Package 'qfa'

November 9, 2012

Version 0.0-3

Date 2011-10-20

Title Tools for modelling the growth dynamics of arrays of large numbers of colonies and performing quantitative fitness analysis (QFA).

Author Conor Lawless <conor.lawless@ncl.ac.uk>, with contributions from Alexander Young <alextisyoung@gmail.com> and Darren Wilkinson <d.j.wilkinson@ncl.ac.uk>

Maintainer Conor Lawless <conor.lawless@ncl.ac.uk>

Depends R (>= 2.10.1), sp, DEoptim

Suggests

Description Quantitative Fitness Analysis (QFA) is a complementary series of experimental and computational methods for estimating the fitness of thousands of microbial cultures in parallel. QFA is suitable for focussed, high-quality studies of the effect of genetic mutations or drug interventions on growth in model microbial organisms such as brewer's yeast. Culture growth is observed by time-lapse photography of solid agar plates inoculated with cultures in rectangular arrays. Growth curves are constructed by analysing image series using Colonyzer image analysis software (http://research.ncl.ac.uk/colonyzer/) which converts images to arrays of cell density estimates. This R package is for a) fitting the generalised logistic model to potentially thousands of parallel growth curves, b) using inferred parameter values to calculate fitnesses for each culture and c) comparing fitnesses between QFA experiments with different genetic backgrounds or treatments to deduce interaction strengths. This package facilitates quantifying the fitness of thousands of independent microbial strains and tracking them throughout growth curve experiments. The model is fit either by least-squares optimisation (fast: qfa), or using a hierarchical Bayesian model for all of the colonies in an experiment (CPU intensive: qfaBayes). The Bayesian fit is performed using Gibbs Sampling, as implemented by JAGS in the package rjags. With appropriately designed experiments, qfa can also estimate genetic interaction strengths and produce epistasis plots.

License Artistic-2.0

URL http://qfa.r-forge.r-project.org/

BugReports conor.lawless@ncl.ac.uk

R topics documented:

`colonyzer.read` *Reads raw cell density timecourse data from Colonyzer output files*

Description

Reads in and binds together all of the Colonyzer output files in a directory, so they are checked and ready for bayesian or likelihood inference. Colonyzer is an open source image analysis tool for quantifying cell densities on agar plates: http://research.ncl.ac.uk/colonyzer/

Usage

```
colonyzer.read(path=".",files=c(),experiment="ExptDescription.txt",
ORF2gene="",libraries="LibraryDescriptions.csv",background="")
```

Arguments

`path`	The path to the folder containing the Colonyzer .dat files to be read: working directory by default.
`files`	Character vector giving locations of Colonyzer .dat files to be read - overrides path
`experiment`	Name of text file describing the inoculation times, library and plate number for unique plates. Taken relative to path if specified. File must be a tab-delimited text file with no header containing the following columns: • Barcode - Plate identifier • Area - Culture area (pixels) • Spot Row - Plate row number for culture

- Trimmed Area - Integrated Optical Density, sum of pixel intensities in culture area
- Spot Column - Plate column number for culture
- Intensity - Total pixel intensity for square tile containing culture
- Edge Pixels - Number of pixels classified as culture on edge of square tile
- Threshold - Pixel intensity threshold used for image segmentation (after lighting correction)
- X Offset - x-oordinate of top left hand corner of culture tile (pixels)
- Y Offset - y-coordinate of top left hand corner of culture tile (pixels)
- Treatments - Temperature(s) at which cultures were grown
- Medium - Nutrients in plate agar
- Image Name - Full name at image capture (includes barcode and date-time)
- ORF Name - Array deletion y-number
- Date of Image - Date of image capture
- Screen Name - Name of screen (identifies biological repeats, and experiment)
- Library Name - Identifier for library configuration (identifies particular culture location)
- MasterPlate Number - Library plate identifier
- Timeseries order - Photograph number
- Colony Color R - Culture red channel intensity
- Colony Color G - Culture green channel intensity
- Colony Color B - Culture blue channel intensity
- Background Color R - Background red channel intensity (for current tile)
- Background Color G - Background green channel intensity (for current tile)
- Background Color B - Background blue channel intensity (for current tile)
- Edge length - Number of culture pixels classified as being microcolony edge pixels (useful for classifying contaminants)
- Tile Dimensions X - Culture tile width (pixels)
- Tile Dimensions Y - Culture tile height (pixels)

