Package 'cellGrowth'

June 24, 2019

Maintainer Julien Gagneur <gagneur@genzentrum.lmu.de>

License Artistic-2.0

Title Fitting cell population growth models

Author Julien Gagneur <gagneur@genzentrum.lmu.de>, Andreas Neudecker <a.neudecker@arcor.de>

Description This package provides functionalities for the fitting of cell population growth models on experimental OD curves.

Version 1.28.0

biocViews ImmunoOncology, CellBasedAssays, MicrotitrePlateAssay, DataImport, Visualization, TimeCourse

Date 2012-07-30

Depends R (>= 2.12.0), locfit (>= 1.5-4)

Imports lattice

Collate 'bandwidthCV.R' 'baranyi.R' 'fitCellGrowth.R' 'fitCellGrowths.R' 'getRowColumn.R' 'getWellIdsTecan.R' 'gompertz.R' 'guessCellGrowthParams.R' 'logistic.R' 'plot.cellGrowthFit.R' 'plot.well.R' 'plotPlate.R' 'readGenios.R' 'readYeastGrower.R' 'rosso.R' 'standardWellId.R' 'wellDataFrame.R'

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

git_branch RELEASE_3_9

git_last_commit cc4507c

git_last_commit_date 2019-05-02

Date/Publication 2019-06-23

R topics documented:

bandwidthCV 2
baranyi 3
fitCellGrowth 4
fitCellGrowths 5
getRowColumn 7
getWellIdsTecan 7
gompertz 8

guessCellGrowthParams 9
logistic 10
plot.cellGrowthFit 11
plot.well 11
plotPlate 12
readGenios 13
readYeastGrower 14
rosso 15
standardWellId 16
wellDataFrame 16

Index **18**

`bandwidthCV` *Bandwidth cross-validation*

Description

Perform cross-validation to detect optimal bandwidth

Usage

```
bandwidthCV(well, fileParser = readYeastGrower,
  getWellIds = getWellIdsTecan,
  bandwidths = seq(0.5 * 3600, 10 * 3600, length.out = 30),
  nFold = 10, nWell = 100, cutoff = 0.95,
  calibration = identity, scaleY = log2)
```

Arguments

`well`	well dataframe. See `wellDataFrame`.
`fileParser`	Converts the file generated by the machine to proper R format. See `readYeastGrower` for details.
`getWellIds`	function or vector. If function its parameter is the return value of fileParser. It should return a vector containing the well ids (e.g. A01, A02, ...). You can set the well ids vector directly. See `getWellIdsTecan`.
`bandwidths`	vector of bandwidths to test on
`nFold`	`integer`. In how many parts is the sample divided for cross-validation?
`nWell`	`integer`. How many wells out of the well dataframe will be used for cross validation?
`cutoff`	`scalar` between 0 and 1. See details.
`calibration`	`function` or `list` of `functions`. If function, calibration is applied to all raw data. If list, the well dataframe must contain a column `machine`. Depending on that column the according function in the list is applied to the raw data. See details
`scaleY`	`function` applied to the calibrated data.

Details

This function needs a few minutes time. The "optimal" bandwidth is the largest bandwidth which is in 95% (cutoff parameter) of all cases within one standard deviation of the best bandwidth. This should make the derivative of the fitted curve more robust. The raw values from the machine might not be directly optical densities (OD), which is needed to infer doubling time. Calibration functions for each machine can be provided to map raw values into OD using the `calibration` parameter.

Value

list with entries

`bandwidth`	"optimal" bandwidth
`well`	well dataframe
`bandwidths`	tested bandwidths
`err2`	squared error
`err2std`	Standard deviation of squared error
`muStd`	Standard deviation of max growth rate
`oneStdOfMini`	bandwidths within one std of best

Author(s)

Julien Gagneur and Andreas Neudecker

Examples

```
folder <- system.file("extdata", package="cellGrowth")
well <- wellDataFrame(file.path(folder, "plateLayout.txt"), file.path(folder,"machineRun.txt"))

## for a fast example, we use nWell = 1 here. Use a large number (default 100) for real life applications
bw <- bandwidthCV(well, nWell=1)
```

`baranyi` *Baranyi growth model*

Description

Baranyi growth model as defined in Kelly et al.

