Package ‘microPop’

October 13, 2022

Type Package

Title Process-Based Modelling of Microbial Populations

Version 1.6

Date 2022-02-02

Description Modelling interacting microbial populations - example applications include human gut microbiota, rumen microbiota and phytoplankton. Solves a system of ordinary differential equations to simulate microbial growth and resource uptake over time. This version contains network visualisation functions.

License GPL-3 | file LICENSE

URL https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210X.12873

Depends deSolve, visNetwork

Imports testthat, methods

Suggests rmarkdown, R.rsp, knitr, webshot

VignetteBuilder knitr

RoxygenNote 7.1.2

LazyData true

Collate 'applyTraitTradeOffs.R' 'assignNAsToMFGs.R' 'assignStrainTraits.R' 'checkResInfo.R' 'checkSolution.R' 'checkStoichiom.R' 'combineGrowthLimFuncDefault.R' 'combinePathsFuncDefault.R' 'convertFlowsToMoles.R' 'convertStatesToMoles.R' 'createDF.R' 'data.R' 'derivsDefault.R' 'entryRateFuncDefault.R' 'extraGrowthLimFuncDefault.R' 'getAllResources.R' 'getGroupName.R' 'getKeyRes.R' 'getMolarStoichiom.R' 'getMolarYields.R' 'getNonBoostFrac.R' 'getNumPaths.R' 'getPHcorners.R' 'getStrainPHcorners.R' 'getStrainParamsFromFile.R' 'getVNPlotObject.R' 'getValues.R' 'growthLimFuncDefault.R' 'makeInflowFromSoln.R' 'makeNetworkMatrices.R' 'makeParamMatrixG.R' 'makeParamMatrixS.R' 'massBalanceFuncDefault.R' 'meanTraitFunc.R' 'microPop-package.R' 'microPopModel.R' 'networkDFfromMPinput.R' 'networkDFfromMPoutput.R'

'pHFuncDefault.R' 'pHLimFuncDefault.R' 'pHcentreOfMass.R'
'plotMicrobes.R' 'plotResources.R' 'plotTraitChange.R'
'productionFuncDefault.R' 'quickPlot.R' 'quickPlot1.R'
'quickPlot2.R' 'removalRateFuncDefault.R' 'uptakeFuncDefault.R'
'rateFuncsDefault.R' 'replaceListItems.R' 'reshapeFlowMat.R'
'runMicroPopExample.R' 'subsetFunc.R' 'sumConcOverStrains.R'
'sumFlowOverStrains.R' 'sumFlowsOverPaths.R'
'waterUptakeRatio.R'

NeedsCompilation no

Author Helen Kettle [aut, cre]

Maintainer Helen Kettle <Helen.Kettle@bioss.ac.uk>

Repository CRAN

Date/Publication 2022-02-03 11:50:02 UTC

R topics documented:

`microPop-package` *Microbial Population modelling*

Description

microPop can be used to model the dynamics and interactions of microbial populations.

Author(s)

Helen Kettle

References

To be done

`Acetogens` *Acetogens dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
Acetogens
```

Format

dataframe

See Also

MFG

applyTraitTradeOffs *Internal function to trade off one trait against another (used when assigning randomly generated strain traits)*

Description

works by finding the values for each strain for par1 and par2 and then sorting them in opposite orders. This means the parameter values don't change number but they are assigned to different strains.

Usage

```
applyTraitTradeOffs(
  microbeNames,
  tradeOffParams,
  numPaths,
  numStrains,
  Pmats,
  resourceNames
)
```

Arguments

`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`tradeOffParams`	(vector of two strings) - parameters to trade off against each other
`numPaths`	Named vector. Number of paths for each microbial group
`numStrains`	Integer or vector of integers. Number of strains per group
`Pmats`	List containing lists and matrices: [[param]][[strainName]][path,rname]
`resourceNames`	Vector of strings which contains the names of the resources in the system

Value

new version of Pmats where parameter values are traded off

assignStrainTraits *Internal function to assign stochastic strain traits*

Description

Produces a random distribution of trait values where the mean is groupVal and the range is determined by strainOptions$percentTraitRange (if not pHtrait) and by maxPHshift if it is the pHtrait (see strainOptions)

Usage

```
assignStrainTraits(
  numStrains,
  groupVal,
  strainOptions,
  parName = "unspecified param",
  pHtrait = FALSE,
  gname = "None"
)
```

Arguments

`numStrains`	Integer. Number of strains per group
`groupVal`	Scalar. Group parameter value i.e. the mean parameter value
`strainOptions`	list from microPopModel inputs. Contains 'distribution' i.e. the shape of the distribution ('normal' or 'uniform'). If it is not for a pH trait and the distribution is 'normal' then its std dev is groupVal*percentRange/200, if distribution is 'uniform' then its range is groupVal*(1 +/- percentRange/100). For a pH trait, 'maxPHshift' is the max shift in pH units and 'normal' has std dev = maxPHshift/2, and 'uniform' distribution has range groupVal +/- maxPHshift;
`parName`	Name of parameter. This is only used to help with error catching
`pHtrait`	TRUE/FALSE whether or not trait is the pH trait.
`gname`	Microbe name (for indexing strainOptions$percentTraitRange)

Value

vector of values for each strain for one parameter

`Bacteroides` *Bacteroides dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
Bacteroides
```

Format

dataframe

See Also

MFG

`ButyrateProducers1` *ButyrateProducers1 dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
ButyrateProducers1
```

Format

dataframe

See Also

MFG

`ButyrateProducers2` *ButyrateProducers2 dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
ButyrateProducers2
```

Format

dataframe

See Also

MFG

`ButyrateProducers3` *ButyrateProducers3 dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
ButyrateProducers3
```

Format

dataframe

See Also

MFG

`checkResInfo` *Checks whether the all the resources needed are included in the system information file (e.g. start value, washout rate etc)*

Description

Checks whether the all the resources needed are included in the system information file (e.g. start value, washout rate etc)

Usage

```
checkResInfo(resNames, sys.data)
```

Arguments

`resNames`	Vector of strings which contains the names of the resources in the system
`sys.data`	data frame sysInfoRes i.e. resource sys info data frame

Value

nothing

checkSolution — *Checks whether the solution generated by the ODE solver contains negative values*

Description

Checks whether the solution generated by the ODE solver contains negative values

Usage

```
checkSolution(soln, tol = -0.1)
```

Arguments

`soln`	Matrix from ode solver out$solution
`tol`	tolerance

checkStoichiom — *Checks whether the stoichiometries in each MFG conserve mass within a specified tolerance If they do not then if reBalanceStoichiom=TRUE the stoichiometry will be adjusted*

Description

Checks whether the stoichiometries in each MFG conserve mass within a specified tolerance If they do not then if reBalanceStoichiom=TRUE the stoichiometry will be adjusted

Usage