`ORF2gene` Path to a tab-delimited text file containing two columns (with no headers) associating unique, systematic strain identifiers (e.g. yeast ORF Y-numbers) with human readable gene names (e.g. standard names from SGD).

`libraries` Tab-delimited text file describing the array strains present in each row-column coordinate of each plate in a series of rectangular arrayed libraries. Header row format is: "Library ORF Plate Row Column Notes". Columns are:

- Library - Library identifier (e.g. SDLV1)
- ORF - Systematic strain identifier
- Plate - Plate number
- Row - Row number
- Column - Column number
- Notes - Optional strain notes (e.g. is strain especially sick or missing?)

`background` Experiment identifier (e.g. query mutation)

Value

An R data.frame where each row corresponds to a single observation on a single colony, with the value of the growth measurement in 'Growth', and the date and time of the measurement in 'Date.Time'. Other information about the observation is stored in the other columns. Several columns returned are direct copies of Colonyzer output and mapped as follows:

- Image.Name - Image Name
- Row - Spot Row
- Col - Spot Column
- X.Offset - X Offset
- Y.Offset - Y Offset
- Area - Area
- Trimmed - Trimmed Area
- Threshold - Threshold
- Intensity - Intensity
- Edge.Pixels - Edge Pixels
- Colony.Color.R - Colony Color R
- Colony.Color.G - Colony Color G
- Colony.Color.B - Colony Color B
- Background.Color.R - Background Color R
- Background.Color.G - Background Color G
- Background.Color.B - Background Color B
- Edge.length - Edge length
- Tile.Dimensions.X - Tile Dimensions X
- Tile.Dimensions.Y - Tile Dimensions Y

Extra columns are automatically added as follows. Some of this information is derived from auxiliary files passed to the function such as the experimental description file, the orf-gene dictionary and the library description file:

- Growth - A cell density surrogate built from trimmed Area normalised by tile area and maximum achievable pixel intensity: Trimmed/(Tile.Dimensions.X*Tile.Dimensions.Y*255)
- Barcode - Plate identifier, essentially image name with date time and file extension stripped
- Date.Time - Date time of image capture in YYYY-MM-DD_hh-mm-ss format
- Inoc.Time - Date time that plate was inoculated. If plate is grown at a high temperature, date time at which plate was moved into high temperature incubator. The assumption in this case being that negligible growth occurred before plate temperature was shifted the the target temperature.
- Treatments - Treatments applied to plate (e.g. temperature)
- Medium - Medium contained in agar (e.g. nutrients or drugs added to agar)
- Screen.Name - Unique identifier for experiment (usually identifies repeat number also if multiple repeats carried out).
- RepQuad - Identifier for experiments scaling down from 1536 format plates to 384, indicating which quadrant on the original 1536 source plate the current 384 format plate belongs to.

- MasterPlate.Number - Identifies which plate in the source library (as described in the library description file) corresponds to the current plate
- Timeseries.order - Ordinal describing which photograph captured
- Library.Name - Identifies which of the libraries identified in the library description file was used to construct this plate
- ORF - Unique systematic identifier for the genotype of the strain at this location (e.g. yeast Y-number), as defined by library description file
- Gene - Standard, human readable genotype identifier for the strain at this location, as defined by the ORF-Gene dictionary
- Background - Tag identifying experiment, typically used to construct file names and axes titles in plots
- Expt.Time - Number of days passed between inoculation (start of experiment) and current time

Finally, as well as returning the object above, this function prints a small report to screen, summarising the data returned. This includes number of unique barcodes read, number of photos read, number of genotypes in experiment, number of unique culture observations made, a list of treatments applied, a list of media used, a list of unique screen names (e.g. replicates carried out), the plate dimensions (e.g. 1536, 384 or 96 format) and a list of unique inoculation dates.

