testNorm", bar=0.005)

## Show all registered fit models
sapply(getRegisteredNormalizationMethodKeys(),
       function(key) {
           c(key = key,
             label=getRegisteredNormalizationMethodLabel(key))
       })

## End(Not run)
```

RPPA-class *Class "RPPA"*

Description

The RPPA class represents the raw quantification data from a reverse-phase protein array experiment.

Usage

```
RPPA(file,
     path=".",
     slideNumber=NA,
     antibody=NULL,
     tracking=NULL,
     seriesToIgnore=NULL,
  warningsFileName="warnings.txt"
  )
is.RPPA(x)
## S4 method for signature 'RPPA'
dim(x)
## S4 method for signature 'RPPA'
image(x, measure="Net.Value",
     main = .mkPlotTitle(measure,
```

```
                x@antibody),
     colorbar=FALSE, col=terrain.colors(256), ...)
## S4 method for signature 'RPPA'
summary(object, ...)
seriesNames(rppa)
seriesToUseToMakeCurve(rppa)
```

Arguments

file	character string or connection specifying text file containing quantifications of a reverse-phase protein array experiment
path	character string specifying the path from the current directory to the file. The default value assumes the file is contained in the current directory. If file is a connection, this argument is ignored.
antibody	character string specifying antibody name. If missing, default value is filename (referenced by file argument) without extension.
slideNumber	integer containing the index of the slide currently being processed.
warningsFileName	character string holding the name of the file to which to write out warning messages generated during processing.
tracking	data.frame used to track the points data from a slide and how they are used. (see section 'Tracking' below)
seriesToIgnore	Comma separated list of series names to ignore. These series will not be used to calculate the curve used to fit data. Names in list must match series names in sample file.
object	object of class RPPA
x	object of class RPPA
measure	character string containing the name of the measurement column in data that should be displayed by the image method
main	character string used to title the image plot
colorbar	logical scalar that determines whether to include a color bar in the plot. If TRUE, the image cannot be used as one panel in a window with multiple plots. Default is FALSE.
col	graphics parameter used by image.
...	extra arguments for generic or plotting routines
rppa	object of class RPPA

Details

The data frame slot (data) in a valid RPPA object constructed from a quantification file using the RPPA generator function is guaranteed to contain at least 14 columns of information:

Order	Spot number order in file
Main.Row	logical location of spot on the array

Main.Col	logical location of spot on the array
Sub.Row	logical location of spot on the array
Sub.Col	logical location of spot on the array
Series.Id	unique numeric identifier of sample spotted at location
Spot.Type	type of spot at location
Dilution	measurement representing background-corrected mean intensity of the spot
Net.Value	measurement representing background-corrected mean intensity of the spot
Raw.Value	measurement representing mean intensity of the spot
Background.Value	measurement representing mean background intensity of the spot
Spot.X.Position	X location of spot on graphic image
Spot.Y.Position	Y location of spot on graphic image
Original.Order	Spot number order in original input file

Taken together, the four components (Main.Row, Sub.Row, Main.Col, Sub.Col) give the logical location of aspot on an array. Additional columns may be included.

Value

The RPPA generator returns an object of class RPPA.

The is.RPPA method returns TRUE if its argument is an object of class RPPA.

The dim method returns a numeric vector of length 4.

The image method invisibly returns the RPPA object on which it was invoked.

The summary method returns a summary of the underlying data frame.

The seriesNames function returns a character vector containing the names of the unique (non-control) dilution series on the array.

The seriesToUseToMakeCurve function returns a character vector containing the names of the unique (non-control) dilution series on the array that are used to create a curve to fit samples to.

Tracking

An object for tracking how points in the slide are to be used in the process. The information comes from the sample file of the first slide that has a valid layout. The layout of all other slides are compared to this and skipped if they don't have an identical layout.

Slots

spotType	Spot.Type according to design file.
isNegCtrl	Is point a Negative Control Point.
isPosCtrl	Is point a Positive Control Point.
isCtrl	Is point a Control Point.
applySpatialCorrection	Apply spatial correction to point
makePartOfCurve	Should point be used to create curve to which to fit data?
fitToCurve	Should point be fit to curve?
isNoise	Should point be used in noise calculations
isSample	Is point a sample point?
badPoint	Does this point have a value that was not used or caused problems in processing and whos outpu

dilution	Dilution value for this point. (Decimal value from slide).

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the RPPA generator function.

Slots

data data.frame containing the contents of a quantification file

file character string specifying the name of the file that the data was loaded from

slideNumber: integer containing the index of the slide currently being processed.

antibody character string specifying name of antibody

tracking data.frame used to track the points data from a slide and how they are used. (see section 'Tracking' below)

seriesToIgnore NULL or Comma separated list of series names to ignore. These series will not be used to calculate the curve used to fit data. Names in list must match series names in sample file.

warningsFileName character string holding the name of the file to which to write out warning messages generated during processing.

Methods

dim signature(x = "RPPA"):
Returns the dimensions of the slide layout.

image signature(x = "RPPA"):
Produces a "geographic" image of the measurement column named by the measure argument. The colors in the image represent the intensity of the measurement at each spot on the array, and the display locations match the row and column locations of the spot. Any measurement column can be displayed using this function. An optional color bar can be added, placed along the right edge.

summary signature(object = "RPPA"):
Prints a summary of the underlying data frame.

Author(s)

Kevin R. Coombes <coombes.3@osu.edu>, P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

RPPADesignParams, RPPAFit

RPPADesignParams-class

Class "RPPADesignParams"

Description

The RPPADesignParams class is used to bundle the design parameter set together for easier re-use.

Usage