Usage

```
baranyi(x, mu, l, z0, zmax)
```

Arguments

`x`	`numeric` vector: time points for which log(OD) must be computed
`mu`	`numeric` scalar: maximal growth rate parameter
`l`	`numeric` scalar: time lag parameter
`z0`	`numeric` scalar: minimal log(OD) parameter
`zmax`	`numeric` scalar: maximal log(OD) parameter

Value

numeric vector: log(OD) for the time points given in x

Author(s)

Julien Gagneur

References

Kelly et al., The use of dummy data points when fitting bacterial growth curves, IMA Journal of Mathematics Applied in Medicine and Biology (1999) 16, 155-170

Examples

```
x = 1:1000
y = baranyi(x, mu=0.01, l=200, z0=1, zmax=5)
plot(x,y)
```

---

fitCellGrowth *Fit growth curves*

Description

Fit a cell growth curve

Usage

```
fitCellGrowth(x, z, model = "locfit",
  locfit.h = 3 * 60 * 60, locfit.deg = 2,
  relative.height.at.lag = 0.1)
```

Arguments

x	numeric vector: time points
z	numeric vector: log(OD)
model	which model to fit.
locfit.h	numeric: h parameter (window size) in call to locfit. The default value is set to three hours assuming x given in seconds. You can detect a better bandwidth by calling bandwidthCV
locfit.deg	numeric: deg parameter (polynomials degree) in call to locfit
relative.height.at.lag	Parameter used by guessCellGrowthParams

Details

For the non-parametric "locfit" model, local regression is done by a call to locfit. The returned maximum growth rate values the maximum value of the fitted derivative over the data points. For the parametric models "logistic", "gompertz", "rosso" and "baranyi", the function does a non-least square fit by calling nls. Initial parameters values are generated by guessCellGrowthParams. The returned maximum growth rate values the mu parameter of these models.

Value

Fit as returned by locfit for the "locfit" model and as returned by nls for the "logistic", "gompertz", "rosso" and "baranyi" models. The returned value also has an attribute maxGrowthRate valueing the inferred maximum growth rate as well as an attribute pointOfMaxGrowthRate valuing the datapoint at which the growth rate is maximal. Also, it has an attribute max valuing the inferred maximum among the time points as well as pointOfMax valuing the datapoint of max fitted value. It gets the additional class cellCurveFit assigned.

Author(s)

Julien Gagneur and Moritz Matthey

See Also

nls, locfit, baranyi, gompertz, logistic, rosso, guessCellGrowthParams, fitCellGrowths

Examples

```
x = 1:1000
          z = gompertz(x, mu=0.01, l=200, z0=1, zmax=5) + rnorm(length(x),sd=0.1)
          f = fitCellGrowth(x, z, model = "gompertz")
          floc = fitCellGrowth(x, z, model = "locfit", locfit.h=500)
           plot(x,z, main="simulated data\nGompertz model")
           lines(x, predict(f,x), lwd=2, col="red")
           lines(x, predict(floc,x), lwd=2, col="blue")
        legend( "right", legend=c("gompertz fit", "locfit"), lwd=1, col=c("red","blue") )
```

---

fitCellGrowths *Fit multiple growth curves*

Description

Fit growth curves for multiple wells

Usage

```
fitCellGrowths(well, plot.folder = NULL,
  model = "locfit", xlab = "time",
  ylab = expression(log2(Absorption)), scaleX = 1,
  scaleY = log2, calibration = identity,
  fileParser = readYeastGrower,
  getWellIds = getWellIdsTecan, locfit.h = 3 * 60 * 60,
  bandwidths = seq(0.5 * 3600, 10 * 3600, length.out = 30),
  nFold = 10, nWell = 100, cutoff = 0.95, ...)
```

Arguments

well	data.frame with mandatory columns directory, filename, well. See wellDataFrame
plot.folder	see details
model	model to choose for fitting growth curve

fileParser	Converts the file generated by the machine to proper R format. See readYeastGrower for details.
xlab	plot parameter
ylab	plot parameter
scaleX	useful if you want to get the doubling in another unit, e.g. days instead of seconds.
scaleY	function applied to the calibrated data.
calibration	function or list of functions. If function, calibration is applied to all raw data. If list, the well dataframe must contain a column machine. Depending on that column the according function in the list is applied to the raw data. See details
getWellIds	function or vector. If function its parameter is the return value of fileParser. It should return a vector containing the well ids (e.g. A01, A02, ...). You can set the well ids vector directly. See getWellIdsTecan.
locfit.h	bandwidth parameter for local polynomial fitting. If set to "bandwidthCV" bandwidth is automatically selected through bandwidthCV
bandwidths	passed to bandwidthCV if locfit.h="bandwidthCV"
nFold	passed to bandwidthCV if locfit.h="bandwidthCV"
nWell	passed to bandwidthCV if locfit.h="bandwidthCV"
cutoff	passed to bandwidthCV if locfit.h="bandwidthCV"
...	Parameter is passed to fitCellGrowth

