Package 'immunoassay'

March 31, 2011

Type Package

Title Functions for working with immunoassay data.

Version 0.3

Date 2011-03-29

Author Michal J. Figurski, PhD; Leslie M. Shaw, PhD.

Maintainer Michal J. Figurski <mfigrs@gmail.com>

Description This package contains several functions for loading immunoassay data, fine-tuning of sigmoidal models beyond the limits of instrument software; and for batch processing of immunoassay output files and creation of reports with nice graphics.

Depends R (>= 2.0.0), plotrix

License GPL-2

LazyLoad yes

R topics documented:

immunoassay-package *Functions to aid fitting and batch processing of immunoassay data*

Description

The package immunoassay contains several functions to aid batch processing of data from immunoassay runs, although currently only the loading of Multiplex data is implemented. There are functions to load data: read.multiplex, fit a choice of sigmoidal models: sigfit, and a batch-processing function to process a whole directory of data files, validate the results and produce reports: batch. There also are associated print, summary and plot methods, that can be used in report templates. Function batch can produce text or LaTeX reports, where the latter are made based upon project-specific templates, that are separate R/Sweave scripts. These functions are in early stage of development - they've been tested on data from a single instrument platform, in a single study environment, and with 2- and 3-analyte runs only. In other projects that utilize different plate layout, or with greater number of analytes, or simply with different sample-naming conventions, these functions may require major modifications in order to work properly. Nevertheless, this package proved remarkably useful in our laboratory allowing for fast processing of large amounts of data (projects of 70 and more runs) and creation of nice reports and summary plots.

Details

Package:	immunoassay
Type:	Package
Version:	0.3
Date:	2011-03-28
License:	GPL-2
LazyLoad:	yes

There are three main functions in this package:

read.multiplex A function to extract data from a multiplex .csv file.

sigfit A function that fits a choice of sigmoidal models to the data.

batch A batch-processing function that uses the above two functions to process folders of data files.

The functions in the immunoassay package define two classes: ima class for raw immunoassay data, and sigfit class for fitted sigmoidal models. For these classess print, summary and plot methods exist. See help on individual functions for more details.

Additionally the immunoassay package relies on several global vectors:

immunoassay.kits A data frame containing kit information (concentrations of Standards and QCs for every analyte) for the kits used in producing the data to be processed. This is a required element - functions will not run without it, however the batch function will load the kit data automatically if it exists in proper location and has a proper name. For more information, see help on read.kits and batch functions.

immunoassay.coefs A list created by batch function, that contains equation coefficients for every fitted model. This list is used by the fitting function to calculate starting values for subsequent

fits. This element is not required to start (default starting values will be used), but it will be automatically created. For more information, see help on batch function.

immunoassay.options A list of all the model settings for the data files in the project folder - used by the batch function. It is not required to start, but will be automatically created by this function. If the processing fails, the records in this list can be modified and re-used. For more information, see help on batch function.

immunoassay.environment A pointer to the environment of the batch function, for use within the Sweave project report templates. It is automatically created by the batch function. It's necesary due to known shortcoming of Sweave that can work in the global environment only.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com> and Leslie M. Shaw of the Biomarker Research Laboratory, University of Pennsylvania, Philadelphia, PA.

See Also

batch, read.multiplex, read.kits and sigfit.

---

batch *Batch-processing function for immunoassay data.*

Description

This is a wrapper function making use of other lower-level functions in this package. It looks for immunoassay data files in a given folder, processes these files and accumulates results into a single data frame. Optionally, it can create a project report - either in a simple text form, or in LaTeX format.

Usage

```
batch(path, subfolder = "", kit.file = NULL, analytes = "all",
    run.options     = list(CTS = 20, MFI = "last", CV = 20),
    model.options   = list(model = "L.5", weights = "sqrt",
                      refit = 0.2, use = 1, stvals = "adaptive"),
    project.options = list(report = "text", template = "short",
                      trace = TRUE),
    correct.errors  = NULL)
```

Arguments

path	Character string. Path to the main project folder. A csv file with kit data should be located in that folder. Also, all subfolders are created in this folder.
subfolder	Character string. If the csv files with immunoassay data are located in the subfolder of the main folder, provide its name here. Otherwise, leave blank.
kit.file	Character string. File name of the csv file containing kit information. If a data.frame named "immunoassay.kits" exists, it will be used, and this parameter will be ignored.