```
RPPADesignParams(
 center = FALSE,
 seriesToIgnore = NULL,
 majorXDivisions=as.integer(NA),
 majorYDivisions=as.integer(NA))

is.RPPADesignParams(x)
## S4 method for signature 'RPPADesignParams'
paramString(object, slots, ...)
## S4 method for signature 'RPPA'
plot(x, measure, main, ...)
```

Arguments

center	logical scalar. If TRUE, then dilution steps are centered around 0.
x	object of class RPPADesignParams (or RPPA in plot method)
seriesToIgnore	object of class list or NULL where list members are numeric Series.Id values to ignore when fitting the data to a curve.
majorXDivisions	integer to describe distance between grid lines on the X axis of the R2 residuals plot. Defaults to 10 if NA or invalid value provided.
majorYDivisions	integer to describe distance between grid lines on the Y axis of the R2 residuals plot. Defaults to 10 if NA or invalid value provided.
object	object of class RPPADesignParams in paramString method
slots	strings specifying RPPADesignParams slotnames to display (for debugging)
main	overall title for plot
measure	character string specifying measure to plot
...	extra arguments for generic or plotting routines

Details

Allows control of some specific controls for how RPPA slides are processed.

Value

The RPPADesignParams generator returns an object of class RPPADesignParams.

The is.RPPADesignParams method returns TRUE if its argument is an object of class RPPADesignParams.

The paramString method returns a character vector, possibly empty but never NULL.

Objects from the Class

Although objects of these classes can be created by a direct call to new, the preferred method is to start with the RPPADesignParams generator, followed by the RPPADesignFromParams function to construct the final object (the RPPADesign generator is directly implemented in this way).

Slots

For RPPADesignParams class:

center: see corresponding argument above

seriesToIgnore: see corresponding argument above

majorXDivisions: see corresponding argument above

majorYDivisions: see corresponding argument above

Methods

paramString signature(object = "RPPADesignParams"):
Returns string representation of object.

Warning

The paramString method should not be called by user except for informational purposes. The content and format of the returned string may vary between different versions of this package.

Author(s)

Kevin R. Coombes <coombes.3@osu.edu>, P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

RPPA

Examples

```
showClass("RPPADesignParams")
designparams <- designparams <- RPPADesignParams(center=FALSE,
         seriesToIgnore=list(),
         majorXDivisions = as.integer(11),
         majorYDivisions = as.integer(11)
         )
paramString(designparams)
```

RPPAFit-class *Class "RPPAFit"*

Description

Objects of the RPPAFit class represent the results of fitting a statistical model of response to the dilution series in a reverse-phase protein array experiment.

Usage

```
## S4 method for signature 'RPPAFit'
coef(object, ...)
## S4 method for signature 'RPPAFit'
coefficients(object, ...)
## S4 method for signature 'RPPAFit'
fitted(object,
       type=c("Y", "y", "X", "x"),
       ...)
## S4 method for signature 'RPPAFit'
hist(x,
     type=c("Residuals", "StdRes", "ResidualsR2"),
     xlab=NULL,
     main=NULL,
     ...)
## S4 method for signature 'RPPAFit'
image(x,
      measure=c("Residuals", "ResidualsR2", "StdRes", "X", "Y"),
      main,
      ...)
## S4 method for signature 'RPPAFit,missing'
plot(x, y,
     type=c("cloud", "series", "individual", "steps", "resid"),
     col=NULL,
     main,
     xform=NULL,
     xlab="Log Concentration",
     ylab="Intensity",
     ...)
## S4 method for signature 'RPPAFit'
resid(object,
      type=c("raw", "standardized", "r2"),
      ...)
## S4 method for signature 'RPPAFit'
residuals(object,
          type=c("raw", "standardized", "r2"),
          ...)
## S4 method for signature 'RPPAFit'
```