```
checkStoichiom(
  stoichiom,
  Rtype,
  microbeNames,
  numPaths,
  stoiTol,
  reBalanceStoichiom = FALSE
)
```

Arguments

`stoichiom`	Array. stoichiom[gname,R,path]
`Rtype`	Resource type matrix[gname, res.name, path.name]
`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')

`numPaths`	Named vector. Number of paths for each microbial group
`stoiTol`	Scalar. tolerance i.e. if abs(prod-up)>stoiTol then warnings are given
`reBalanceStoichiom`	Logical to turn off or on rebalancing

Value

new stoichiom matrix

`combineGrowthLimFuncDefault` *combines the growth limitation functions and max growth rates to get the growth rate of strain*

Description

Returns the specific growth rate in units of inverse time

Usage

```
combineGrowthLimFuncDefault(
  strainName,
  groupName,
  pathName,
  subst,
  ess,
  boost,
  bio.sub,
  maxGrowthRate,
  growthLim,
  keyResName,
  nonBoostFrac
)
```

Arguments

`strainName`	Name of the strain that is being looped through in the ODE solver
`groupName`	Name of microbial group that is being looped through in the ODE solver
`pathName`	Name of metabolic path (e.g. path1) that is being looped through in the ODE solver
`subst`	Vector of strings giving the names of the substitutable resources for given strain, pathway
`ess`	Vector of strings giving the names of the essential resources for given strain, pathway
`boost`	Vector of strings giving the names of the boosting resources for given strain, pathway

`bio.sub`	Vector of strings giving the names of the microbial resources for given strain, pathway
`maxGrowthRate`	Vector containing maximum growth rate on each resource (named by resourceNames). If a resource is not on the pathway the value is NA
`growthLim`	Vector containing the growth limitation from each resource (named by resourceNames). If a resource is not on the pathway the value is NA
`keyResName`	String giving the name of the key resource on this pathway
`nonBoostFrac`	(scalar) Fraction of max growth achievable if boosting resource is not present but is required on this pathway

Value

(scalar) specific growth rate in units of inverse time

`combinePathsFuncDefault` *Combine microbial growth on different pathways by one microbe*

Description

Returns a vector specifying the fraction of the total microbial growth on each pathway. This function is needed to ensure that groups which have the most paths do not automatically have the most growth - i.e. need to weight the growth on each pathway.

Usage

```
combinePathsFuncDefault(
  strainName,
  groupName,
  growthRate,
  num.paths,
  pathNames
)
```

Arguments

`strainName`	Name of the strain that is being looped through in the ODE solver
`groupName`	Name of microbial group that is being looped through in the ODE solver
`growthRate`	(vector) microbial growth rate (mass per unit time) on each pathway
`num.paths`	(integer) is the number of paths for the given strain
`pathNames`	Vector of names of all metabolic paths e.g. c('path1','path2')

Value

vector specifying the fraction of the total microbial growth on each pathway

convertFlowsToMoles *convertFlowsToMoles*

Description

convert network flows from mass to moles

Usage

```
convertFlowsToMoles(allStrainNames, flow, molarMass)
```

Arguments

allStrainNames	is a vector containing the names of the microbes (strings)
flow	is the list output from reshapeFlowMat()
molarMass	is a named vector containing the molar mass for each resource e.g. out$parms$molarMass

convertStatesToMoles *convertStatesToMoles*

Description

convert network nodes from mass to moles for resources (microbes remain as mass)

Usage

```
convertStatesToMoles(nodeMass, MolarMass)
```

Arguments

nodeMass	is the value of each node in the network (named vector)
MolarMass	is a named vector containing the molar mass for each resource e.g. out$parms$molarMass

`createDF` *Create a dataframe from a CSV file*

Description

Create a dataframe from a CSV file

Usage

```
createDF(filename)
```

Arguments

`filename`	A string containing the path to the csv file

Value

A dataframe

`derivsDefault` *Differential Equations called by ODE solver*

Description

Differential Equations called by ODE solver

Usage

```
derivsDefault(t, y, parms)
```

Arguments

`t`	time
`y`	vector of state variables
`parms`	list of parameters

`entryRateFuncDefault` *entry Rate Function*

Description

Return the rate of entry to the system for any state variable

Usage

```
entryRateFuncDefault(
  varName,
  varValue,
  stateVarValues,
  time,
  inflowRate,
  parms
)
```

Arguments

`varName`	(string) Name of state variable of interest (resource name or strain name)
`varValue`	(scalar) value of state variable of interest
`stateVarValues`	(named vector) values of all state variables
`time`	(scalar) time
`inflowRate`	(named vector) on inflow rates (specified in SysInfo files)
`parms`	List containing all system parameters

Value

(scalar) rate of entry (quantity per unit time) for any state variable

`extraGrowthLimFuncDefault` *Extra Growth Limitation Function*

Description

Return the value of extraGrowthLim (number between 0 and 1)

Usage

```
extraGrowthLimFuncDefault(
  strainName,
  groupName,
  pathName,
  stateVarValues,
  stateVarNames,
  time,
  parms
)
```

Arguments

`strainName`	Name of strain
`groupName`	Name of group
`pathName`	metabolic path name e.g. 'path1'
`stateVarValues`	values of all state variables at the current time step
`stateVarNames`	names of all state variables
`time`	time,t, in ODE solver
`parms`	list of all parameters

Value

(scalar) limitation on growth (between 0 and 1)

`getAllResources` *Makes vector of unique resource names*

Description

Makes vector of unique resource names

Usage

```
getAllResources(microbeNames, gutModel = FALSE, myPars = NULL)
```

Arguments

`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`gutModel`	Logical. TRUE if using with the microPopGut package
`myPars`	list of extra parameters

Value

vector of resource names

getGroupName *Convert strain name to its group name e.g. 'Bacteroides.1' becomes 'Bacteroides' updated (Dec 2019) so that MFG names can contain dots*

Description

Convert strain name to its group name e.g. 'Bacteroides.1' becomes 'Bacteroides' updated (Dec 2019) so that MFG names can contain dots

Usage

```
getGroupName(xname, microbeNames)
```

Arguments

`xname`	a string (may be strain name or something else)
`microbeNames`	vector of strings of microbial group names

Value

group name (string) if xname is a strain name. If xname is not a the name of a strain it will simply return xname unchanged.

getKeyRes *Finds the name of the key resource for each path for each MFG*

Description

Finds the name of the key resource for each path for each MFG

Usage

```
getKeyRes(microbeNames, numPaths)
```

Arguments

`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`numPaths`	Named vector. Number of paths for each microbial group. Names are microbe-Names

Value

list of vectors where the names are microbeNames

getNonBoostFrac — *obtains the none boosting fraction of growth for given MFG if there is a boosting resource*

Description

obtains the none boosting fraction of growth for given MFG if there is a boosting resource

Usage