`dtl` *Culture Doubling Time for Generalised Logistic Function (as a function of time t)*

Description

Calculates the Generalised Logistic Model (http://en.wikipedia.org/wiki/Generalised_logistic_function) version of culture doubling time as a function of time t.

Usage

```
dtl(K,r,g,v,t)
```

Arguments

`K`	Culture carrying capacity (AU). Same units as (normalised) cell density observed in growth curve.
`r`	Culture growth rate parameter (per day).
`g`	Inoculum denisty (AU). Same units as (normalised) cell density observed in growth curve.
`v`	Shape parameter. Recover logistic model with v = 0.
`t`	Time since inoculation (d).

getDeadLocations *Find dead cultures in SGA plates (1536 format), and report their location in spotted plates (384 format).*

Description

Given Colonyzer quantifications of SGA plates in 1536 format, locate dead cultures and report the 384-format locations of those cultures for stripping.

Usage

```
getDeadLocations(SGAFile,SGAExpt,CutoffFrac=0.0025)
```

Arguments

SGAFile	The Colonyzer output for the final SGA plates. These are 1536 format plates, and the Colonyzer file should be a .dat tab delimited text file.
SGAExpt	Experiment description file for the SGA, linking barcode with plate number.
CutoffFrac	Optional argument for specifying the minimum value for Growth (normalised IOD) corresponding to detection of cells on 1536 plate.

Value

A data frame containing columns ROW384, COLUMN384, REP384 (repeat or quadrant identifier) and PLATE which will be useful for stripping.

Glogist *Generalised Logistic growth curve model*

Description

Calculates the Generalised Logistic Model (http://en.wikipedia.org/wiki/Generalised_logistic_function) version of culture doubling time as a function of time t.

Usage

```
Glogist(K,r,g,v,t)
```

Arguments

K	Culture carrying capacity (AU). Same units as (normalised) cell density observed in growth curve.
r	Culture growth rate parameter (per day).
g	Inoculum denisty (AU). Same units as (normalised) cell density observed in growth curve.
v	Shape parameter. Recover logistic model with v = 1.
t	Time since inoculation (d).

`loapproxfun` *Model free growth curve approximation*

Description

This is a function closure. Given a timeseries dataset (growth curve data) it returns an appropriate approximating function. If a loess smoothing span parameter appropriate for the data capture frequency (frequency of photographs) is specified, the approximating function will be a smoothed version of the data in the range of observations. For all points before the first observation, the approximating function takes the value of the first smoothed version of the data. Simiarly, beyond the final observation, the function returns the smoothed version of the data at the final timepoint. If an inappropriate span parameter is passed to this function it will return a linear interpolation approximating function instead. This can be prefarable where the loess smoother would add spurious curves to datasets with sparse observations (e.g. data captured manually 2 or 3 times per day) and should give very similar results.

Usage

```
loapproxfun(t,g,span)
```

Arguments

`t`	List of observation times.
`g`	List of cell density observations corresponding to the times in t.
`span`	Loess smoothing span. If the user specifies too small a value for a given frequency of data capture, loess smoothing will not be possible and linear interpolation will be used instead.

Value

Returns a function of time t.

Examples

```
t=c(0,1,2,3,4,5)
g=c(0,2,4,5,5,4)
# Span is too small, revert to linear interpolation
func1=loapproxfun(t,g,span=0.2)
curve(func1,0,5,xlab="Time",ylab="Cell density")
# Span is big enough
func2=loapproxfun(t,g,span=3)
curve(func2,0,5,col="red",add=TRUE)
points(t,g)
```

logist *Logistic growth curve model*

Description

Logistic Model(http://en.wikipedia.org/wiki/Logistic_function#In_ecology:_modeling_population_growth) for culture growth curves, describing variation in culture cell density with time t.