```
summary(object, ...)
```

Arguments

object	object of class RPPAFit
x	object of class RPPAFit
type	character string describing the type of fitted values, residuals, images, histograms, or plots
measure	character string specifying measure to compute from fit
xlab	graphics parameter specifying how the x-axis should be labeled
ylab	graphics parameter specifying how the y-axis should be labeled
main	character string specifying title for the plot
xform	function to transform the raw data associated with the measure for the plot. If NULL, no transformation occurs.
y	not used
col	graphics parameter, used only if type='series', to color the lines connecting different dilution series. Eight default colors are used if the argument is NULL.
...	extra arguments for generic or plotting routines

Details

The RPPAFit class holds the results of fitting a response model to all the dilution series on a reverse-phase protein array. For details on how the model is fit, see the RPPAFit function. By fitting a joint model, we assume that the response curve is the same for all dilution series on the array. The real point of the model, however, is to be able to draw inferences on the δ_i, which represent the (log) concentration of the protein present in different dilution series.

Value

The coef and coefficients methods return the numeric model coefficients from objects returned by modeling functions.

The fitted method returns a numeric vector.

The hist method returns an object of class histogram.

The image method invisibly returns the object x on which it was invoked.

The plot method invisibly returns the object x on which it was invoked.

The resid and residuals methods return a numeric vector.

The summary method invisibly returns NULL.

Objects from the Class

Objects should be constructed using the RPPAFit function.

Slots

`call`: object of class `call` specifying the function call that was used to generate this model fit

`rppa`: object of class `RPPA` containing the raw data that was fit

`measure`: character string containing the name of the measurement column in the raw data that was fit by the model

`method`: character string containing the name of the method that was used to estimate the upper and lower limit parameters in the model

`trimset`: numeric vector of length 5 containing the low and high intensities, the low and high concentrations that mark the trimming boundaries, and the trim level used

`model`: object of class `FitClass` unique to the model that was fit

`noise`: numeric vector of estimated relative background concentrations for noise for use in calculating qc values for positive control dilution series with Spot.Types designated as posCtrl-Noise or Noise.

`concentrations`: numeric vector of estimates of the relative log concentration of protein present in each sample

`lower`: numeric vector containing the lower bounds on the confidence interval of the log concentration estimates

`upper`: numeric vector containing the upper bounds on the confidence interval of the log concentration estimates

`conf.width`: numeric scalar specifying width of the confidence interval

`intensities`: numeric vector containing the predicted observed intensity at the estimated concentrations for each dilution series

`ss.ratio`: numeric vector containing statistic measuring the R^2 for each individual dilution series

`warn`: character vector containing any warnings that arose when trying to fit the model to individual dilution series

`version`: character string containing the version of RPPASPACE that produced the fit

Methods

coef `signature(object = "RPPAFit")`:
Extracts model coefficients from objects returned by modeling functions.

coefficients `signature(object = "RPPAFit")`:
An alias for `coef`.

fitted `signature(object = "RPPAFit")`:
Extracts the fitted values of the model. This process is more complicated than it may seem at first, since we are estimating values on both the X and Y axes. By default, the fitted values are assumed to be the intensities, Y, which are obtained using either an uppercase or lowercase 'y' as the `type` argument. The fitted log concentrations are returned when `type` is set to either uppercase or lowercase 'x'. In the notation used above to describe the model, these fitted values are given by $X_i = X - \delta_i$.

hist `signature(x = "RPPAFit")`:
Produces a histogram of the residuals. The exact form of the residuals being displayed depends on the value of the `type` argument.

image `signature(x = "RPPAFit")`:
Produces a 'geographic' plot of either the residuals or the fitted values, depending on the value of the `measure` argument. The implementation reuses code from the `image` method for an RPPA object.

plot `signature(x = "RPPAFit", y = "missing")`:
Produces a diagnostic plot of the model fit. The default `type`, 'cloud', simply plots the fitted X values against the observed Y values as a cloud of points around the jointly estimated sigmoid curve. The 'series' plot uses different colored lines to join points belonging to the same dilution series. The 'individual' plot produces separate graphs for each dilution series, laying each one alongside the jointly fitted sigmoid curve.

resid `signature(object = "RPPAFit")`:
An alias for `residuals`.

residuals `signature(object = "RPPAFit")`:
Reports the residual errors. The 'raw' residuals are defined as the difference between the observed intensities and the fitted intensities, as computed by the `fitted` function. The 'standardized' residuals are obtained by standardizing the raw residuals.

summary `signature(object = "RPPAFit")`:
Prints a summary of the `RPPAFit` object, which reports the function call used to fit the model and important fitting parameters.

Author(s)

Kevin R. Coombes <coombes.3@osu.edu>, P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

RPPA, RPPADesignParams, RPPAFit, hist

`RPPAFitParams-class` *Fitting Dilution Curves to Protein Lysate Arrays with Class "RPPAFitParams"*

Description

The `RPPAFit` function fits an intensity response model to the dilution series in a reverse-phase protein array experiment. Individual sample concentrations are estimated by first matching individual sample dilution series to the overall logistic response for the slide and then fitting a second time using the specified model, usually cobs. The `RPPAFitParams` class is a convenient place to wrap the parameters that control the model fit into a reusable object.

Usage

```
RPPAFit(rppa,
        measure,
        model="logistic",
```

```
        xform=NULL,
        method=c("nls", "nlrob", "nlrq"),
        trim=2,
        ci=FALSE,
        ignoreNegative=TRUE,
        trace=FALSE,
        verbose=FALSE,
        veryVerbose=FALSE,
        warnLevel=0,
  residualsrotation = as.integer(0)
  )

RPPAFitParams(  measure,
    model="logistic",
    xform=NULL,
    method=c("nls", "nlrob", "nlrq"),
    trim=2,
    ci=FALSE,
    ignoreNegative=TRUE,
    trace=FALSE,
    verbose=FALSE,
    veryVerbose=FALSE,
    warnLevel=0,
    residualsrotation = as.integer(0)
   )

RPPAFitFromParams(rppa,
                  fitparams,
                  progmethod=NULL)