```
getNonBoostFrac(microbeNames, resourceNames, numPaths)
```

Arguments

`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`resourceNames`	Vector of strings which contains the names of the resources in the system
`numPaths`	Named vector. Number of paths for each microbial group

Value

an array with format [group,resource,path]

getNumPaths — *get the number of metabolic pathways for the given group*

Description

get the number of metabolic pathways for the given group

Usage

```
getNumPaths(microbeNames)
```

Arguments

`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')

Value

a named vector of the number of paths for each group if numPathways is not in dataframe then it is set to 1.

`getPHcorners` *get pH corners Function*

Description

Returns the values of the pH values of the limit function i.e. where the limit is c(0,1,1,0) Reads these in from the microbe group dataframes

Usage

```
getPHcorners(microbeNames, pHLimit)
```

Arguments

`microbeNames`	(vector of strings). Names of microbes in the system
`pHLimit`	(logical) Is microbial growth affected by pH?

Value

(matrix) values of the pH values of the limit function i.e. where the limit is c(0,1,1,0). Row names are microbeNames

`getStrainParamsFromFile`
get strain parameter values from a csv file

Description

get strain parameter values from a csv file

Usage

```
getStrainParamsFromFile(Pmats, strainPHcorners, strainOptions)
```

Arguments

`Pmats`	List of parameter matrices
`strainPHcorners`	Matrix of pH corners for each strain
`strainOptions`	List which is input to microPopModel

Value

(list) - first entry is new version of Pmats, second is new version of strainPHcorners

---

`getStrainPHcorners` *get stochastically generated pH corners for each strain*

Description

Returns the values of the pH values of the limit function i.e. where the limit is c(0,1,1,0) Reads these in from the microbe group dataframes

Usage

```
getStrainPHcorners(
  microbeNames,
  allStrainNames,
  numStrains,
  pHcorners,
  pHLimit,
  strainOptions,
  oneStrainRandomParams
)
```

Arguments

`microbeNames`	(vector of strings). Names of microbes in the system
`allStrainNames`	(vector of strings)
`numStrains`	Integer or named vector of integers
`pHcorners`	vector of 4 scalars definining the pH lim func
`pHLimit`	(logical) Is microbial growth affected by pH?
`strainOptions`	list from microPopModel inputs
`oneStrainRandomParams`	logical from microPopModel inputs

Value

(matrix) values of the pH values of the limit function i.e. where the limit is c(0,1,1,0) for each strain

---

`getValues` *get system quantity (e.g. startValue, inflowRate, washOut) for all state variables (convention is that microbes are before resources)*

Description

get system quantity (e.g. startValue, inflowRate, washOut) for all state variables (convention is that microbes are before resources)

Usage

```
getValues(
  sysInfoMicrobes,
  sysInfoRes,
  stateVarNames,
  quantity,
  strainNames,
  microbeNames,
  resourceNames,
  numStrains
)
```

Arguments

`sysInfoMicrobes`	sys info dataframe for microbes
`sysInfoRes`	sys info dataframe for resources
`stateVarNames`	Vector of names of all the state variables
`quantity`	String. Name of quantity to get value for e.g. 'startValue'
`strainNames`	Vector of strings of strain names
`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`resourceNames`	Vector of strings which contains the names of the resources in the system
`numStrains`	Integer. Number of strains per group

`getVNPlotObject` *getVNPlotObject*

Description

uses visNetwork to produce an interactive network plot based on the links and edges dataframes

Usage

```
getVNPlotObject(
  nodes,
  edges,
  addLegend = FALSE,
  addExport = TRUE,
  figType = "png",
  mainTitle = NULL,
  subTitle = NULL,
  layoutSeed = NA,
  scaleNodes = FALSE,
  scaleEdges = FALSE,
  microbeCol = "gold",
  resourceCol = "lightblue",
  productionCol = "magenta",
  uptakeCol = "darkgrey",
  figWidth = 700,
  figHeight = 700
)
```

Arguments

`nodes`	data frame or a list with nodes information. Needs at least column "id". See visNetwork::visNodes
`edges`	data frame or a list with edges information. Needs at least columns "from" and "to". See visNetwork::visEdges
`addLegend`	Logical. If true adds a legend to plot. Default is FALSE
`addExport`	Logical. If true adds button to export fig from html plot
`figType`	Type of export. One of "png" (default), "jpeg" or "pdf". Puts a button on the html plot
`mainTitle`	Optional list containing "text" (string for plot title) and "style" (e.g. 'font-family:Times','font-family:Arial' etc).
`subTitle`	Optional list containing "text" (string for plot subtitle) and "style" (e.g. 'font-family:Times','font-family:Arial' etc)
`layoutSeed`	: NA. Random seed for the layout of the plot. To get identical plots set this to a number
`scaleNodes`	Logical. If true the node sizes differ with concentration (in moles for resources and mass or concentration for microbes)
`scaleEdges`	Logical. If true the edge sizes differ with the amount of moles flowing through them
`microbeCol`	String for microbe node colour. Default is 'orange'
`resourceCol`	String for resource node colour. Default is 'lightBlue'
`productionCol`	String for production edge colour. Default is 'darkGrey'
`uptakeCol`	String for uptake edge colour. Default is 'magenta'
`figWidth`	numeric value to control size of plotting window. Default is 700
`figHeight`	numeric value to control size of plotting window. Default is 700

Value

a visNetwork object that can be shown using print() function.

growthLimFuncDefault *growth rate limitation function*

Description

Returns the value of growthLim (must lie in interval [0,1] i.e. unitless) of strainName on varName which is used to scale the maximum growth rate Contains two options - one for essential resources and one for substitutable resources (based on Ballyk and Wolkowicz, 1993)

Usage

```
growthLimFuncDefault(
  strainName,
  groupName,
  pathName,
  varName,
  resourceValues,
  allSubType,
  strainHalfSat,
  stateVarValues,
  parms
)
```

Arguments

strainName	Name of the strain that is being looped through in the ODE solver
groupName	Name of microbial group that is being looped through in the ODE solver
pathName	Name of metabolic path (e.g. path1) that is being looped through in the ODE solver
varName	(string) Name of variable (resource) of interest
resourceValues	State vector of resources (with names)
allSubType	Vector of strings (with names corresponding to the resourceNames) which describes the type of each resource ('Rtype') - Rtypes are S (substitutable resource), Se (essential resource), Sb (booster resource), Sm (microbial resource), P (product) and Pb (biomass product)
strainHalfSat	Vector (with names corresponding to the resourceNames) of half-saturation constants for the given strain. If resource is not a substrate for the given strain, the value is NA
stateVarValues	State vector (resources and microbes) (with names)
parms	list of parameter values

Value

scalar giving limitation on growth rate - must be >=0 and <=1

`LactateProducers` *LactateProducers dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
LactateProducers
```

Format

dataframe

See Also

MFG

`makeInflowFromSoln` *Used for running microPop with multiple compartments Takes the solution (state of system) from the previous compartment (out$solution) and then finds the washout rate of each state variable using removalRateFunc to find the inflow rate to the next downstream compartment*