Usage

```
logist(K,r,g,t)
```

Arguments

K	Culture carrying capacity (AU). Same units as (normalised) cell density observed in growth curve.
r	Culture growth rate parameter (per day).
g	Inoculum denisty (AU). Same units as (normalised) cell density observed in growth curve.
t	Time since inoculation (d).

makeFitness *Generate QFA fitnesses*

Description

This function generates a variety of informative fitnesses from generalised logistic model parameters (K,r,g,v). It takes a data frame as generated by the qfa.fit function and appends columns for Maximum Doubling Rate (MDR), Maximum Doubling Potential (MDP), Addinall et al. style fitness (MDRMDP), Doubling Time (DT), Area Under Curve (AUC). Note that this model-based AUC is disinct from the model-free nAUC which is generated directly from observed data by the qfa.fit function. The two versions of AUC should be very similar in the vast majority of cases, however.

Usage

```
makeFitness(results,AUCLim,dtmax)
```

Arguments

results	Data frame as output by qfa.fit function.
AUCLim	AUC is calculated by integrating the generalised logistic function between t=0 and t=AUCLim. The default value is 5 days.
dtmax	Although doubling time is an attractive and popular growth phenotype, it is not well defined for dead cultures, which appear regularly in high-throughput screens (doubling time goes to infinity as growth rate goes to zero). To get around this numerical difficulty, this function sets all calculated doubling times above dtmax equal to dtmax. Default value is 25 hours.

mdp — *Maximum Doubling Potential (MDP) for Generalised Logistic Function*

Description

Calculates the Generalised Logistic Model (http://en.wikipedia.org/wiki/Generalised_logistic_function) version of the Maximum Doubling Potential as presented in Addinall et al. 2011 (http://www.plosgenetics.org/doi/pgen.1
MDP is the number of doublings undergone by the culture population from the inoculum density (g) to carrying capcity (K) throughout the experiment.

Usage

```
mdp(K,r,g,v)
```

Arguments

K	Culture carrying capacity (AU). Same units as (normalised) cell density observed in growth curve.
r	Culture growth rate parameter (per day).
g	Inoculum denisty (AU). Same units as (normalised) cell density observed in growth curve.
v	Shape parameter. Recover logistic model with v = 0.

mdr — *Maximum Doubling Rate (MDR) for Generalised Logistic Function*

Description

Calculates the Generalised Logistic Model (http://en.wikipedia.org/wiki/Generalised_logistic_function) version of the Maximum Doubling Rate as presented in Addinall et al. 2011 (http://www.plosgenetics.org/doi/pgen.10013
MDR is the inverse of the doubling time at the beginning of the experiment, where competition between cultures is at its lowest level.

Usage

```
mdr(K,r,g,v)
```

Arguments

K	Culture carrying capacity (AU). Same units as (normalised) cell density observed in growth curve.
r	Culture growth rate parameter (per day).
g	Inoculum denisty (AU). Same units as (normalised) cell density observed in growth curve.
v	Shape parameter. Recover logistic model with v = 0.

mdrmdp *Fitness value for Generalised Logistic Function*

Description

Calculates the Generalised Logistic Model (http://en.wikipedia.org/wiki/Generalised_logistic_function) version of the fitness estimate as presented in Addinall et al. 2011 (http://www.plosgenetics.org/doi/pgen.1001362). It is the product of MDR and MDP.