is.RPPAFit(x)
is.RPPAFitParams(x)
## S4 method for signature 'RPPAFitParams'
paramString(object, slots, ...)
```

Arguments

rppa	object of class RPPA containing the raw data to be fit
fitparams	object of the class RPPAFitParams, bundling together the following arguments.
progmethod	user defined function that will take a string telling which portion of the process is running and do with it as the function specifies. Default is a function that does nothing.
measure	character string identifying the column of the raw RPPA data that should be used to fit to the model.
model	character string specifying the model for the response curve fitted for the slide. Valid values are:

"logistic"	assumes a logistic shape for the curve
"loess"	fits a loess curve to the response
"cobs"	fits a b-spline curve to the slide with the constraint that curve be strictly increasing

xform — optional function that takes a single input vector and returns a single output vector of the same length. The measure column is transformed using this function before fitting the model.

method — character string specifying the method for matching the individual dilution series to the response curve fitted for the slide. Valid values are:

"nls"	uses the optimal fit based on nonlinear least squares
"nlrob"	uses nlrob which is robust nls from **robustbase** package
"nlrq"	uses nlrq which is robust median regression from **quantreg** package

trim — numeric or logical scalar specifying trim level for concentrations. If positive, concentrations will be trimmed to reflect min and max concentrations we can estimate given the background noise. If TRUE, the trim level defaults to 2, which was originally the hardcoded value; otherwise, raw concentrations are returned without trimming.

ci — logical scalar. If TRUE, computes 90% confidence intervals on the log concentration estimates.

ignoreNegative — logical scalar. If TRUE, converts negative values to NA before fitting the model.

trace — logical scalar passed to nls in the method portion of the routine

verbose — logical scalar. If TRUE, prints updates while fitting the data

veryVerbose — logical scalar. If TRUE, prints voluminous updates as each individual dilution series is fitted

warnLevel — integer scalar used to set the warn option before calling method. Since this is wrapped in a try function, it won't cause failure but will give us a chance to figure out which dilution series are failing. Setting warnLevel to two or greater may change the values returned.

residualsrotation — numeric scalar containing 90 degree value to rotate the generated residuals image by when generating the output graphic. This should be used if the layout of the information in the input txt file does not match the orientation of the slide input image.

object — object of class RPPAFitParams

x — object of class RPPAFit (or RPPAFitParams)

slots — strings specifying RPPAFitParams slotnames to display (for debugging)

... — extra arguments for generic routines.

Details

The basic mathematical model is given by

$$Y = f(X - \delta_i),$$

where Y is the observed intensity, X is the designed dilution step and f is the model for the protein response curve. By fitting a joint model, we assume that the response curve is the same for all dilution series on the array. The real point of the model, however, is to be able to draw inferences on the δ_i, which represent the (log) concentration of the protein present in different dilution series.

As the first step in fitting the model, we compute crude estimates of the individual δ_i assuming a rough logistic shape for the protein response curve.

Next, we fit an overall response curve for the slide f using the estimated concentrations and observed intensities $Y = f(\delta_i)$. The model for f is specified in the $model$ parameter.

Next, we update the estimates of the individual δ_i using our improved fitted model f for the overall slide response curve. These individual series are matched to the overall slide response curve using the algorithm specified in `method`. The default method is `nls`, a least squares match-up, but we also offer robust alternatives which can do better.

Finally, we re-estimate f using the improved estimates for δ_i. We continue to iterate between f and δ_i. We do this twice since that seems to give reasonable convergence.

If the `ci` argument is `TRUE`, then the function also computes confidence intervals around the estimates of the log concentration. Since this step can be time-consuming, it is not performed by default. Moreover, confidence intervals can be computed after the main model is fit and evaluated, using the `getConfidenceInterval` function.

Value

The `RPPAFit` generator and `RPPAFitFromParams` function return an object of class `RPPAFit`.

The `RPPAFitParams` generator returns an object of class `RPPAFitParams`.

The `is.RPPAFit` method returns `TRUE` if its argument is an object of class `RPPAFit`.

The `is.RPPAFitParams` method returns `TRUE` if its argument is an object of class `RPPAFitParams`.

The `paramString` method returns a character vector, possibly empty but never `NULL`.

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the `RPPAFitParams` function.

Slots

`measure`: character; see arguments above

`xform`: function or `NULL`; see arguments above

`method`: character; see arguments above

`ci`: logical scalar; see arguments above

`ignoreNegative`: logical scalar; see arguments above

`trace`: logical scalar; see arguments above

`verbose`: logical scalar; see arguments above

`veryVerbose`: logical scalar; see arguments above

`warnLevel`: numeric; see arguments above

`trim`: numeric; see arguments above

`model`: character; see arguments above

`residualsrotation`: numeric; see arguments above

Methods

paramString `signature(object = "RPPAFitParams")`:
Returns string representation of object.

Warning

The `paramString` method should not be called by user except for informational purposes. The content and format of the returned string may vary between different versions of this package.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, Kevin R. Coombes <coombes.3@osu.edu>, James M. Melott <jmmelott@mdanderson.org>

See Also

RPPAFit, RPPAFit-class, RPPA, RPPADesignParams

Examples

```
showClass("RPPAFitParams")
fitparams <- RPPAFitParams(measure="Net.Value",
         method="nls",
         model="cobs",
         trim=2,
         ci=FALSE,
         ignoreNegative=FALSE,
         warnLevel=-1
         )
paramString(fitparams)
```

RPPANormalizationParams-class

Class "RPPANormalizationParams"

Description

The RPPANormalizationParams class is used to bundle the parameter set together that control how to perform spatial adjustment into a reusable object.

Usage

```
RPPANormalizationParams(method,
                        arglist=NULL)
is.RPPANormalizationParams(x)
## S4 method for signature 'RPPANormalizationParams'
paramString(object, slots, ...)
```

Arguments

method	character string specifying normalization method to use
arglist	list of named key/value pairs representing argument list to be passed upon invocation of normalize method
object	object of class RPPANormalizationParams
x	object of class RPPANormalizationParams
slots	strings specifying RPPANormalizationParams slotnames to display (for debugging)
...	extra arguments for generic routines

Details

The method argument is combined with the arglist argument prior to invocation of normalize method.

Value

The RPPANormalizationParams generator returns an object of class RPPANormalizationParams.

The is.RPPANormalizationParams method returns TRUE if its argument is an object of class RPPANormalizationParams.

The paramString method returns a character vector, possibly empty but never NULL.

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the RPPANormalizationParams generator function.

Slots

name: character string; see arguments above

method: character string; see arguments above

arglist: list of named key/value pairs; see arguments above

Methods

paramString signature(object = "RPPANormalizationParams"):
Returns string representation of object.

Warning

The paramString method should not be called by user except for informational purposes. The content and format of the returned string may vary between different versions of this package.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

normalize

Examples

```
showClass("RPPANormalizationParams")
normparams <- RPPANormalizationParams(method="medpolish",
                                      arglist=list(calc.medians=FALSE))
paramString(normparams)
```

RPPAPreFitQC-class *Class "RPPAPreFitQC"*

Description

The RPPAPreFitQC class represents the inputs necessary to determine the quality control rating of a reverse-phase protein array slide.

Usage