Description

Used for running microPop with multiple compartments Takes the solution (state of system) from the previous compartment (out$solution) and then finds the washout rate of each state variable using removalRateFunc to find the inflow rate to the next downstream compartment

Usage

```
makeInflowFromSoln(out)
```

Arguments

`out`	output from microPopModel()

Value

matrix of flow rates (conc/time) with named columns (the same as out$solution)

makeNetworkMatrices *makeNetworkMatrices*

Description

make links and nodes matrices for use in network plotting software

Usage

```
makeNetworkMatrices(
  chosen.time,
  out,
  convertToMoles = TRUE,
  sumOverStrains = TRUE
)
```

Arguments

`chosen.time`	the time you want to plot
`out`	the output from microPopModel()
`convertToMoles`	Logical. Default is TRUE
`sumOverStrains`	Logical. Default is TRUE

massBalanceFuncDefault *mass balance Function*

Description

Doesn't return anything but prints to screen if mass does not balance after the equations for biological growth have been derived This is only run if checkMassConv is TRUE

Usage

```
massBalanceFuncDefault(uptake, production, growthRate, balanceTol, strainName)
```

Arguments

`uptake`	Matrix (with names) where columns are resources and rows are pathways, giving uptake rate (mass/time) of given strain
`production`	Matrix (with names) where columns are resources and rows are pathways, giving production rate (mass/time) of given strain
`growthRate`	(vector) microbial growth rate (mass per unit time) for one strain on each metabolic pathway
`balanceTol`	(scalar) Defined in microPopModel input list checkingOptions
`strainName`	(string) Name of strain in ODE solver loop

meanTraitFunc *calculate the mean trait at the end of the model run*

Description

calculate the mean trait at the end of the model run

Usage

```
meanTraitFunc(out, trait.name, gname, resource.name, path)
```

Arguments

`out`	Output from microPopModel()
`trait.name`	can be 'halfSat','yield','maxGrowthRate' and 'pHtrait' or 'strainpHcorners'
`gname`	name of group or microbe
`resource.name`	String
`path`	String

Methanogens *Methanogens dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
Methanogens
```

Format

dataframe

See Also

MFG

`MFG` *Microbial Functional Group (MFG) dataframes*

Description

This is a generic description of the dataframes describing the pathways and parameters of each microbial functional group. Each resource (substrate, metabolic product or biomass (if microbial production is included in the chemical stoichiometry)) has a column. The first column can be used for describing the units of each parameter. This is optional and just for clarity - it is not used within microPop (note, the units column must be labelled 'units' and it can not contain NAs). The row names and their details are given below:

- Rtype Describes the type of resource. Can be S (substitutable substrate), Se (essential substrate), Sb (boosting substrate), Sm (microbial substrate), Sw (water as a substrate), P (product), Pb (biomass product) or X (not used)
- halfSat Half-saturation constant for Monod Equation growth. Units must match the units of the resources. Resources that aren't used for growth will have entry NA.
- yield This is the biomass yield i.e. mass of microbes/mass of substrate consumed. Note this is NOT a mol/mol yield! Resources that aren't used for growth will have entry NA.
- maxGrowthRate Maximum growth rate of the group. Units are per unit time where time has the same units as those used for the microPopModel input arguments 'times'. Resources that aren't used for growth must have entry NA.
- stoichiom The chemical stoichiometry in moles of each resource (note that this may also include biomass (see Xsu)).
- keyResource If the stoichiometry is specified and all resources are essential then stoichiom will be used to determine rates of production and uptake and now 'yield' is the biomass produced per gram of the key resource specified here.
- pHcorners Specified using 4 values in the first 4 columns. The pH limitation on growth is described by a trapezium. For increasing pH values the limitation goes from 0,1,1,0 at the points specified by the pHcorners.
- numPathways The number of metabolic pathways the group has. If this is greater than 1 see details below for naming conventions.

Usage

```
MFG
```

Format

A dataframe with the row names in the itemised list below and a column for units (optional) and for each resource required by the microbial group.

Details

If there is more than one pathway the row names are as above but followed by .2 for second pathway, .3 for third pathway and so on. E.g. halfSat.2, yield.2

Note, when constructing new dataframes for new microbial functional groups (MFGs), the order of the rows does not matter but the names of the rows must be the same as those above. Also, the order of the resources columns does not matter (although if there is a 'units' column it must be the first column). The resources may be different for each MFG (e.g. See Bacteroides and Xsu).

When the user tells microPop which groups to use via the microbeNames input argument, the package will determine the names of all the resources and MFGs in the system and then check they are also in the system information files.

Note that the optional units column can not contain NAs. For entries without units put 'none'.

`microbeSysInfo` *microbeSysInfo*

Description

Data frame describing the system information for the microbial state variables

Usage

```
microbeSysInfo
```

Format

A dataframe with the row names in the itemised list below and a column for units (optional) and for each microbial functional group (MFG) in the system to be simulated.

Details

Each MFG has a column. The first column can be used for describing the units of each variable. This is optional and just for clarity - it is not used within microPop (note, the units column must be labelled 'units'). The data frame must contain the following rows:

- startValue The value of each MFG at the start time of the simulation (e.g. units are g/l)
- inflowRate The value of the rate of inflow of each MFG (e.g. units are g/l/d)
- washOut The specific washout rate of each MFG (e.g. units are /d)

`microbeSysInfoHuman` *microbeSysInfoHuman dataframe*

Description

Table of information describing the inflows, outflows, start values of each microbial group for the R script microPop/inst/DemoFiles/human*.R See help(microbeSysInfo) or for an explanation of the contents

Usage

```
microbeSysInfoHuman
```

Format

dataframe

See Also

microbeSysInfo

`microbeSysInfoRumen` *microbeSysInfoRumen dataframe*

Description

Table of information describing the inflows, outflows, start values of each microbial group for the R script microPop/inst/DemoFiles/rumen.R See help(microbeSysInfo) or for an explanation of the contents

Usage

```
microbeSysInfoRumen
```

Format

dataframe

See Also

microbeSysInfo

microPopModel *Runs the microbial population model*

Description

creates a system of ordinary differential equations and solves them

Usage