Usage

```
mdrmdp(K,r,g,v)
```

Arguments

K	Culture carrying capacity (AU). Same units as (normalised) cell density observed in growth curve.
r	Culture growth rate parameter (per day).
g	Inoculum denisty (AU). Same units as (normalised) cell density observed in growth curve.
v	Shape parameter. Recover logistic model with v = 0.

normalisePlates *Normalising culture fitness by plate*

Description

Sometimes estimated culture fitnesses vary systematically depending on the plate on which they are inoculated. Agar in individual plates could come from different batches, and therefore have slightly different levels of nutrients or water. Plates could be inoculated at different times, and stored at slightly different temperatures for example. Depending on inoculation method, inoculation time specified may be less accurate for individual plates. Any of these issues could effect simulated fitness slightly. This function allows us to normalise culture fitnessses across plates to eliminate such effects. It should only really be used for small differences. The technical causes of larger differences should be corrected experimentally before analysis instead of attempting to normalise them away.

Starting with a data frame describing the output from the qfa.fit function (with optional added columns from the makeFitness function) makeFitness finds all unique treatments in that data frame, calculates a median value from the indicated column for all plates subjected to that treatment and then normalises the fitnesses of each culture on each plate so that the median fitness on each plate is equal to the median fitness for all plates which have undergone the given treatment.

Usage

```
normalisePlates(d,column)
```

Arguments

d	Dataframe (output from qfa.fit) for normalisation.
column	String name of column to normalise (typically the fitness measure of interest). See qfa.fit and makeFitness help files for descriptions of available culture fitness measures.

---

pgis *Calculate strength and significance of genetic interaction.*

Description

This function carries out t-test (parametric) or Mann-Whitney test (non-parameteric) for the difference between a set of observed query strain fitnesses and the predicted query strain fitnesses, given a set of observed control strain fitnesses (see epistasis calculations in qfa.epi). It deals with non-uniqueness of replicate observations (e.g. all replicate strains are dead, giving repeated values of zero for fitness) and insufficient numbers of replicates sensibly. Too many tied observations can render tests invalid.

Usage

```
pgis(orf,m,cFs,dFs,wilcoxon)
```

Arguments

orf	Open Reading Frame (ORF) of array library gene (typically a deletion) of interest.
m	Slope of linear regression through origin fit to all available (typically genome-wide) observations.
cFs	List of available replicate observations of control strain fitnesses, labelled by ORF.
dFs	List of available replicate observations of query (or double mutant) strain fitnesses, labelled by ORF.
wilcoxon	Boolean specifying whether to use the Mann-Whitney, non-parameteric test for significance of difference between control and query strains (TRUE) or the parametric t-test (FALSE). Default is TRUE.

---

qfa.epi *Finds genetic interaction strengths and p-values*

Description

Fits a genetic independence model between control strains and double mutant strains, either using rjags and a Bayesian linear regression model, or lm and maximum likelihood. For each ORF, the probability that it is a false discovery of a suppressor or enhancer is calculated. These probabilities are then fdr corrected and returned along with genetic interaction scores.

Usage

```
qfa.epi(double,control,qthresh,orfdict="ORF2GENE.txt",
GISthresh=0.0,plot=TRUE,modcheck=TRUE,fitfunct=mdrmdp,wctest=TRUE)
```

Arguments

double	Either a qfa.posterior or the results of qfa.fit for the double mutants
control	Either a qfa.posterior or the results of qfa.fit for the control strains
qthresh	The FDR corrected cut off
orfdict	Location of file giving a column of ORFs first and a column of corresponding gene names second - so gene names can be plotted
GISthresh	When returning interaction hitlists, this variable determines the cutoff for strength of genetic interaction.
plot	If TRUE, then a 2-way fitness plot is made.
modcheck	If TRUE then diagnostic residual plots are output to "ModelCheck.pdf"
fitfunct	The name of a fitness function whose arguments are, in order, (K,r,g,v) (carrying capacity, rate and initial size of population and shape parameter for generalised logistic growth model).
wctest	If TRUE, then use the Wilcoxon test for differences in medians as a measure of statistical significance of genetic interaction. This is the default. If FALSE, then use a t-test for difference in mean fitnesses instead.