Details

Essentially a wrapper for fitCellGrowth. The function gets a well object and fits a growth curve on all wells. It computes the doubling frequency observed in a well and extracts the maximal growth rate (1/minimal doubling time). The raw values from the machine might not be directly optical densities (OD), which is needed to infer doubling time. Calibration functions for each machine can be provided to map raw values into OD using the calibration parameter. If the parameter plot.folder is set, the function creates a folder within plot.folder for each file in the well object. For each well a plot is written into that folder, named well_id.png.

Value

dataframe with entries

maxGrowthRate	maximal growth rate
pointOfMaxGrowthRate	datapoint where growth rate is maximal
max	inferred maximum among the time points
pointOfMax	datapoint of the max fitted value

Author(s)

Julien Gagneur and Andreas Neudecker

See Also

fitCellGrowth

Examples

```
plateLayout <- system.file("extdata", "plateLayout.txt", package="cellGrowth")
machineRun <- system.file("extdata", "machineRun.txt", package="cellGrowth")
well <- wellDataFrame(plateLayout,machineRun)
cal <- function(x){x+1}
fit <- fitCellGrowths(well,plot.folder="data",calibration=cal)
```

---

getRowColumn *Convert well ids to row and column*

Description

Converts well ids to row and column

Usage

```
getRowColumn(wellId)
```

Arguments

wellId	vector of well ids

Value

vector of lists containing row and column

Author(s)

Andreas Neudecker

Examples

```
getRowColumn(c("A01","B05"))
```

---

getWellIdsTecan *Get aliases for wells*

Description

The aliases are generated by extracting the information from parsed data of the file generated by the tecan machine. See readYeastGrower and readGenios.

Usage

```
getWellIdsTecan(data)
```

Arguments

data	parsed data of the file. See readYeastGrower and readGenios

Value

vector containing the aliases

Author(s)

Julien Gagneur, Andreas Neudecker

Examples

```
data <- readYeastGrower( system.file("extdata", "Plate1_YPFruc.txt", package="cellGrowth"))
ids <- getWellIdsTecan(data)
```

---

gompertz *Gompertz growth model*

Description

Gompertz growth model as defined in Zwietering et al.

Usage

```
gompertz(x, mu, l, z0, zmax)
```

Arguments

x	numeric vector: time points for which log(OD) must be computed
mu	numeric scalar: maximal growth rate parameter
l	numeric scalar: time lag parameter
z0	numeric scalar: minimal log(OD) parameter
zmax	numeric scalar: maximal log(OD) parameter

Value

numeric vector: log(OD) for the time points given in x

Author(s)

Julien Gagneur

References

Zwietering, et al. Modeling of the Bacterial Growth Curve, APPLIED AND ENVIRONMENTAL MICROBIOLOGY, 1990.

Examples

```
x = 1:1000
y = gompertz(x, mu=0.01, l=200, z0=1, zmax=5)
plot(x,y)
```

guessCellGrowthParams *Guess growth models parameters*

Description

Guess initial parameters values for growth models

Usage

```
guessCellGrowthParams(x, z, relative.height.at.lag = 0.1)
```

Arguments

x	numeric vector: time points
z	codenumeric vector: log(OD)
relative.height.at.lag	numeric scalar (see Details)

Details

The relative.height.at.lag parameter should be close to the relative height of the point, where the curve reaches its maximal slope. If the fitting fails, try to set this parameter to a different value.

Value

A list with entries:

mu	numeric scalar: maximal growth rate parameter
l	numeric scalar: time lag parameter
z0	numeric scalar: minimal log(OD) parameter
zmax	numeric scalar: maximal log(OD) parameter

Author(s)

Julien Gagneur

Examples

```
x <- 1:1000
z <- gompertz(x, mu=0.01, l=200, z0=1, zmax=5)+rnorm(length(x),mean=0,sd=0.25)
guess <- guessCellGrowthParams(x,z,relative.height.at.lag=0.5)
fit <- nls(z~gompertz(x,mu,l,z0,zmax),start=guess)
plot(x,z)
lines(x,predict(fit,x),lwd=2,col="red")
```

`logistic` *Logistic growth model*

Description

Logistic growth model as defined in Zwietering et al.