```
microPopModel(
  microbeNames,
  times,
  resourceSysInfo,
  microbeSysInfo,
  rateFuncs = rateFuncsDefault,
  odeFunc = derivsDefault,
  numStrains = 1,
  oneStrainRandomParams = FALSE,
  pHLimit = FALSE,
  pHVal = NA,
  plotOptions = list(),
  odeOptions = list(),
  strainOptions = list(),
  checkingOptions = list(),
  microbeMolarMass = 113,
  bacCutOff = 1e-14,
  networkAnalysis = FALSE,
  myPars = NULL,
  ...
)
```

Arguments

`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens'). A dataframe for each of the same name must also exist in the workspace.
`times`	Vector of times at which the solution is required, e.g. seq(0,10,0.1)
`resourceSysInfo`	String giving the name of a csv file or a dataframe object, which describes the initial conditions, inflow and outflow (if constant) and molar mass of each resource. See help(resourceSysInfo) for more info.
`microbeSysInfo`	String giving the name of a csv file (e.g. 'systemInfoMicrobes.csv') or a dataframe object, which describes the initial conditions, inflow and outflow (if constant) of each microbial group. See help(microbeSysInfo) for more info.
`rateFuncs`	A list of functions which are used to solve the ODEs in odeFunc. Default is rateFuncsDefault.R (provided in the package). See ?rateFuncs

`odeFunc`	The function the ODE solver will use - the default is derivsDefault provided by the package but if the user wants to make significant changes a new ODE function file can be used. See ?derivsDefault
`numStrains`	Integer (or named vector of integers) stating the number of strains in each microbial group. If this is a single number it is the same for all groups. If it is a vector it must be named using microbeNames. Default is 1.
`oneStrainRandomParams`	Logical to allow randomization of parameters even if there is only one strain. The default is FALSE which means that if numStrains=1 then the group params are used; if numStrains>1 then the parameters are automatically randomised according to info given in strainOptions. If oneStrainRandomParams=TRUE then even if there is only one strain its parameters will be randomised according to info given in strainOptions.
`pHLimit`	TRUE if pH limits microbial growth rates. Default is FALSE. If TRUE then rateFuncs$pHLimFunc is called.
`pHVal`	Scalar. If the pH value is fixed it can be specified here and this is then used in the default rateFuncs$pHFunc function.
`plotOptions`	List containing instructions for plotting: Default is list(plotFig=TRUE, sumOverStrains=FALSE, resourceLegendPosition="topleft", microbeLegendPosition="topleft", saveFig=FALSE, figType='eps', figName='microPopFig', yLabel='Concentration (g/L)', xLabel='Time'). To turn off plot generation set plotFig=FALSE. If there are multiple strains these are all plotted if sumOverStrains=FALSE, otherwise they will be summed over each group. To save plot, saveFig=TRUE, figType (format) can be 'eps','png', 'pdf' or 'tiff' and is specified in figType (string), the name is figName (string) to which the string 'Microbes' or 'Resources' will be added for the respective plots.
`odeOptions`	List containing instructions for the ODE solver ('deSolve'). Default: list('atol'=1e-6,'rtol'=1e-6,'method'='lsoda'). See ?ode for more details.
`strainOptions`	List containing instructions for specifying strain parameters. Default: list(randomParams=c('halfSat', 'yield', 'maxGrowthRate', 'pHtrait'), seed=1, distribution='uniform', percentTraitRange=0, maxPHshift=0, applyTradeOffs=FALSE, tradeOffParams=NULL, paramsSpecified=FALSE, paramDataName=NULL). • randomParams (vector) specifying which parameters need to be stochastically generated. • seed (number) seed for random number generator. • distribution (string) - either 'uniform' or 'normal' specifying the shape of the distribution from which to draw the random strain parameters. • percentTraitRange (single number or named vector of numbers) this is the percentage either side of the group parameter value which the strain parameter may range e.g. if percentTraitRange=10 then range is 0.9x to 1.1x for group mean x. This can be specified for each microbial data file in microbeNames using a named vector, however, if only one number is given it is assumed to apply to all microbes. • maxPHshift (number) pH units to range over (either one value which is applied to all microbe groups or a named vector with a value for each group and microbeNames for its names).

- applyTradeOffs (logical) to trade off ‘good’ and ‘bad’ parameter values.
- tradeOffParams (vector of two strings) - parameters to trade off against each other. Note that pHtrait can not be traded off as whether this trait is good or bad depends on the environmental pH.
- paramsSpecified (logical) TRUE if strain parameters are read in from a file (whose name is specified in paramDataName). The file must have colnames c(strainName, paramName, paramVal, paramUnit, resource,path) and where strainName is in format ’groupName.i’ where i is the strain number.

`checkingOptions`
(List) Default is list(checkMassConv=FALSE, balanceTol=1e-2, reBalanceStoichiom=FALSE, stoiTol=0.1, checkForNegs=TRUE, negTol=-1e-2).

- checkMassConv=TRUE checks for mass conservation in the ODE solver with a tolerance of ’balanceTol’ (default is FALSE).
- reBalanceStoichiom will check the mass balance of the stoichiometries on every metabolic path and rebalance if these are not conserving mass within a tolerance of stoiTol (a warning message will be issued). Rebalancing will only affect the final solution if the pathway contains only essential resources (Rtype ’Se’) and microbial biomass is a product (Rtype ’Pb’).
- checkForNegs If TRUE the function checkSolution is called and the solution for each variable, x, is checked for negative values that are greater in magnitude than negTol*max(x). If negative values occur then the solution is incorect and either the problem is incorrectly specified or the tolerances in the ODE solver need to be smaller.

`microbeMolarMass`
Scalar. Mass of 1 mole of microbes - default is 113g/mol (Batstone et al., 2002)

`bacCutOff` Scalar. Amount of bacteria below which the bacteria are considered to have left the system and can’t grow, default =1e-14. If this is set to zero then bacteria will always be able to grow again as zero is never reached.

`networkAnalysis`
Logical. If you want to use the network analysis functions on your model results set as TRUE (default is FALSE)

`myPars` List containing extra parameter values - used if gutModel is TRUE i.e. with microPopGut package

`...` Add your own input arguments

Value

The output is a list containing a matrix called ’solution’ where rows are points in time and the columns are the state variables, and another list called parms which contains all the information needed to run the model. Within parms there are a number of other lists (e.g. Pmats for parameter values and Smats for system settings etc - try names(out$parms)).

Examples

```
#simplest example - define one microbial group (Archea) with 4 resources and
#simulate growth over 50 days
```

```
#make microbial group data frame:
MFG=matrix(NA,ncol=4,nrow=6,dimnames=list(c('Rtype','halfSat','yield',
'maxGrowthRate','stoichiom','keyResource'),c('H2','CO2','CH4','H2O')))
MFG['Rtype',]=c('Se','Se','P','P')
MFG['halfSat',c('H2','CO2')]=1e-6
MFG['yield','H2']=0.2
MFG['maxGrowthRate','H2']=2
MFG['keyResource',1]='H2'
MFG['stoichiom',]=c(4,1,1,2)
Archea=data.frame(MFG,stringsAsFactors=FALSE)

#make resourceSysInfo data frame
Rmat=matrix(NA,ncol=4,nrow=4,dimnames=list(c('startValue','inflowRate',
'washOut','molarMass'),c('H2','CO2','CH4','H2O')))
Rmat['startValue',]=c(1,1,0,0)
Rmat['inflowRate',]=c(1,5,0,0)
Rmat['washOut',]=c(0.1,0.1,0.1,0.1)
Rmat['molarMass',]=c(2,44,16,18)