`analytes` Character or numeric. This parameter is passed to `read.multiplex` function. See more information there.

`run.options` List of 3 elements that provide validation criteria: `CTS` - numeric, is the threshold for COUNTS; `MFI` - character, either "last" or "first", that is the last of first calibrator becomes the criteria for lowest MFI; and `CV` - numeric, percent CV that is threshold for between-replicate variability.

`model.options`
A list of 5 elements. Element `model` - a character or list of character values of length N, where "N" is the number of data files in the folder. Element `weights` - a character or list of character values of length N. Element `refit` - a numeric value or list of numeric values of length N. Element `use` - a numeric value or list of numeric values of length N. Element `stvals` - a character or list of character values of length N. For more information on these parameters, see help for `sigfit`

`project.options`
A list of 3 elements. Element `report` - character, either "text" or "latex" options are currently available, defaults to "text". Element `template` - character. For `report="text"` option two templates are built-in: `"short"`, that produces quick summary of each analyte for each plate, and `"full"` - an extended template with more information. Defaults to `"short"`. For `report="latex"` option, this is the name of a LaTeX template, located in the main project folder. For more information, see details below. Element `trace` - logical, debugging option. If true, trace information will be displayed when the function is processing data files. Defaults to `FALSE`.

`correct.errors`
A list - an alternative means to introduce corrections to the fit parameters for a small number of items. In this list, each named element is a list. The named element's name must be the name of the datafile that is to be corrected. There must be two elements in each named sub-list: `name`, that names the parameter to be corrected (currently only `name="use"` is implemented), and the second element `value` is the new value of the named parameter. Defaults to `NULL`.

Details

The `batch` function is the core function of this library. It implements automation in processing of entire folders of immunoassay data files. The function first looks for immunoassay files (currently limited to multiplex data files) under the provided path - it does so intelligently, so it can distinguish immunoassay run files from other .csv files. Then it creates a list of names of these files and processes through this list according to provided options.

For each file from the list, this function loads it using `read.multiplex` and fits the sigmoidal model using `sigfit` function with the set of parameters given in `model.options`. Next, it validates the fitted data using the criteria provided in `run.options` parameter, and finally, it creates a report as set in the `project.options`.

There are two major advantages of using this function, instead of manually processing the data: it can save substantial amount of time in processing of large number of data files, while preserving considerable flexibility in setting fit parameters and applying corrections. The second major advantage is the ability to create reports. Text reports are useful for obtaining general insight into the data and fitting process, but this function can really shine in connection with `Sweave` LaTeX report templates, that can be elaborate programs on themselves.

In order to use LaTeX template option, the user must create a minimum of two template files. These must be named: "my-project-name.run-report.r" - a template for each plate, and "my-project-name.project-report.r" - a project-wide template incorporating (or not) individual plate reports. The

"my-project-name" part of the template name must be provided as the template= parameter, when report="latex" option is used. These templates are "Sweaved" in the process of running this function.

There is one known problem associated with use of the Sweave function - it's that the Sweave works in global environment and doesn't "see" the environment within the batch function it has been called from. To overcome this, the batch function sets a global variable - a pointer to its environment, named "immunoassay.environment". Objects and data from within batch function can be then accessed within the Sweave templates using immunoassay.environment$ pre-fix.

For LaTeX template programmers, the list of the accessible objects within immunoassay.environment is as follows: validate - validation function; immunoassay.coefs (global) - list of all fit coefficients; immunoassay.kits (global) - a data.frame with kits information; ppath - full path to the data files; files - list of valid immunoassay files; N - number of files (length(files)); l.analytes - vector of analyte names; n - number of analytes (length(l.analytes)); l.run - a data.frame of class ima, containing the validated run data; and fits - a list of fitted nls models, of length "n". After all the plates are processed, additional object becomes available: results - a data.frame being a collection of all results from all plates, for use in the project-wide report.

Value

This function returns a simple data frame that is the collection of results from all processed files. It does not contain any fit information - if fit information are required, some form of report must be called.