Value

Returns an R list containing three data frames: Results, Enhancers and Suppressors. Each data frame has the following columns:

- ORF - Unique strain genotype identifier (e.g. Y-number for yeast strains)
- Gene - Human readable genotype identifier
- P - p-value for significance of difference between control and query strain fitnesses
- Q - q-value for significance of difference between control and query strain fitnesses. This is FDR corrected p-value
- GIS - Genetic interaction strength. Deviation of (mean or median, depending on value of wctest) observed query strain fitness from expected fitness given control query strain fitness and a multiplicative model of genetic interaction.
- QueryFitnessSummary - Summary statistic for all available replicate observations of query strain fitness (mean or median, depending on value of wctest).
- ControlFitnessSummary - Summary statistic for all available replicate observations of control strain fitness (mean or median, depending on value of wctest).
- QuerySE - Standard error on mean of query strain fitness observations
- ControlSE - Standard error on mean of control strain fitness observations
- TestType - Type of statistical test for significant difference carried out (i.e. Wilcoxon or t-test)
- SummaryType - Type of summary statistic used for fitnesses (i.e. mean or median)
- cTreat - Treatment applied to control plates
- cMed - Medium added to agar in control plates
- cBack - Control plate background tag (experiment identifier)

- qTreat - Treatment applied to query plates
- qMed - Medium added to agar in query plates
- qBack - Query plate background tag (experiment identifier)
- Type - Type of genetic interaction observed (suppressor, enhancer, positive, negative). This is assigned for strains with abs(GIS)>GISthresh and by comparing q-value with qthresh.

`qfa.epiplot` *Makes an epistasis plot from the full results of qfa.epi*

Description

Creates a scatterplot of control fitnesses on the x-axis and query fitnesses on the y-axis, with those deemed to be hits (by FDR adjusted p-value) coloured. Essentially, this function assumes that the experiment consists of a series of paired fitness observations for a collection (typically genome-wide) of deletion mutations either single mutations (x-axis) or the same single mutation in combination with a common background or query mutation (y-axis). Fitting a linear regression to all observations (forced through the origin) and searching for significant deviations from that regression is equivalent to searching for mutations which show significant deviation from a multiplicative model of genetic interaction. Genes whose deletions deviate from this model significantly can be said to interact with the query gene.

Usage

```
qfa.epiplot(results,qthresh,fitratio=FALSE,ref.orf="YOR202W",xxlab="Control Fitness",
yylab="Query Fitness",mmain="Epistasis Plot",fmax=0)
```

Arguments

`results`	The results of interaction analysis returned by the qfa.epi function.
`qthresh`	The fdr adjusted cutoff point for determining hits.
`fitratio`	The ratio of background mutant fitness to wildtype fitness, from the genetic indepdence model. If FALSE, this is estimated from results using linear regression.
`ref.orf`	ORF for a reference strain (typically wild-type or a surrogate), whose fitness will be marked on the control and query axes of the interaction plot (horizontal and vertical blue lines). HIS3 is the default strain.
`xxlab`	x axis label
`yylab`	y axis label
`mmain`	Plot label
`fmax`	Maximum fitness range for both x-axis (control axis) and y-axis (query axis). If 0, axis ranges are automatically chosen to include all data points.

qfa.fit *Growth curve modelling*

Description

Given a series of culture density observations from colonyzer.read, this function will fit the generalised logistic growth model to timecourse observations for all colonies by least squares using either the L-BFGS-B algorithm in R's optim function, or the differential evolution, stochastic global optimisation package DEoptim. It will also calculate a numerical Area Under Curve (nAUC) fitness measure by integrating under a loess smooothed version of the dataset if there are sufficient observations or under a linear interpolation between observations if observations are too infrequent.