#make microbeSysInfo data frame
Mmat=matrix(NA,ncol=1,nrow=3,dimnames=list(c('startValue','inflowRate',
'washOut'),c('Archea')))
Mmat['startValue',]=1
Mmat['inflowRate',]=0
Mmat['washOut',]=0.1

out=microPopModel(
    microbeNames='Archea',
    times=seq(0,50,0.1),
    resourceSysInfo=data.frame(Rmat,stringsAsFactors=FALSE),
    microbeSysInfo=data.frame(Mmat,stringsAsFactors=FALSE)
)
```

`networkDFfromMPinput` *networkDFfromMPinput*

Description

make node and edge data frames to use in visNetwork from microPop microbial data frames

Usage

```
networkDFfromMPinput(microbeNames)
```

Arguments

`microbeNames`	vector of strings of the names of the microbial data frames you want to plot. These can be intrinsic data frames or loaded in by user.

Value

a list containing the edges and nodes

`networkDFfromMPoutput` *networkDFfromMPoutput*

Description

make node and edge data frames from microPop output to use in visNetwork

Usage

```
networkDFfromMPoutput(
  chosen.time,
  MPoutput,
  groupNames = NULL,
  sumOverPaths = TRUE,
  sumOverStrains = TRUE,
  convertToMoles = TRUE
)
```

Arguments

`chosen.time`	the time you want to plot
`MPoutput`	the output from microPopModel()
`groupNames`	Default is NULL which plots all the microbes. To plot a subset of all the groups, specify a vector of strings of the names of the groups you want to plot.
`sumOverPaths`	Logical. Default is TRUE which sums flows between the same nodes even if they are on different metabolic paths
`sumOverStrains`	Logical. Default is TRUE which means the strains are put into their functional group nodes and the flow are summed. When it is FALSE, each strain will have its own node.
`convertToMoles`	Logical. Default is TRUE

Value

a list containing the edges and nodes

`NoButyFibreDeg` *NoButyFibreDeg dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
NoButyFibreDeg
```

Format

dataframe

See Also

MFG

`NoButyStarchDeg` *NoButyStarchDeg dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
NoButyStarchDeg
```

Format

dataframe

See Also

MFG

`pHcentreOfMass` *Find the pH value which is the centre of mass of the pH limitation function (used for the pH trait)*

Description

Find the pH value which is the centre of mass of the pH limitation function (used for the pH trait)

Usage

```
pHcentreOfMass(strainName, groupName, pHLimFunc, parms)
```

Arguments

`strainName`	Name of the strain
`groupName`	Name of microbial group
`pHLimFunc`	function specified in rateFuncs$pHLimFunc
`parms`	List of all parameters

Value

pH value at centre of mass

`pHFuncDefault` *pH Function*

Description

Return the value of pH in pH units

Usage

```
pHFuncDefault(time, parms, stateVarValues = NULL)
```

Arguments

`time`	(scalar). The current time point in the ODE solver.
`parms`	List which contains all information required by the ODE solver
`stateVarValues`	State vector (resources and microbes) (with names)

Value

(scalar) pH at the given time

pHLimFuncDefault *pH Limitation Function*

Description

Return the value of pHLim (must lie in interval [0,1])

Usage

```
pHLimFuncDefault(strainName, groupName, pH, parms)
```

Arguments

strainName	Name of the strain that is being looped through in the ODE solver
groupName	Name of microbial group that is being looped through in the ODE solver
pH	(scalar). The current pH value.
parms	List of all parameters

Value

(scalar) pH limitation (0 to 1)

plotMicrobes *Generic plotting of microbes over time*

Description

Generic plotting of microbes over time

Usage

```
plotMicrobes(
  out,
  sumOverStrains = TRUE,
  yLabel = "Concentration",
  xLabel = "Time",
  legendPosition = "topleft",
  cex.title = 1,
  cex.ax = 1,
  cex.legend = 1
)
```

Arguments

`out`	output from microPopModel()
`sumOverStrains`	Logical. Default=TRUE
`yLabel`	String for y axis label. Default is 'Concentration'
`xLabel`	String for x axis label. Default is 'Time'
`legendPosition`	String. Position of legend in microbe plot, default is 'topleft'
`cex.title`	Scaling for title text
`cex.ax`	Scaling for axes text (labels and ticklabels)
`cex.legend`	Scaling for legend text

Value

Nothing just generates a plot

`plotResources` *Generic plotting of resources over time*

Description

Generic plotting of resources over time

Usage

```
plotResources(
  out,
  yLabel = "Concentration",
  xLabel = "Time",
  legendPosition = "topleft",
  cex.title = 1,
  cex.ax = 1,
  cex.legend = 1
)
```

Arguments

`out`	output from microPopModel()
`yLabel`	String for y axis label. Default is 'Concentration'
`xLabel`	String for x axis label. Default is 'Time'
`legendPosition`	String. Position of legend in resource plot, default is 'topleft'
`cex.title`	Scaling for title text
`cex.ax`	Scaling for axes text (labels and ticklabels)
`cex.legend`	Scaling for legend text

Value

Nothing just generates a plot

plotTraitChange *plot changes in trait over time*

Description

plot changes in trait over time

Usage

```
plotTraitChange(
  out,
  trait.name,
  group.names,
  resource.name = NULL,
  path = NULL,
  xlabel = "Time (days)",
  saveFig = FALSE,
  figType = "eps",
  figName = "Traits"
)
```

Arguments

out	Output from microPopModel()
trait.name	can be 'halfSat','yield','maxGrowthRate' and 'pHtrait' or 'strainpHcorners'
group.names	can be a vector of group names or just one string for one name
resource.name	String
path	String
xlabel	String
saveFig	Logical
figType	String
figName	String

productionFuncDefault *Production Function*

Description

Production rate of resource (units are resource mass/time)

Usage

```
productionFuncDefault(
  strainName,
  groupName,
  pathName,
  varName,
  all.substrates,
  keyResName,
  stoichiom,
  products,
  bio.products,
  uptake,
  growthRate,
  yield,
  parms,
  water
)
```

Arguments

`strainName`	Name of the strain that is being looped through in the ODE solver
`groupName`	Name of microbial group that is being looped through in the ODE solver
`pathName`	Name of metabolic path (e.g. path1) that is being looped through in the ODE solver
`varName`	(string). Calculate production of this variable
`all.substrates`	Vector of strings giving the names of the all the substrates used on this pathway
`keyResName`	(string). Name of the key resource on this pathway
`stoichiom`	Named vector (names are resourceNames) giving the mass of each resource in the stoichiometry i.e. molar mass of resource multiplied by the number of moles in the stoichiometry
`products`	Vector of strings giving the names of the all the metabolic products created on this pathway
`bio.products`	Vector of strings giving the names of the all the microbial products created on this pathway
`uptake`	Vector with names given by resourceNames which given mass uptake of each resource per unit time
`growthRate`	(scalar) microbial growth rate (mass per unit time) on the given pathway
`yield`	Named vector (names are resourceNames) giving the mass yield of biomass on each resource (mass microbe/mass resource)
`parms`	List containing all system parameters
`water`	Name of resource with Rtype ’Sw’ - i.e resource could be called ’water’ or ’H2O’ etc

Value

(scalar) production rate of given resource (units are resource mass/time)

`PropionateProducers` *PropionateProducers dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
PropionateProducers
```