Note

Examples of LaTeX report templates from an example project are located in the "templates" directory in the main package folder. See help for the example project for more information.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com> of the Biomarker Research Laboratory, University of Pennsylvania, Philadelphia, PA.

See Also

immunoassay package, example project, read.multiplex, read.kits and sigfit.

Examples

```
## Not run:

a <- batch("path-to-my-project", subfolder="Results",
     kit.file="kit-file-name.csv",  analytes=1:2,
     model.options   = list(model="L.5", weights=c("sqrt","1/y"), refit=0.2,
                       use=1, stvals="adaptive"),
     project.options = list(report="latex", template="my-project-name",
                       trace=FALSE),
     correct.errors  = list(`Plasma Pl072.csv`=list(name="use",
                       value=c(0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1))))


## End(Not run)
```

check *Function to examine the sigmoidal fit done by "sigfit".*

Description

This is a helper function that returns several error measurement quantities that serve to help evaluate models fitted by the `sigfit` function.

Usage

```
check(x, ...)
```

Arguments

`x`	An object of class `sigfit`.
`...`	Currently ignored.

Details

This function is used internally by the `batch` and `sigfit` functions to evaluate models.

Value

`St.error`	Mean and median value of error of fit to Standards (calibrators) in the measurement scale (not MFI scale).
`QC.error`	Mean and median value of error of fit to Quality Control samples in the measurement scale.
`SSE`	Sum of squares of error values for the fit in the measurement scale. This can be heavily affected by selection of calibrators.
`Sigma`	Residual sum of squares from the fitted model in the measurement scale. This can be heavily affected by weighting.
`Syx`	The Sy|x error value for the fit - in the measurement scale.
`r.squared`	R squared of the fit in the measurement scale. Keep in mind that this is the `nls` fit, and therefore, in certain situations, this can be a negative value.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com>

See Also

`sigfit`, `predict.sigfit`

Examples

```
## Not run:
run = read.luminex("your-path-here")
fit = sigfit(run)
check(fit)

## End(Not run)
```

EXAMPLE-project *Exemplary functions and report templates*

Description

In addition to general utility and batch processing functions, the immunoassay library allows for creating nicely formatted reports. The reports are actually a separate, customizable Sweave scripts that can define their own R functions or utilize the vast array of functions already available in R. There are two LaTeX report templates included in this library: "example.run-report.R" and "example.project-report.R". Additionally, there are two functions utilized by these templates: project.means and project.qcplot function.

Usage

```
project.means(x)
project.qcplot(x, samples=c("ConA","ConB"), type="pred",
    breaks=c(1,length(unique(x$ID))), log=FALSE, ...)
```

Arguments

x	A data.frame of results from the batch function - either immunoassay.environment$l.run or immunoassay.environment$results.
samples	Character vector of sample names to be plotted. A new plot will be called for each sample, so make sure to include par(mfrow=) parameter when using more than one sample in this place. This serves the purpose of having multiple plots on a single pdf page. Sample names must match the names in the data. The default is c("ConA","ConB").
type	Character. This is the prefix of the column name in the data, that is followed by the analyte name. Currently this can have the following values: "pred" - selects predictions from the model (calculated values); "MFI" - selects MFI values for plotting; or "CTS" - selects counts (useless).
breaks	Numeric vector of "breaks" for the calculation and display of mean and sd bands on the plot. Useful for projects that utilized more than one kit lot number, to separate results from different kits. Must be provided manually - currently no automatic selection of kits is implemented.
log	Logical. If TRUE, the function takes a natural logarithm of y-axis values. Defaults to FALSE.
...	Other parameters passed along to plot.

Details

The two functions mentioned above are simple utility functions called by the EXAMPLE report templates. These templates can be found in the "templates" folder of the immunoassay package, along with "Sweave.sty" LaTeX style. These templates are "Sweaved" in the process of batch-processing of immunoassay data using the batch function. These files should be copied to the main project folder on the user's computer and modified according to user's needs.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com> of the Biomarker Research Laboratory, University of Pennsylvania, Philadelphia, PA.

See Also

immunoassay, read.multiplex, read.kits and sigfit.

Example_run *Example Luminex run file.*

Description

This is an exemplary Luminex run file provided here for testing purpose and as a "proof of concept". It can be loaded into R using read.multiplex function.