Usage

```
logistic(x, mu, l, z0, zmax)
```

Arguments

`x`	`numeric` vector: time points for which log(OD) must be computed
`mu`	`numeric` scalar: maximal growth rate parameter
`l`	`numeric` scalar: time lag parameter
`z0`	`numeric` scalar: minimal log(OD) parameter
`zmax`	`numeric` scalar: maximal log(OD) parameter

Value

`numeric` vector: log(OD) for the time points given in `x`

Author(s)

Julien Gagneur

References

Zwietering, et al. Modeling of the Bacterial Growth Curve, APPLIED AND ENVIRONMENTAL MICROBIOLOGY, 1990.

Examples

```
x = 1:1000
y = logistic(x, mu=0.01, l=200, z0=1, zmax=5)
plot(x,y)
```

plot.cellGrowthFit *Generic plot function for datatype cellGrowthFit*

Description

Plot of a growth curve showing raw data and fitted curve

Usage

```
plot.cellGrowthFit(x, scaleX = 1, xlab = "time",
  ylab = "log2(OD)", lwd = 0.5, ...)
```

Arguments

x	growth curve object. See fitCellGrowth
scaleX	scalar affecting the scaling of the x-axis.
xlab	plot parameter
ylab	plot parameter
lwd	plot parameter
...	optional plot parameters passed to the plot function

Author(s)

Andreas Neudecker

Examples

```
# Parse file
dat = readYeastGrower( system.file("extdata", "Plate1_YPFruc.txt", package="cellGrowth") )

# fit
n <- names( dat$OD)[36]
fit <- fitCellGrowth(x=dat$time,z=log2(dat$OD[[n]]), model = "locfit")
plot(fit)
```

plot.well *Generic plot function for datatype well*

Description

Plots well plate as lattice xyplot.

Usage

```
plot.well(x, file = NULL, labelColumn = NULL,
  calibration = identity, ...)
```

Arguments

x	the well object
file	which plate file to plot? If NULL (default) the first file is taken.
labelColumn	column in the well object to take label for the wells from
calibration	function or list of functions. If calibration is a function it is applied to all raw data. If it is a list, the well dataframe must contain a column named machine. Depending on that column the according function in the list is applied to the raw data.
...	optional plot parameters, see details

Details

This function calls plotPlate for the plate plate. The ... parameter is passed to the plotPlate function.

Author(s)

Andreas Neudecker

---

plotPlate *Plot of a well plate*

Description

Plot of a well plate directly from a file using a lattice xyplot

Usage

```
plotPlate(file, labels = NULL,
  fileParser = readYeastGrower,
  getWellIds = getWellIdsTecan, calibration = identity,
  extractRowColumn = getRowColumn, cex = 0.05,
  scaleX = 1, scaleY = log2, strip.lines = 1.05,
  strip.cex = 0.8, xlab = "time", ylab = "log2(OD)",
  main = basename(file),
  scales = list(x = list(rot = 45)), ...)
```

Arguments

file	file name
labels	vector of characters indicating the label of the wells
fileParser	the file parser which reads the file generated by the machine
getWellIds	function or vector. If getWellIds is a function its parameter is the parsed data of the file parsed by fileParser. It should return a vector containing the well identifiers, e.g. A01, A02, .. You can as well set the well identifiers as a vector directly
calibration	calibration function applied to the raw data (before scaleY is applied)
extractRowColumn	function which converts well identifiers into row and column names

cex	plot parameter
scaleX	factor which scales the x-axis
scaleY	function how to convert the y-axis (e.g. log2)
strip.lines	height in lines of the labels
strip.cex	text-size of the labels
xlab	plot parameter
ylab	plot parameter
main	plot parameter
scales	plot parameter
...	optional plot parameter. See details

Details

All plot parameters are passed to the xyplot function

Author(s)

Andreas Neudecker

Examples

```
plotPlate( system.file("extdata", "tecan_genios.txt", package="cellGrowth"), fileParser=readGenios)
```