```
RPPAPreFitQC(rppa, useAdjusted=FALSE)
is.RPPAPreFitQC(x)
## S4 method for signature 'RPPAPreFitQC'
qcprob(object, ...)
## S4 method for signature 'RPPAPreFitQC'
summary(object, ...)
```

Arguments

rppa	object of class RPPA containing the raw data to be assessed
useAdjusted	logical scalar. If TRUE, spatially adjusted measures are used instead of Net.Value and Raw.Value.
object	object of (sub)class RPPAPreFitQC
x	object of (sub)class RPPAPreFitQC
...	extra arguments for generic routines

Value

The RPPAPreFitQC generator returns an object of subclass of class RPPAPreFitQC.

The is.RPPAPreFitQC method returns TRUE if its argument is an object of subclass of class RPPAPreFitQC.

The summary method returns a summary of the underlying data frame.

Objects from the Class

Objects are created by calls to the RPPAPreFitQC factory method.

Methods

qcprob signature(object = "RPPAPreFitQC"):
Placeholder method which must be implemented by subclass.

summary signature(object = "RPPAPreFitQC"):
Placeholder method which must be implemented by subclass.

Warning

The current implementation only handles designs with 5 dilution series.
Anything else will fail.

Author(s)

P. Roebuck <paul_roebuck@comcast.net> James M. Melott <jmmelott@mdanderson.org>

RPPASet-class *Class "RPPASet"*

Description

The RPPASet class fits rppaspace curves to an entire directory of reverse-phase protein array experiments.

Usage

```
RPPASet(path,
        designparams,
        fitparams,
        spatialparams=NULL,
        normparams,
        doprefitqc=FALSE,
    parallelClusterSize,
  residualsrotation = as.integer(0),
    warningsFileName="warnings.txt",
    printTimings=TRUE
)
```

```
is.RPPASet(x)
## S4 method for signature 'RPPASet'
normalize(object,
          ...)
## S4 method for signature 'RPPASet'
summary(object,
        onlynormqcgood=ran.prefitqc(object),
        ...)
## S4 method for signature 'RPPASet'
write.summary(object,
              path,
              prefix="rppaspace",
              graphs=TRUE,
     createcombinedoutputimage=FALSE,
              imagedir=NULL,
              onlynormqcgood=ran.prefitqc(object),
              imageextension=".tif",
              imagerotation=as.integer(0),
              residualsrotation=as.integer(0),
     majorXDivisions = object@design@majorXDivisions,
     majorYDivisions = object@design@majorYDivisions,
              ...)
```

Arguments

path	character string specifying a directory. In the case of the RPPASet generator, it specifies the directory containing the quantification files (.txt) to be processed. In the case of the write.summary method, it specifies the directory where output should be stored.
designparams	object of class RPPADesignParams describing features common to all quantification files
fitparams	object of class RPPAFitParams containing parameters used to fit the rppaspace model
spatialparams	object of class RPPASpatialParams containing parameters used to perform spatial adjustment, or NULL
normparams	object of class RPPANormParams containing parameters used to normalize the concentrations
doprefitqc	logical scalar. If TRUE, performs pre-fit quality control.
printTimings	TRUE/FALSE whether or not to print out the time taken as the method is run. Used for performance debugging purposes.
object	object of class RPPASet
prefix	character string used as a filename prefix on files generated by the write.summary method.
graphs	logical scalar. If TRUE, produces fit graphs.
createcombinedoutputimage	logical scalar. If TRUE, produces output png consisting of combined png images and scaled version of original slide image.

`imagedir` character string specifying the directory containing the images corresponding to the quantification files

`imageextension` character string specifying extension to use when searching for images matching the slide file names.

`imagerotation` numeric scalar containing 90 degree value to rotate the input image by when appending it to the generated graphs in the combined output image file for each slide.

`residualsrotation`
numeric scalar containing 90 degree value to rotate the generated residuals image by when generating the output graphic. This should be used if the layout of the information in the input txt file does not match the orientation of the slide input image.

`majorXDivisions`
integer to describe distance between grid lines on the X axis of the R2 residuals plot. Defaults to 10 if NA or invalid value provided.

`majorYDivisions`
integer to describe distance between grid lines on the Y axis of the R2 residuals plot. Defaults to 10 if NA or invalid value provided.

`warningsFileName`
character string specifying file to append any warnings generated by this function.

`onlynormqcgood` logical scalar. If `TRUE`, filters the slides to be normalized according to their pre-fit quality control scores.

`x` object of class `RPPASet`

`parallelClusterSize`
Number of parallel cores to use when processing.

`...` extra arguments for generic or plotting routines

Details

Quantify all the slides in a directory using `RPPASet` generator. This returns an object containing slide data and fits for each slide. Typically this is followed by a call to `write.summary` to write the resulting quantifications and diagnostic plots to a directory.

Potentially generates multiple CSV and TSV files: one for the raw concentrations (rppaspace_conc_raw.csv"), one for the R^2 statistics (rppaspace_ss_ratio.csv), and one for the normalized concentrations (rppaspace_conc_norm_[norm_method].csv); a fourth file containing the goodness of fit probabilities (rppaspace_prefit_qc.csv) may be present if pre-fit QC analysis was requested. If spatial adjustments were requested, a TSV file (spatial_adjustments.tsv) will be created. If positive control dilution series have been declared as Noise or PosCtrl-Noise points in the design file, an additional CSV file of noise statistics will be created (rppaspace_noise.csv). If prefit QC analysis was done and/or noise qc metric were created, a combined qc metrics file will be created (rppaspace_combined_qc.csv) as well. Additionally, a TSV file detailing completion of each stage of processing for each slide is produced (rppaspace_summary.tsv).

If `imagedir` is `NULL`, the directory is assumed to be a sibling directory to `path` named "tif". If `graphs` is `TRUE`, two PNG files containing output graphs are created per antibody. The original slide image is merged with these output PNG graph files, generating an additional JPEG file per antibody.

Value

The RPPASet generator returns an object of class RPPASet.

The is.RPPASet method returns TRUE if its argument is an object of class RPPASet.

The summary method returns an object of class RPPASetSummary.

The write.summary method invisibly returns NULL.

Objects from the Class

Although objects of the class can (in theory) be created by a direct call to new, the only realistic method is to use the RPPASet generator function.

Slots

call: object of class call specifying the function call that was used during construction

version: character string containing the version of this package used to construct the object

design: object of class RPPADesignParams, common to all the slides

errorsFileName: character string holding the name of the file to which to write out error messages generated during processing.

warningsFileName: character string holding the name of the file to which to write out warning messages generated during processing.

rppas: array of objects of class RPPA

spatialparams: object of class RPPASpatialParams that was used to perform spatial adjustment, or NULL

prefitqcs: array of objects of class RPPAPreFitQCParams

fitparams: object of class RPPAFitParams that was used to construct the model fits

normparams: object of class RPPANormalizationParams used to normalize the raw concentrations

fits: array of fitted objects of class RPPAFit

completed: logical matrix specifying stage completion for each slide

Methods

normalize signature(object = "RPPASet"):
Assembles matrix of concentrations from all fits in object, using the object's normalization settings.

summary signature(object = "RPPASet"):
Creates an object of class RPPASetSummary.

write.summary signature(object = "RPPASet"):
Writes a record of the entire RPPASet, including fitted values, residuals, and images of the processed slides.

Author(s)

Kevin R. Coombes <coombes.3@osu.edu>, P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

RPPA, RPPADesignParams, RPPAFit, RPPASetSummary

RPPASetSummary-class *Class "RPPASetSummary"*

Description

The RPPASetSummary class contains the summary information derived from an RPPASet object.

Usage