Format

dataframe

See Also

MFG

`quickPlot` *Generic plotting showing results of microPop*

Description

Generic plotting showing results of microPop

Usage

```
quickPlot(
  soln,
  numR,
  numStrains,
  microbeNames,
  yLabel,
  xLabel,
  sumOverStrains,
  resourceLegendPosition = "topleft",
  microbeLegendPosition = "topleft",
  saveFig = FALSE,
  figType = "eps",
  figName = "microPopFig"
)
```

Arguments

`soln`	ODE output from microPopModel() i.e. matrix out$solution
`numR`	Scalar. Number of resources
`numStrains`	Scalar. Number of strains per group
`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`yLabel`	String for y axis label
`xLabel`	String for x axis label
`sumOverStrains`	Logical
`resourceLegendPosition`	String. Position of legend in resource plot, default is 'topleft'
`microbeLegendPosition`	String. Position of legend in microbe plot, default is 'topleft'
`saveFig`	Logical. Default is FALSE
`figType`	String. Default is "eps"
`figName`	String. Default is "microPopFig"

Value

Nothing just generates a plot

`quickPlot1` *Generic plotting showing results of microPop*

Description

Generic plotting showing results of microPop

Usage

```
quickPlot1(
  soln,
  numR,
  numStrains,
  microbeNames,
  yLabel,
  xLabel,
  sumOverStrains,
  resourceLegendPosition = "topleft",
  microbeLegendPosition = "topleft",
  saveFig = FALSE,
  figType = "eps",
  figName = "microPopFig"
)
```

Arguments

`soln`	ODE output from microPopModel() i.e. matrix out$solution
`numR`	Scalar. Number of resources
`numStrains`	Scalar. Number of strains per group
`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`yLabel`	String for y axis label
`xLabel`	String for x axis label
`sumOverStrains`	Logical
`resourceLegendPosition`	String. Position of legend in resource plot, default is 'topleft'
`microbeLegendPosition`	String. Position of legend in microbe plot, default is 'topleft'
`saveFig`	Logical. Default is FALSE
`figType`	String. Default is "eps"
`figName`	String. Default is "microPopFig"

Value

Nothing just generates a plot

`quickPlot2` *Generic plotting showing results of microPop Now shows resources and microbes on one plot.*

Description

Generic plotting showing results of microPop Now shows resources and microbes on one plot.

Usage

```
quickPlot2(
  soln,
  numR,
  numStrains,
  microbeNames,
  yLabel = "Concentration (g/L)",
  xLabel = "Time",
  sumOverStrains = TRUE,
  resourceLegendPosition = "topleft",
  microbeLegendPosition = "topleft",
  saveFig = FALSE,
  figType = "eps",
  figName = "microPopFig",
  cex.plot = 1,
  cex.legend = 0.7
)
```

Arguments

`soln`	ODE output from microPopModel() i.e. matrix out$solution
`numR`	Scalar. Number of resources
`numStrains`	Scalar. Number of strains per group
`microbeNames`	Vector of strings which contains the names of the microbial groups in the system e.g. c('Bacteroides','Acetogens')
`yLabel`	String for y axis label. Default is "Concentration (g/L)"
`xLabel`	String for x axis label. Default is "Time"
`sumOverStrains`	Logical. Default=TRUE
`resourceLegendPosition`	String. Position of legend in resource plot, default is 'topleft'
`microbeLegendPosition`	String. Position of legend in microbe plot, default is 'topleft'
`saveFig`	Logical. Default is FALSE
`figType`	String. Default is "eps"
`figName`	String. Default is "microPopFig"
`cex.plot`	Multiplier for text size on axes text. Default is 1
`cex.legend`	Multiplier for text size in legend. Default is 0.7

Value

Nothing just generates a plot

`rateFuncsDefault` *List of functions that are used by the ODE solver these functions can be changed by the user but all must be listed.*

Description

rateFuncsDefault=list(pHFunc=pHFuncDefault, pHLimFunc=pHLimFuncDefault, extraGrowthLimFunc=extraGrowthLimFuncDefault, growthLimFunc=growthLimFuncDefault, combineGrowthLimFunc=combineGrowthLimFuncDefault, uptakeFunc=uptakeFuncDefault, productionFunc=productionFuncDefault, combinePathsFunc=combinePathsFuncDefault, massBalanceFunc=massBalanceFuncDefault, entryRateFunc=entryRateFuncDefault, removalRateFunc=removalRateFuncDefault)

Usage

```
rateFuncsDefault
```

Format

An object of class `list` of length 11.

`removalRateFuncDefault` *Removal Rate Function*

Description

Return the rate of removal of any state variable from the system This is called in the ODE derivs func

Usage

```
removalRateFuncDefault(varName, varValue, stateVarValues, time, washOut, parms)
```

Arguments

`varName`	(string) Name of state variable of interest (this is group name or a resource name - NOT a strain name)
`varValue`	(scalar) value of state variable of interest
`stateVarValues`	(named vector) values of all state variables
`time`	(scalar) time
`washOut`	(named vector) of wash out rates (per unit time) of groups and resources (specified in SysInfo files)
`parms`	List containing all system parameters

Value

(scalar) rate of removal (quantity per unit time) for the state variable varName

---

`replaceListItems` *used to replace items in list.in in list.default needed for processing microPop input args like plotOptions*

Description

used to replace items in list.in in list.default needed for processing microPop input args like plotOptions

Usage

```
replaceListItems(list.in, list.default)
```

Arguments

`list.in`	input List
`list.default`	Default List

Value

list.default updated with entries from list.in

`reshapeFlowMat` *reshapeFlowMat*

Description

reshapes the flow matrices out$flow.uptake or out$flow.production into a list elements of the list are the microbeNames and then there is a matrix [path,res]

Usage

```
reshapeFlowMat(time.step, flow.direction, out)
```

Arguments

`time.step`	is the index of the chosen time
`flow.direction`	is either 'uptake' or 'production'
`out`	is the output from microPopModel with networkAnalysis=TRUE

Value

a list with microbeNames as elements and a matrix of [path,resource] showing the chosen flow direction (eg. uptake or production). Note theses flows have not been converted to moles.

`resourceSysInfo` *resourceSysInfo*

Description

Data frame describing the system information for the state variables that are resources (i.e. substrates or metabolic products).

Usage

```
resourceSysInfo
```

Format

A dataframe with the row names in the itemised list below and a column for units (optional) and for each resource in the system to be simulated.