---

readGenios *Read Tecan Genios data files*

Description

Read raw data file form Tecan Genios instrument

Usage

```
readGenios(file)
```

Arguments

file	filename

Value

a list with entries:

time	a numeric vector of time points
OD	a data.frame vector of measured OD. The colnames are the well names.
read	a numeric vector of read numbers
temperature	a numeric vector of temperatures
header	a character vector: the header of the file

Author(s)

Julien Gagneur

See Also

readYeastGrower

Examples

```
# Get file names
# Parse file
dat = readGenios( system.file("extdata", "tecan_genios.txt", package="cellGrowth") )

# fit
n <- names( dat$OD)[36]
fit <- fitCellGrowth(x=dat$time,z=log2(dat$OD[[n]]), model = "locfit",locfit.h=6*60*60)
plot(fit)
```

---

readYeastGrower *Read Yeast Grower data files*

Description

Read raw data file from Yeast Grower software

Usage

```
readYeastGrower(file)
```

Arguments

file	filename

Value

a list with entries:

time	a numeric vector of time points
OD	a data.frame vector of measured OD. The colnames are the well names.
read	a numeric vector of read numbers
temperature	a numeric vector of temperatures
header	a character vector: the header of the file

Author(s)

Julien Gagneur

See Also

readGenios

Examples

```
# Get file names
# Parse file
dat = readYeastGrower( system.file("extdata", "Plate1_YPFruc.txt", package="cellGrowth") )

# fit
n <- names( dat$OD)[36]
fit <- fitCellGrowth(x=dat$time,z=log2(dat$OD[[n]]), model = "locfit")
plot(fit)
```

`rosso` *Rosso growth model*

Description

Rosso growth model

Usage

```
rosso(x, mu, l, z0, zmax)
```

Arguments

`x`	vector: time points for which log(OD) must be computed
`mu`	scalar: maximal growth rate parameter
`l`	scalar: time lag parameter
`z0`	scalar: minimal log(OD) parameter
`zmax`	scalar: maximal log(OD) parameter

Details

Rosso model is z0 if x<=l zmax - log(1 + (exp(zmax-z0) -1)*exp(-mu*(x-l))) otherwise

Value

vector: log(OD) for the time points given in `x`

Author(s)

Julien Gagneur

Examples

```
x = 1:1000
y = rosso(x, mu=0.01, l=200, z0=1, zmax=5)
plot(x,y)
```

standardWellId *Make standard names for well ids*

Description

Make standard names for well in 96 well plates

Usage

```
standardWellId(wellId)
```

Arguments

wellId	vector of well ids

Details

A1 -> A01 A01 -> A01

Value

standard well name

Author(s)

Julien Gagneur

Examples

```
standardWellId( c("A1", "B01", "H2"))
```

wellDataFrame *Create a well data frame*

Description

Load a plate layout file and a file specifying the machine runs

Usage

```
wellDataFrame(plateLayoutFile, machineRunFile)
```

Arguments

plateLayoutFile	a file containing the plate layout. The file must contain a column named plate and a column named well
machineRunFile	a file containing the machine runs The file must contain columns named directory, filename and plate specifying the directory and filename of the data for the corresponding run. The column use is optional. If present, only rows with use == TRUE are put into the dataframe.

Details

See the provided example files for the layout and machine run file formats.

Value

an object of class well and data.frame

Author(s)

Andreas Neudecker

Examples

```
plateLayout <-  system.file("extdata", "plateLayout.txt", package="cellGrowth")
machineRun <-  system.file("extdata", "machineRun.txt", package="cellGrowth")
well <- wellDataFrame(plateLayout,machineRun)
plot(well,plate=1)
```

Index

bandwidthCV, 2, *4*, *6*
baranyi, 3, *5*

fitCellGrowth, 4, *6*, *11*
fitCellGrowths, *5*, 5

getRowColumn, 7
getWellIdsTecan, *2*, *6*, 7
gompertz, *5*, 8
guessCellGrowthParams, *4, 5*, 9

locfit, *4, 5*
logistic, *5*, 10

nls, *4, 5*

plot.cellGrowthFit, 11
plot.well, 11
plotPlate, *12*, 12

readGenios, *7*, 13, *14*
readYeastGrower, *2*, *6, 7*, *14*, 14
rosso, *5*, 15

standardWellId, 16

wellDataFrame, *2*, *5*, 16

xyplot, *13*