```
RPPASetSummary(rppaset,
               onlynormqcgood=ran.prefitqc(rppaset))
is.RPPASetSummary(x)
## S4 method for signature 'RPPASetSummary'
write.summary(object,
              path,
              prefix="rppaspace",
              ...)
```

Arguments

rppaset	object of class RPPASet
onlynormqcgood	logical scalar. If TRUE, filters the slides to be normalized according to their pre-fit quality control scores.
x	object of class RPPASetSummary
object	object of class RPPASetSummary
path	character string specifying the path from the current directory to the directory containing the files to be processed
prefix	character string used as a prefix on files generated by the write.summary method
...	extra arguments for generic routines

Value

The RPPASetSummary generator returns an object of class RPPASetSummary.

The is.RPPASetSummary method returns TRUE if its argument is an object of class RPPASetSummary.

The write.summary method invisibly returns NULL.

Objects from the Class

Although objects of the class can (in theory) be created by a direct call to new, the only realistic method is to use the RPPASetSummary generator function.

Slots

`raw`: numeric matrix of raw concentrations

`ss`: numeric matrix of R^2 statistical values

`norm`: numeric matrix of normalized concentrations

`probs`: numeric vector of goodness of fit probabilities, or `NULL` (if pre-fit QC analysis was not requested)

`completed`: logical matrix specifying stage completion for each slide

`noise`: numeric vector of calculated log concentrations for noise qc values for positive control dilution series with Spot.Types designated as posCtrl-Noise or Noise.

`design`: object of class `RPPADesignParams`, common to all the slides

`onlynormqcgood`: logical scalar specifying if raw concentrations were filtered according to their pre-fit quality control scores prior to normalization

`version`: character string containing the version of this package used to construct the object

Methods

write.summary `signature(object = "RPPASetSummary")`:

Potentially generates multiple CSV and TSV files: one for the raw concentrations (rppaspace_conc_raw.csv"), one for the R^2 statistics (rppaspace_ss_ratio.csv), and one for the normalized concentrations (rppaspace_conc_norm_[norm_method].csv); a fourth file containing the goodness of fit probabilities (rppaspace_prefit_qc.csv) may be present if pre-fit QC analysis was requested. If spatial adjustments were requested, a TSV file (spatial_adjustments.tsv) will be created. If positive control dilution series have been declared as Noise or PosCtrl-Noise points in the design file, an additional CSV file of noise statistics will be created (rppaspace_noise.csv). If prefit QC analysis was done and/or noise qc metric were created, a combined qc metrics file will be created (rppaspace_combined_qc.csv) as well. Additionally, a TSV file detailing completion of each stage of processing for each slide is produced (rppaspace_summary.tsv).

Note

The three CSV files may be reordered (to match that of the original input) when written to disk.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

`RPPASet`

RPPASPACESettings-class
Class "RPPASPACESettings"

Description

The RPPASPACESettings class represents the arguments needed to perform curve fitting.

Usage