Format

This is an output file from Luminex IS software. To review it's internal structure please open it in a text editor or a spreadsheet.

Details

This output file is provided with the immunoassay package as an example of a loadable Luminex output file.

Examples

```
## Not run:
    read.multiplex("your-path-here/Example_run.csv")

## End(Not run)
```

---

immunoassay.kits *Example dataset with kits data for use by function "sigfit".*

Description

This is an exemplary data.frame with information on kits used in the projects in our laboratory. It is a result of calling function read.kits on the example ".csv" file located in the "immunoassay/data" folder of this package. Such dataset must be provided in order for the function sigfit to work. The batch-processing function batch will load this data automatically if not already present in the search path. See more information there.

Usage

```
data(immunoassay.kits)
```

Format

A data frame with 25 observations on the following 17 variables.

`Lot` Factor. Kit lot names / numbers.

`Analyte` Factor. Analyte names.

`Unit` Factor. Analyte units.

`St1` Numeric. Concentration for Standard #1.

`St2` Numeric. Concentration for Standard #2.

`St3` Numeric. Concentration for Standard #3.

`St4` Numeric. Concentration for Standard #4.

`St5` Numeric. Concentration for Standard #5.

`St6` Numeric. Concentration for Standard #6.

`St7` Numeric. Concentration for Standard #7.

`St8` Numeric. Concentration for Standard #8.

`ConA.lo` Numeric. Low concentration limit for Control A.

`ConA.hi` Numeric. High concentration limit for Control A.

`ConB.lo` Numeric. Low concentration limit for Control B.

`ConB.hi` Numeric. High concentration limit for Control B.

`ConA` Numeric. Concentration for Control A.

`ConB` Numeric. Concentration for Control B.

Details

It is important, due to current limitations of the `immunoassay` functions, that the names of standards and controls are the same in the data files as in this kit file, and that this convention is kept consistently throughout the project.

Source

"InnoBIA Plasma Abeta forms" and "InnoBIA AlzBio3 CSF biomarkers" kit information.

Examples

```
data(immunoassay.kits)
str(immunoassay.kits)
```

KitLots *Example .csv dataset with kits data.*

Description

This is an exemplary .csv file with kit information - to be read into R using `read.kits` function.

Format

The structure of this file is exactly as the `immunoassay.kits` data file, except two last columns with calculated average values for "ConA" and "ConB" samples. See documentation for `immunoassay.kits` for more information.

Details

It is important, due to current limitations of the `immunoassay` functions, that the names of standards and controls are the same in the data files as in this .csv file, and that this convention is kept consistently throughout the project. Use this file to enter new kits data.

Source

"InnoBIA Plasma Abeta forms" and "InnoBIA AlzBio3 CSF biomarkers" kit information.

Examples

```
## Not run:
read.kits(path="your-path", file="KitLots.csv")

## End(Not run)
```

`plot.ima` *Methods for plotting and printing "ima" objects.*

Description

Use these functions to print out and make plots of raw (as well as fitted) immunoassay run objects of class `ima`.

Usage

```
    ## S3 method for class 'ima'
plot(x, type = "cts", analyte = 1, ref = 0.25, cts.scale = 350, ...)
    ## S3 method for class 'ima'
print(x, ...)
```

Arguments

`x`	An object of class `ima`.
`type`	Character. This can be either `"cts"`, `"cv"`, or `"accuracy"`. The `type="cts"` is the default for un-fitted `ima` data objects and it plots COUNTS for visual inspection. The `type="cv"` for the un-fitted `ima` object displays the %CV of the MFIs, and for fitted `ima` objects (must contain predictions from the model), displays the %CV of the results. The `type="accuracy"` is an extension of capability of this function for fitted immunoassay data objects and it plots accuracy of the predictions for Standards and QCs.
`analyte`	Integer of range from 1 to the number of analytes. Determines which analyte will be plotted.
`ref`	Numeric in range from 0 to 1, determines the size of the reference circle for type `"cts"` and `"accuracy"` plots. The default is 0.25 (25%). Use `ref=0` to suppress plotting of reference circles.
`cts.scale`	Numeric. This is the scaling factor for COUNTS only. The default is 350, which made the size of the circles small enough to fit within cells - within our project.
`...`	Other graphical parameters passed along to `plot` function. In `print` method - currently ignored.