Usage

```
qfa.fit(d,inocguess,ORF2gene="ORF2GENE.txt",fmt,minK=0.025,detectThresh=0.0005,globalOpt=FALSE,l
```

Arguments

d	The data.frame containing the timecourse data for each colony (returned from colonyzer.read).
inocguess	The best guess for starting density of viable cells in each colony. This is the g parameter in the generalised logistic model. Typically, for dilute inoculum 384 format spotted cultures, this value cannot be observed directly by photography. inocguess should be in the same units as the values in the Growth column in d. If figG=TRUE, only values of g within the range 0.9*inocguess and 1.1*inocguess will be assessed during optimisation. Otherwise values within 0.01*inocguess and 100.0*inocguess will be assessed.
ORF2gene	The location of the text file whose first column is of the relevant ORF names and whose second column is of corresponding gene names. If human readable gene names are not important and unique strain identifiers will suffice, set to FALSE.
fmt	The date.time format that the inoculation time (Inoc.Time) and measurement times (Date.Time) are stored in
minK	The minimum value of K above which a strain is said to be alive. Strains with K optimised to lie below this value will be classified as dead, by setting r to be zero.
detectThresh	The minimum detectable cell density (or Growth value) which reliably identifies the presence of cells. Cell densities below this value are classified as noise and discarded.
globalOpt	Flag indicating whether qfa.fit should use the slower, but more robust DEoptim global optimisation functions to fit the generalised logistic model to the data, or the quicker optim function.
logTransform	Experimental flag signalling use of different objective function for optimisation. You should probably ignore this or set it to FALSE
fixG	Flag indicating whether to allow g parameter to vary over a wide or narrow range during optimisation. fixG=TRUE corresponds to narrow constraints on g.
AUCLim	Numerical AUC (nAUC) is calculated as the integral of an approximation of the growth curve between time 0 and AUCLim

STP	Time to use for "Single Time Point" fitness estimate. Defaults to 20 days (very late in growth curve) which is like carrying capacity.
modelFit	Boolean indicating whether to generate growth curve model parameters (Generalised logistic model). Model-free fitnesses (nAUC and nSTP) are generated in any case, but setting this value to FALSE disables model fitting (and can save a little time).
...	Extra arguments passed to optim

Value

R data.frame, similar to that returned by the colonyzer.read function. The major difference is that instead of a row for every cell density observation for every culture, this object summarises all timecourse density observations for each culture with fitted generalised logistic parameters and nAUC.

- Barcode - See colonyzer.read
- Row - See colonyzer.read
- Col - See colonyzer.read
- Background - See colonyzer.read
- Treatment - See colonyzer.read
- Medium - See colonyzer.read
- ORF - See colonyzer.read
- K - Generalised logistic model carrying capacity
- r - Generalised logistic model rate parameter
- g - Generalised logistic model inoculum density (referred to in vignette as g_0)
- v - Generalised logistic model shape parameter (set to 1 to recover logistic model)
- obj - Objective function value at selected optimum
- t0 - Time of first detectable cell density observation (i.e. above detectThresh)
- nAUC - Numerical Area Under Curve. This is a model-free fitness estimate.
- nSTP - Single Time Point fitness. Cell density at time STP, as estimated with approximating function. This is a model-free fitness estimate.
- d0 - Normalised cell density of first observation (be careful about condensation on plates when using this). Note this is not necessarily the density at t0.
- Screen.Name - See colonyzer.read
- Library.Name - See colonyzer.read
- MasterPlate.Number - See colonyzer.read
- Timeseries.order - See colonyzer.read
- Inoc.Time - See colonyzer.read
- TileX - See colonyzer.read
- TileY - See colonyzer.read
- XOffset - See colonyzer.read
- YOffset - See colonyzer.read
- Threshold - See colonyzer.read
- EdgeLength - See colonyzer.read
- EdgePixels - See colonyzer.read
- RepQuad - See colonyzer.read

qfa.plot — *Plots fitted model and data for all the colonies in results of qfa.fit*

Description

Produces a multipage pdf of growth curves. Each page corresponds to a single plate and growth curves are arrayed on the page according to their position on the plate. Both observations and the fitted growth curve from qfa.fit are shown for each culture. Where available, various fitness estimates are displayed for each culture, together with culture genotype. These .pdfs are useful for visually checking quality of model fit & data.