Details

Each resource (substrate, metabolic product or biomass if microbes are a resource e.g. in the case of viruses) has a column. The first column can be used for describing the units of each variable. This is optional and just for clarity - it is not used within microPop (note, the units column must be labelled 'units'). The data frame must contain the following rows:

- startValue The value of each resource at the start time of the simulation (e.g. units are g/l)
- inflowRate The value of the rate of inflow of each resource (e.g. units are g/l/d)
- washOut The specific washout rate of each resource (e.g. units are /d)
- molarMass The mass in grams of one mole of the resource (units are g/mol)

`resourceSysInfoHuman` *resourceSysInfoHuman dataframe*

Description

Table of information describing the inflows, outflows, start values and molar masses of each resource for the R script microPop/inst/DemoFiles/human*.R See help(resourceSysInfo) or for an explanation of the contents

Usage

```
resourceSysInfoHuman
```

Format

dataframe

See Also

resourceSysInfo

`resourceSysInfoRumen` *resourceSysInfoRumen dataframe*

Description

Table of information describing the inflows, outflows, start values and molar masses of each resource for the R script microPop/inst/DemoFiles/rumen*.R See help(resourceSysInfo) or for an explanation of the contents

Usage

```
resourceSysInfoRumen
```

Format

dataframe

See Also

resourceSysInfo

`runMicroPopExample` *runMicroPopExample*

Description

This function is similar to the demo() function but requires less interaction It is used to run the canned examples from the microPop package.

Usage

```
runMicroPopExample(name = NULL)
```

Arguments

`name`	Name of the example to run. If Name is NULL the list of examples will be printed.

`strainParams` *strainParams dataframe*

Description

Table containing some parameter values for specific strains for the R script microPop/inst/DemoFiles/human4.R The file must have colnames c(strainName, paramName, paramVal, paramUnit, resource,path) where strainName is in format 'groupName.i' where i is the strain number.

Usage

```
strainParams
```

Format

dataframe

`sumConcOverStrains` *sumConcOverStrains*

Description

sum concentration of each strain into the group it is in

Usage

```
sumConcOverStrains(
  concentration.orig,
  allStrainNames,
  groupNames,
  resourceNames
)
```

Arguments

`concentration.orig`	the row of out$solution at the required time point
`allStrainNames`	is a vector containing the names of the microbial strains (strings)
`groupNames`	is a vector containing the names of the microbial groups (strings)
`resourceNames`	is a vector of strings containing the names of all the resources

`sumFlowOverStrains` *sumFlowOverStrains*

Description

make links and nodes matrices for use in network plotting software

Usage

```
sumFlowOverStrains(flowList, allStrainNames, groupNames)
```

Arguments

`flowList`	is list containing the production or uptake flows (the output from reshapeFlow-Mat())
`allStrainNames`	is a vector containing the names of the microbial strains (strings)
`groupNames`	is a vector containing the names of the microbial groups (strings)

`sumFlowsOverPaths` *sumFlowsOverPaths*

Description

sum flows over links between the same nodes i.e. if the link has more than one metabolic path

Usage

```
sumFlowsOverPaths(links)
```

Arguments

`links`	data frame or matrix of links

Value

matrix of links

`systemInfoMicrobesPhyto` *systemInfoMicrobesPhyto dataframe*

Description

Table of information describing the inflows, outflows, start values of each microbial group for the R script microPop/inst/DemoFiles/phyto.R See help(microbeSysInfo) or for an explanation of the contents

Usage

```
systemInfoMicrobesPhyto
```

Format

dataframe

See Also

microbeSysInfo

`systemInfoMicrobesVirus` *systemInfoMicrobesVirus dataframe*

Description

Table of information describing the inflows, outflows, start values of each microbial group for the R script microPop/inst/DemoFiles/phages.R See help(microbeSysInfo) or for an explanation of the contents

Usage

```
systemInfoMicrobesVirus
```

Format

dataframe

See Also

microbeSysInfo

`systemInfoResourcesPhyto`

systemInfoResourcesPhyto dataframe

Description

Table of information describing the inflows, outflows, start values and molar masses of each resource for the R script microPop/inst/DemoFiles/phyto.R See help(resourceSysInfo) or for an explanation of the contents

Usage

```
systemInfoResourcesPhyto
```

Format

dataframe

See Also

resourceSysInfo

`systemInfoResourcesVirus`

systemInfoResourcesVirus dataframe

Description

Table of information describing the inflows, outflows, start values and molar masses of each resource for the R script microPop/inst/DemoFiles/phages.R See help(resourceSysInfo) or for an explanation of the contents

Usage

```
systemInfoResourcesVirus
```

Format

dataframe

See Also

resourceSysInfo

uptakeFuncDefault *Uptake Function*

Description

Return the value of resource uptake per biomass (i.e. resource quantity per unit time per mass unit of biomass) for given resource

Usage

```
uptakeFuncDefault(
  strainName,
  groupName,
  pathName,
  varName,
  keyResName,
  subst,
  ess,
  boost,
  maxGrowthRate,
  growthLim,
  yield,
  nonBoostFrac,
  stoichiom,
  parms
)
```

Arguments

strainName	Name of the strain that is being looped through in the ODE solver
groupName	Name of microbial group that is being looped through in the ODE solver
pathName	Name of metabolic path (e.g. path1) that is being looped through in the ODE solver
varName	(string). Calculate uptake of this variable
keyResName	(string). Name of the key resource on this pathway
subst	Vector of strings giving the names of the substitutable resources for given strain, pathway
ess	Vector of strings giving the names of the essential resources for given strain, pathway
boost	Vector of strings giving the names of the boosting resources for given strain, pathway
maxGrowthRate	Vector containing maximum growth rate on each resource (named by resourceNames). If a resource is not on the pathway the value is NA
growthLim	Vector containing the growth limitation from each resource (named by resourceNames). If a resource is not on the pathway the value is NA

`yield`	Named vector (names are resourceNames) giving the mass yield of biomass on each resource (mass microbe/mass resource)
`nonBoostFrac`	(scalar) Fraction of max growth achievable if boosting resource is not present but is required on this pathway
`stoichiom`	Named vector (names are resourceNames) giving the mass of each resource in the stoichiometry i.e. molar mass of resource multiplied by the number of moles in the stoichiometry
`parms`	List containing all system parameters

Value

(scalar) uptake of resource per mass unit of biomass (units are resource mass/biomass/time)

`Xaa` *Xaa dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
Xaa
```

Format

dataframe

See Also

MFG

`Xh2` *Xh2 dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
Xh2
```

Format

dataframe

See Also

MFG

`Xsu` *Xsu dataframe*

Description

Table of information describing the behaviour of the microbial functional group. See help(MFG) or ?MFG for explanation of the contents of the microbial functional groups dataframes

Usage

```
Xsu
```

Format

dataframe

See Also

MFG

Index