Details

These functions are utilized to display data from `ima` objects and additionally provide some functionality for fitted objects. These functions have been thoroughly tested in only one project environment and may not work properly in another setting. In particular the naming of samples and their sequence / placement on the plate is critical.

Value

These functions are used for their side effects.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com>

See Also

`plot.sigfit`, `print.sigfit` and `summary.sigfit`.

Examples

```
## Not run:
run = read.multiplex("your-path-here")
plot(run)
run

## End(Not run)
```

---

plot.sigfit *Basic display methods for objects fitted by the "sigfit" function.*

Description

Use these functions to print out, summarize and make plots of fitted immunoassay run objects of class `sigfit`.

Usage

```
    ## S3 method for class 'sigfit'
plot(x, table = TRUE, type = "fit", norm = "weighted", ...)
    ## S3 method for class 'sigfit'
print(x, ...)
    ## S3 method for class 'sigfit'
summary(object, ...)
```

Arguments

`x`	An object of class `sigfit`.
`object`	An object of class `sigfit`, for consistency with generic `summary` method.
`table`	Logical, whether to print the coefficients or not, defaults to `TRUE`.
`type`	Character. Either `"fit"` - to plot the fit line and the data points; or `"resid"` to plot the residuals. Defaults to `"fit"`

norm	Character. For the plot type="resid", this is the normalization type. Either "standardized" for standardized residuals, or "weighted" for weighted residuals. Defaults to "weighted".
...	Other parameters passed along to the plot() function. In print and summary methods - currently ignored.

Details

These are the most basic methods used to display sigmoidal fit objects from the sigfit function. Useful utility functions for LaTeX report templates.

Value

The functions print.sigfit and plot.sigfit are used exclusively for their side effects and do not return anything. Function summary.sigfit returns a list of the following elements:

fit	An actual summary of the nls fit.
res	A table of summary results for Standards and QCs.
stats	A table of fit statistics.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com>

See Also

sigfit and predict.sigfit.

Examples

```
## Not run:
    run = read.multiplex("your-path-here")
    fit = sigfit(run)
    fit
    summary(fit)
    plot(fit)

## End(Not run)
```

---

predict.sigfit *Predict method for "sigfit" function.*

Description

This function calculates predictions from the sigfit models in the scale of results. Major difference between this function and predict method for nls models is that predict.nls would make predictions in the scale of MFI.

Usage

```
    ## S3 method for class 'sigfit'
predict(object, newdata = NULL, e.fit = FALSE, ...)
```

Arguments

object	An object of class "sigfit" - fitted sigmoidal model.
newdata	Data for which to make predictions. If omitted (default), the predictions will be made for the standards that were used to fit the model. Otherwise, newdata can be an object of class "ima", or a regular data.frame with MFI data in the 1st column and a placeholder for calculated concentrations (predictions) in the second column; other columns are ignored. In case the first column in the data.frame is named in the fashion of "MFI.analyte-name", the "analyte-name" must match the name of the analyte in the model.
e.fit	Logical. If TRUE, fit prediction errors (in the percent scale) will be added to the result. The default is FALSE.
...	Currenly ignored.

Details

This function uses the inverse-sigmoid formula to make predictions from nls model in the scale of the result, instead of scale of "MFI".

Value

If provided with an object of class ima as an argument to newdata parameter, this function returns an object of the same type, but with the column of predictions added. Otherwise, it returns a numeric vector of predictions (for e.fit=FALSE) or a matrix of predictions and % error values (for e.fit=TRUE).

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com>

See Also

sigfit

Examples

```
## Not run:
    run = read.multiplex("your-path-here")
    fit = sigfit(run)
    predict(run)

## End(Not run)
```

---

read.kits *Function that reads and processes the kit data csv file.*

Description

This function loads the ".csv" file with kit information, sets format for columns and calculates midpoints for Quality Control sample ranges.

Usage

```
read.kits(path, file)
```

Arguments

path	Character. A path to the file with kits data.
file	Character. Filename of the kits data file.