```
RPPASPACESettings(txtdir,
    imgdir,
    outdir,
    designparams,
    fitparams,
    spatialparams=NULL,
    normparams,
    doprefitqc=FALSE,
    onlynormqcgood=doprefitqc,
    parallelClusterSize=as.integer(1),
    createcombinedoutputimage = FALSE,
    imageextension=".tif",
    imagerotation=as.integer(0),
    residualsrotation=as.integer(0),
    warningsFileName="warnings.txt",
    errorsFileName = "errors.txt"
)
fitCurveAndSummarizeFromSettings(settings)
is.RPPASPACESettings(x)
## S4 method for signature 'RPPASPACESettings'
write.summary(object,
              path=as(object@outdir, "character"),
              ...)
## S4 method for signature 'RPPASPACESettings'
paramString(object,
            designparams.slots,
            fitparams.slots,
            spatialparams.slots,
            normparams.slots,
            ...)
```

Arguments

txtdir	character string specifying the directory containing quantification files in text format
imgdir	character string specifying the directory containing image files associated with each of the aforementioned quantification files, or NULL. All image files for a

given run must be of one image file type. Other files in the directory will be ignored.

`outdir` character string specifying the directory where output from analysis should be stored. Must be writable.

`designparams` object of class `RPPADesignParams`

`fitparams` object of class `RPPAFitParams`

`spatialparams` object of class `RPPASpatialParams`, or `NULL`

`normparams` object of class `RPPANormalizationParams`

`doprefitqc` logical scalar. If `TRUE`, performs pre-fit quality control.

`onlynormqcgood` logical scalar. If `TRUE`, filters the slides to be normalized according to their pre-fit quality control scores.

`parallelClusterSize`
Number of parallel cpus to use on computer when running RPPASPACE. Spatial corrections and fitting diltion series to the calculated curve sections of the code will be done in parallel when this number is greater than 1. Defaults to 1 for backwards compatibility if not specified.

`createcombinedoutputimage`
logical scalar. If `TRUE`, an output png file will be created for each valid slide in the set. The png file that is a composite of the two generated png files and the original slide image.

`imageextension` character string specifying extension to use when searching for images matching the slide file names. Acceptable values are (".tif", ".png", ".bmp", ".gif", ".jpg")

`imagerotation` numeric scalar containing 90 degree value to rotate the input image by when appending it below the generated graphs in the combined output image file for each slide. Defaults to 0 if not specified. Acceptable values (0, 90, 180, 270)

`residualsrotation`
numeric scalar containing 90 degree value to rotate the generated residuals image by when generating the output graphic. This should be used if the layout of the information in the input txt file does not match the orientation of the slide input image. Defaults to 0 if not specified. Acceptable values (0, 90, 180, 270)

`warningsFileName`
character string specifying file to append any warnings generated by this function. Defaults to "warnings.txt"

`errorsFileName` character string specifying file to append any errors generated by this function. Defaults to "errors.txt"

`object` object of class `RPPASPACESettings`

`settings` object of class `RPPASPACESettings`

`x` object of class `RPPASPACESettings`

`path` character string specifying the directory where settings summary should be saved. Must be writable.

`designparams.slots`
strings specifying `RPPADesignParams` slotnames to display (for debugging)

`fitparams.slots`
strings specifying `RPPAFitParams` slotnames to display (for debugging)

spatialparams.slots
: strings specifying RPPASpatialParams slotnames to display (for debugging)

normparams.slots
: strings specifying RPPANormalizationParams slotnames to display (for debugging)

...
: extra arguments for generic routines

Value

The RPPASPACESettings generator returns an object of class RPPASPACESettings.

The is.RPPASPACESettings method returns TRUE if its argument is an object of class RPPASPACESettings.

The paramString method returns a character vector, possibly empty but never NULL.

The write.summary method invisibly returns NULL.

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the RPPASPACESettings generator function.

Slots

txtdir: object of class Directory specifying the directory containing quantification files in text format

imgdir: object of class Directory specifying the directory containing TIFF image files

outdir: object of class Directory specifying the directory where analysis results should be stored

designparams: object of class RPPADesignParams specifying the parameters that describe how a particular set of RPPA slides was designed

fitparams: object of class RPPAFitParams specifying the parameters that control model fit

spatialparams: object of class RPPASpatialParams specifying the parameters that control spatial adjustment

normparams: object of class RPPANormalizationParams specifying the parameters that control normalization

doprefitqc: see argument

createcombinedoutputimage: see argument

imageextension: see argument

imagerotation: see argument

residualsrotation: see argument

onlynormqcgood: see argument

seriesToIgnore: see argument

parallelClusterSize: see argument

warningsFileName: see argument

errorsFileName: see argument

Methods

paramString `signature(object = "RPPASPACESettings")`:
Returns string representation of object.

write.summary `signature(object = "RPPASPACESettings")`:
Writes a text file representation of object.

Warning

The `paramString` method should not be called by user except for informational purposes. The content and format of the returned string may vary between different versions of this package.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

Directory, RPPADesignParams, RPPASpatialParams, RPPAFitParams, RPPANormalizationParams

Examples

```
    ## Not run:
showClass("RPPASPACESettings")

#Insert an existing directory containing txt, img, and out subdirectories
#
analysishome <- "C:/temp"

txtdir <- file.path(analysishome, "txt" )
imgdir <- file.path(analysishome, "img" )
outdir <- file.path(analysishome, "out")
number_cpus_to_use <- 2

warningsFileName <- "warnings.txt"
errorsFileName <- "errors.txt"

designparams <- RPPADesignParams(center=FALSE,
       seriesToIgnore=list(),
       majorXDivisions=as.integer(10),
       majorYDivisions=as.integer(10)
       )

spatialparams <- RPPASpatialParams(cutoff=0.8,
       k=100,
       gamma=0.1,
       plotSurface=FALSE)

fitparams <- RPPAFitParams(measure="Net.Value",
       method="nls",
       model="cobs",
       trim=2,
```

```
        ci=FALSE,
        ignoreNegative=FALSE,
        warnLevel=-1
        )

normparams <- RPPANormalizationParams(method="none")

settings <- RPPASPACESettings(txtdir=txtdir,
    imgdir=imgdir,
    outdir=outdir,
    designparams=designparams,
    spatialparams=spatialparams,
    doprefitqc=TRUE,
    fitparams=fitparams,
    normparams=normparams,
    onlynormqcgood=FALSE,
    imageextension=".jpg",
    createcombinedoutputimage=TRUE,
    warningsFileName=warningsFileName,
    parallelClusterSize=as.integer(number_cpus_to_use))

#Print the created object
paramString(settings)

## End(Not run)
```

`RPPASpatialParams-class` *Class "RPPASpatialParams"*

Description

The RPPASpatialParams class is used to bundle the parameter set together that control how to perform spatial adjustment into a reusable object.

Usage