Usage

```
qfa.plot(file,results,d,fmt="%Y-%m-%d_%H-%M-%S",barcodes=c(),
master.plates=c(),treatments=c(),screen.names=c(),backgrounds=c(),
maxg=0,maxt=0,logify=FALSE)
```

Arguments

file	The file to output the plots to.
results	The output of qfa.fit which contains the fitted curve parameters of colony growth you wish to plot.
d	The original data.frame fed to qfa.fit containing all of the timecourse data
fmt	The format in which Date.Time of measurement and inoculation time are stored
barcodes	Plot only for the plates with barcodes in this character vector; all by default.
master.plates	Plot only for the plates from master.plates in this character vector; all by default.
treatments	Plot only for the plates with treatments in this character vector; all by default.
screen.names	Plot only for the plates with screen.names in this character vector; all by default.
backgrounds	Plot only for the plates with genetic background in this character vector; all by default.
maxg	Upper cell density (y-axis limit) for all growth curve plots. Default value is zero. If maxg=0, then upper fitnesses are chosen automatically to show all datapoints.
maxt	Growth curve is plotted from time t = 0 to maxt.
logify	Boolean indicating whether growth curve plots should be on a semilog scale. Cell density (y-axis) only.

report.epi — *Normalising culture fitness by plate*

Description

Outputs the results from the qfa.epi function to a tab-delimited text file together with a header describing the medium, treatment and background for the control and query experiment, whether mean or median fitness summaries are used, whether reported p-values for significance of genetic interaction strengths are the result of t-tests or Mann-Whitney tests and indicating which version of the R package was used to generate the results.

Usage

```
report.epi(results,filename)
```

Arguments

results	Dataframe describing genetic interactions (output from epi.fit) to be summarised in a text file.
filename	Path to file.

---

rod.read *Reading of ROD raw timecourse data. Deprecated.*

Description

Reads in and binds together all of the ROD output files in a directory, so they are checked and ready for bayesian or likelihood inference . Deprecated.

Usage

```
rod.read(path=".",files=c(),inoctimes="BarcodeTimes.txt",background="",
treatments=c(),barcodes=c(),master.plates=c(),screen.names=c(),ORF2gene = "")
```

Arguments

path	The path to the folder containing the ROD files to be read: working directory by default. Do not have other text files here.
files	Character vector giving locations of ROD files to be read - overrides path
inoctimes	A text file whose first column includes the barcodes in the ROD files and whose second column is the corresponding inoculation date.times. Taken relative to path if specified.
background	The genetic background of the colonies in the ROD files
treatments	Store data only for the plates with treatments in this character vecor; all by default.
barcodes	Store data only for the plates with barcodes in this character vecor; all by default.
master.plates	Store data only for the plates from master.plates in this character vecor; all by default.
screen.names	Store data only for the plates with screen.names in this character vecor; all by default.
ORF2gene	Path to a tab-delimited text file containing two columns (with no headers) associating unique, systematic strain identifiers (e.g. yeast ORF Y-numbers) with human readable gene names (e.g. standard names from SGD).

Value

An R data.frame where each row corresponds to a single observation on a single colony, with the value of the growth measurement in 'Growth', and the date and time of the measurement in 'Date.Time'. Other information about the observation is stored in the other columns.

`rod.write` *Writes a synthetic ROD-like output file to hard-drive. Deprecated.*

Description

Takes the data.frame generated by colonyzer.read, discards some columns, rearranges it, and writes data.frame to file in ROD-like format. This function is deprecated, and is only provided for backwards compatibility with other deprecated code.

Usage

```
rod.write(iman,outf)
```

Arguments

`iman`	Image analysis data.frame as returned by colonyzer.read
`outf`	Name of output file for writing data to hard-drive.

Value

Doesn't return anything, but rather writes a tab delimited text file (ROD style format) to hard-drive.

Index