Details

This function has been pre-defined to a very specific format of input data, and this format must be adhered to if this function is to be used. For more information on the format, see help for immunoassay.kits data frame.

Value

This function returns kits immunoassay.kits object of class data.frame.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com>

See Also

read.multiplex

Examples

```
## Not run:
    immunoassay.kits = read.kits("your-path", "file name")

## End(Not run)
```

---

read.multiplex *Load raw data from multiplex (x-MAP technology) run file*

Description

This fuction reads a multiplex run .csv file from a given folder and extracts raw data from it (MFI and counts). It returns an object of class ima which is a specifically formatted data.frame with immunoassay run parameters stored as attributes. Currently the function can process data files from "Luminex IS" and "xMap" software platforms.

Usage

```
read.multiplex(path, file, analytes = "all")
```

Arguments

path	Character. The path to the data file.
file	Character. Filename of the file to load. Does not need to contain ".csv" extension - it will be added automatically.
analytes	Either character 'analytes="all"' to load all available analytes, or a numeric vector, to select from the analytes available in the data.

Details

This function reads the raw data from multiplex run files: MFIs, COUNTs and run information, and creates a specifically formatted data frame with many run parameters stored as attributes. It can currently recognize two ".csv" file formats - the "Luminex IS" and "xMap" output. If your file doesn't load properly, please email it to the author.

Value

This function returns a structure of class ima composed of the following elements:

data	Data.frame with the actual data.
file	File name.
Assay	Character vector with the assay infromation (platform and S/N).
Kit	Character vector with kit information (Lot # and number of samples).
Analytes	Character vector of analyte names.
Units	Character vector of analyte units.
qc.ranges	Matrix that stores the qc ranges for the analytes in this data file.
Date	The date the run was processed.
Operator	Name of the analyst that processed the run.
Background	Numeric vector of background values for all analytes.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com>

See Also

sigfit and read.kits.

Examples

```
## Not run:
    run = read.multiplex("your-path-here")
    run
    plot(run)

## End(Not run)
```

`sigfit` *Fit the sigmoidal model to raw immunoassay run data.*

Description

This function fits either a 4-PL or 5-PL sigmoidal model (classic logistic model and the Hill's form) to the data from a immunoassay run - nominally an object of class `"ima"`. The user can select the model type, several types of weighting, which calibrators to use and even the starting values for the `nls` fit. This function can also automatically select best fit from user selected list, and it can remove erroneous calibration points (only one replicate) automatically to improve the fit.

Usage

```
sigfit(x, analyte = 1, model = "L.5", weights = "sqrt",
    refit = 0.2, use = 1, stvals = "adaptive", sledz=NULL)
```

Arguments

`x`	An object of class `"ima"`.
`analyte`	Numeric. Selects the analyte for which the fit is desired - from the analytes available in the data
`model`	Character vector, can be any combination of the following values: "L.4", "L.5", "H.4" and "H.5", or alternatively it can have the value `"auto"` which is equivalent to providing all four model types. "L" represent classic Logistic equation models, while "H" represent Log-Logistic (Hill form) models. "4" and "5" after the dot represent "4-PL" and "5-PL" models, respectively. Defaults to "L.5", which we found to be the most robust model.
`weights`	Character or Numeric vector. If `character` - it can be the name or combination of names of any of the five built-in weighting types, or the value of `"auto"` which is equivalent to providing a vector of all weighting type names. The weighting types are as follows: "1/y" - standard 1/y weighting; "sqrt" - square root of 1/y weights; "248" - calibrators are assigned weights of consecutive powers of 2 starting from the lowest calibrator; "123" - similarly to "248", calibrators are assigned weights of consecutive natural numbers; "none" - no weighting. If `numeric` - it must be a vector of weights of the same length as the number of calibrator replicates. Defaults to "sqrt", which according to our experience is the most robust weighting type.
`refit`	Numeric. If automatic selection of calibrator replicates is desired, this is the threshold for the in-accuracy error of the standard. If the post-hoc estimated value for any of the replicates has error above the given threshold, the function will attempt to re-fit the model without this replicate and check if this improves the fit. This can be suppressed by providing value of `0` or `NA`. Defaults to 0.2 (20% in-accuracy). Calibrators removed by this criteria show as "x" on the plot made with the `plot.sigfit` function.
`use`	A numeric vector of length one, or the exact same length as the number of calibrator replicates. It is to be composed of "0"s and "1"s and can be used to

arbitrarily select calibrator replicates to use for fitting. Use the value of `1` to turn the replicate "on" or the value of `0` to turn it "off". Calibrators turned "off" manually show as small solid dots on the plot made with the `plot.sigfit` function.
Defaults to 1, which means to use all replicates.