```
RPPASpatialParams(cutoff=0.8,
                  k=100,
                  gamma=0.1,
                  plotSurface=FALSE)
is.RPPASpatialParams(x)
## S4 method for signature 'RPPASpatialParams'
paramString(object, slots, ...)
```

Arguments

`cutoff`	numeric scalar used to identify the background cutoff with value in closed interval [0..1]. Default is `0.8`.

k	numeric scalar used as smoothing model argument. Default is 100.
gamma	numeric scalar used as model parameter with value in closed interval [0..2]. Default is 0.1.
plotSurface	logical scalar. If TRUE, plots surfaces. Default is FALSE.
object	object of class RPPASpatialParams
x	object of class RPPASpatialParams
slots	strings specifying RPPASpatialParams slotnames to display (for debugging)
...	extra arguments for generic routines

Details

The cutoff argument passed to quantile is percentile of the background estimates used to define the noise region of slide.

The k argument passed to s sets upper limit on degrees of freedom associated with smoothing.

The gamma argument passed to gam provides a constant multiplier used to inflate model degrees of freedom in the GCV or UBRE/AIC score.

Value

The RPPASpatialParams generator returns an object of class RPPASpatialParams.

The is.RPPASpatialParams method returns TRUE if its argument is an object of class RPPASpatialParams.

The paramString method returns a character vector, possibly empty but never NULL.

Objects from the Class

Although objects of the class can be created by a direct call to new, the preferred method is to use the RPPASpatialParams generator function.

Slots

cutoff: numeric scalar; see arguments above

k: numeric scalar; see arguments above

gamma: numeric scalar; see arguments above

plotSurface: logical scalar; see arguments above

Methods

paramString(object) Returns string representation of object.

Warning

The paramString method should not be called by user except for informational purposes. The content and format of the returned string may vary between different versions of this package.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

See Also

spatialCorrection

Examples

```
showClass("RPPASpatialParams")
spatialparams <- RPPASpatialParams(cutoff=0.8,
          k=100,
          gamma=0.1,
          plotSurface=FALSE)
paramString(spatialparams)
```

---

spatialCorrection *Spatial Correction*

Description

This function estimates a smoothed surface from positive control spots on an RPPA slide. The surface is used to perform spatial corrections (i.e., because of uneven hybridization) on the array. It is used before RPPAFit, one slide at a time.

Usage

```
spatialAdjustmentFromParams(rppa,
                            spatialparams)
spatialAdjustment(rppa,
                  cutoff=0.8,
                  k=100,
                  gamma=0.1,
                  plotSurface=FALSE)
spatialCorrection(rppa,
                  measure=c("Net.Value", "Raw.Value"),
                  cutoff=0.8,
                  k=100,
                  gamma=0.1,
                  plotSurface=FALSE)
```

Arguments

rppa	object of class RPPA
spatialparams	object of class RPPASpatialParams containing parameters used to perform spatial adjustment
measure	character string specifying fit measure to smooth
cutoff	numeric scalar used to identify the background cutoff with value in range [0..1]
k	numeric scalar used as smoothing model argument.
gamma	numeric scalar used as model parameter with value in range [0..2]
plotSurface	logical scalar. If TRUE, plots surfaces.

Details

The observed spot intensities are assumed to be a combination of true signal, background noise, and hybridization effects according to the following model:

$$Y_r c = Y * H_r c + B_r c$$

where $Y_r c$ is the observed intensity, Y is the true signal, $H_r c$ is the effect of hybridization, and $B_r c$ is the background noise. The subscripts "r" and "c" refer to the physical row and column of the spot on the array. Background noise is estimated locally by the array software. The hybridization effect is estimated fitting a generalized additive model (GAM) to positive control spots printed uniformly across the array.

The estimated surface is used to scale the intensities on the array. Each intensity is adjusted by the amount that is needed to make the positive control surface flat at the value of the median of the surface. This is done by dividing each spot by the estimated surface value and then multiplying by the median of the surface.

Positive control spots that are expressed below the cutoff for the noise region are excluded from the computation of the surface.

Sometimes, positive control spots are printed in a dilution series to avoid saturation problems with these spots. When this happens, the observed intensities are adjusted by the positive control surface that has the most similar expression level.

The `cutoff` argument passed to `quantile` is percentile of the background estimates used to define the noise region of slide.

The `k` argument passed to `s` sets upper limit on degrees of freedom associated with smoothing.

The `gamma` argument passed to `gam` provides a constant multiplier used to inflate model degrees of freedom in the GCV or UBRE/AIC score.

Value

Returns modified `rppa` with an additional measurement column named after the `measure` with an `Adj.` prefix. For example, if the measure was `Net.Value`, the name of the adjusted column would be `Adj.Net.Value`.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, E. Shannon Neeley <sneeley@stat.byu.edu>, James M. Melott <jmmelott@mdanderson.org>

References

Neeley ES, Baggerly KA, Kornblau SM.
Surface Adjustment of Reverse Phase Protein Arrays Using Positive Control Spots
Cancer Informatics (2012) 11: 77-86.
https://pubmed.ncbi.nlm.nih.gov/22550399/

See Also

`RPPASpatialParams`, `quantile`, `gam`, `s`, `choose.k`

write.summary-method *Method "write.summary"*

Description

write.summary is a generic function used like a summary method that writes to disk, saving summary information from the object in an external format. The method invokes particular methods which depend on the class of the first argument.

Usage

```
## S4 method for signature 'ANY'
write.summary(object, ...)
```

Arguments

object	an object for which saving summary information externally is desired
...	additional arguments affecting the summary information produced

Note

Exactly what is written to disk by write.summary depends on the class of its argument. See the documentation of the particular methods for details of what is written by that method.

Author(s)

P. Roebuck <paul_roebuck@comcast.net>, James M. Melott <jmmelott@mdanderson.org>

Index