`stvals` Character or a list. It can have the following values: NULL - tells the function to use the default starting values for the `nls` fit; "adaptive" (`class "character"`) - tells the function to use adaptive starting values - useful when fitting more than just a few runs, the function will use medians of the coefficients of already fitted models for each analyte to derive starting values for subsequent fits. This can sometimes stabilize the behaviour of the fitting function in larger projects, and this is the default.
If a list, it must be a named list of starting values of all model parameters with their values: `"list(a=, b=, c=, d=)"` for 4-PL models and `"list(a=, b=, c=, d=, f=)"` for 5-PL models. If using the list for this parameter, currently the function will allow only one model type (either 4-PL or 5-PL), but multiple weighting types are still possible.

`sledz` Logical. This turns on internal debugging for use by the developer only. Defaults to FALSE.

Details

The `sigfit` function is the actual work horse of the whole `immunoassay` library. It does the fitting of sigmoidal models by using the `nls` function. Four models are hard-coded in it to choose from: two 4-PL and two 5-PL models, each can be either in the form of standard sigmoidal equation (logistic function), or in the logarithm (Hill's) form.

The user can either select the model and all its parameters explicitly, or can provide vectors of parameters for the function to choose from. The function will fit the models for all combinations of parameters and will select the best-fit model based on the criteria of residual standard error and R-squared. The user can also set the model and weighting parameters to `"auto"` and leave the decision on selecting the model entirely in the hands of the algorithm in this function.

The function can attempt to recognize and remove single replicates of standards that cross the threshold of in-accuracy (`refit` parameter). Of note: once the threshold is crossed by at least one replicate, the function will go through all replicates in the sequence of decreasing error (in-accuracy) and attempt to re-fit the model without them. This may sometimes result in more than one replicate removed (but always only one replicate per standard), even if only one replicate was above the threshold in the beginning. The criteria to "keep the replicate out" is currently that the overall error of the old model must be more than `1+refit` times the error for the re-fit model. Additionally, the QC criteria are checked: whether the QC predictions for the re-fit model are within the limits - if they were within those limits for the old model. If not, the removed standard is re-introduced - this is to prevent situations where automatic removing of calibrators would improve the fit but mess up the QCs.

In situations where the user is certain that some standards must be excluded from fitting, it can be done using the `use` parameter - simply put "0"s for the replicates to be removed.

If the model convergence fails with the default `nls` starting values, that are hard-coded, users are encouraged to experiment with the `"stvals"`. The starting values can be provided directly as a list, or the option `"adaptive"` can be used as well. For the `"adaptive"` starting values to work, a global list named `"immunoassay.coefs"` must be present and appropriately formatted. This list can be created manually, though the function `batch` creates it automatically if it does not exist.

Value

Function `sigfit` returns an object of class `"sigfit"`, that is a list composed of the following items:

`fit`	The actual `nls` fit object.
`data`	The `data.frame` with calibrators data, that was used for the fit.
`qcs`	Similarly, a `data.frame` with Quality Control samples data.
`model`	Character vector of the final model information, describing the model type and weighting type.
`analyte`	Character vector of analyte information: first, analyte names, then units.
`file`	Character, the name of the raw data file, for which the fit was done.
`stats`	A matrix of summary statistics of the models, if more than one model or weighting type was provided as an input.

Author(s)

Michal J. Figurski, PhD <mfigrs@gmail.com> of the Biomarker Research Laboratory, University of Pennsylvania, Philadelphia, PA.

See Also

`immunoassay` package, `batch` function and `plot.sigfit`.

Examples

```
## Not run:
    run = read.multiplex("your-path-here")
    fit = sigfit(run)
    fit
    summary(fit)
    plot(fit)

## End(Not run)
```

Index