Package 'sybilDynFBA'

June 6, 2020

Type Package

Title Dynamic FBA : Dynamic Flux Balance Analysis

Version 1.0.2

Date 2020-6-6

Maintainer Abdelmoneim Amer Desouki <abdelmoneim.amer@uni-duesseldorf.de>

Author Abdelmoneim Amer Desouki [aut, cre]

Depends R (>= 2.12.0), sybil (>= 1.2.0)

Imports methods

Description Implements dynamic FBA technique proposed by Varma et al 1994.

LazyLoad yes

License GPL-3

NeedsCompilation no

Repository CRAN

Date/Publication 2020-06-06 11:50:02 UTC

R topics documented:

`sybilDynFBA-package` *Dynamic Flux Balance Analysis*

Description

The package `sybilDynFBA` implements dynamic flux balance analysis as proposed by Varma et al (1994). It uses functions from package `sybil` to find standard FBA solution. Solution can also be plotted.

Details

Package:	sybilDynFBA
Type:	Package
Version:	1.0.0
Date:	2015-07-24
License:	GPL Version 3
LazyLoad:	yes
Depends:	sybil

Author(s)

Abdelmoneim Amer Desouki

Maintainer: Abdelmoneim Amer Desouki <abdelmoneim.amer@uni-duesseldorf.de>

References

Varma, A. and Palsson, B.O. 1994. Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110. Appl Environ Microbiol 60: 3724-3731.

See Also

sybil

Examples

```
## Not run:
## The examples here require the package glpkAPI to be
## installed. If that package is not available, you have to set
## the argument 'solver' (the default is: solver = "glpk").

## load the example data set
data(Ec_core)
    mod <- Ec_core
```

```
# Change bounds for glucsoe, oxygen and acetate uptake
mod <- changeBounds(mod, react = "EX_glc(e)", lb = -12)
mod <- changeBounds(mod, react = "EX_o2(e)",  lb = -10)
mod <- changeBounds(mod, react = "EX_ac(e)",  lb = -10)

# initial values
init.source <- c("EX_ac(e)","EX_o2(e)","EX_glc(e)")
init.conc   <- c(10,50,28)
init.bmass  <- 0.01

# dFBA
Ec_df <- dynamicFBA(mod,exclUptakeRxns = c(),
                    substrateRxns      = init.source,
                    initConcentrations = init.conc,
                    initBiomass        = init.bmass,
                    timeStep=.1,nSteps=200,verbose=3)


# Plotting
plot(Ec_df,
  plotRxns=c('EX_glc(e)','EX_ac(e)',"EX_for(e)","EX_o2(e)"),
                  legend_cex=0.85,legend_xpos=0,legend_ypos=45)


## End(Not run)
```

---

dynamicFBA *dynamic flux balance analysis*

Description

Calculate concentrations of metabolites of exchange reactions at defined time points given the initial concentrations. To accomplish this task this function calls optimizeProb function to get the fluxes then update the concentrations and the reaction boundaries ..etc.

Usage

```
dynamicFBA(model, substrateRxns, initConcentrations, initBiomass, timeStep, nSteps,
exclUptakeRxns,
retOptSol = TRUE,
 fld = FALSE, verboseMode = 2, ...)
```

Arguments

model	An object of class modelorg.
substrateRxns	List of exchange reaction names for substrates initially in the media that may change (e.g. not h2o or co2)

`initConcentrations`
: The given start concentrations of substrates

`initBiomass`
: The start value of biomass (must be nonzero)

`timeStep`
: Define the points of time to evaluate the problem at.

`nSteps`
: The maximum number of steps, the procedure may stop before completing this number when the substrate run out.

`exclUptakeRxns`
: List of uptake reactions whose substrate concentrations do not change (Default ='EX_co2(e)','EX_o2(e)','EX_h2o(e)','EX_h(e)')

`retOptSol`
: Boolean. indicates if optsol calss will be returned or simple list.
Default: `TRUE`

`fld`
: Boolean. Save the resulting flux distribution.
Default: `FALSE`

`verboseMode`
: An integer value indicating the amount of output to stdout: 0: nothing, 1: status messages, 2: like 1 plus a progress indicator, 3: a table containing the reaction id's and the corresponding min max values.
Default: `2`.

`...`
: Further arguments passed to `sysBiolAlg`. Argument `solverParm` is a good candidate.

Value

returns `optsol_dynamicFBA`

Author(s)

Abdelmoneim Amer Desouki

References

Varma, A. and Palsson, B.O. 1994. Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110. Appl Environ Microbiol 60: 3724-3731.

Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox. *Nat Protoc* **2**, 727–738.

See Also

`modelorg`, `optsol_dynamicFBA`, `optimizeProb`, `sysBiolAlg`, `SYBIL_SETTINGS`

Examples

```
## Not run:
## The examples here require the package glpkAPI to be
## installed. If that package is not available, you have to set
## the argument 'solver' (the default is: solver = "glpk").

## load the example data set
data(Ec_core)
```

```
        lowbnd(Ec_core)[react_id(Ec_core)=='EX_glc(e)']=-10;
        lowbnd(Ec_core)[react_id(Ec_core)=='EX_o2(e)']=-18;
## run dynamicFBA(), Ec_df will be an object of class \code{\link{optsol_dynamicFBA}}
Ec_df <- dynamicFBA(Ec_core,substrateRxns={'EX_glc(e)'},initConcentrations=10,
initBiomass=.035,timeStep=.25,nSteps=20,verbose=3)

## plot biomass and reactions
plot(Ec_df,plotRxns=c('EX_glc(e)','EX_ac(e)'));


## End(Not run)
```

Ec_core *Escherichia coli Core Energy Metabolism Network*

Description

The dataset is a network representation of the *E. coli* core metabolism. It consists of 62 internal reactions, 14 exchange reactions and a biomass objective function.

Usage

```
data(Ec_core)
```

Format

An object of class `modelorg`

Source

http://systemsbiology.ucsd.edu/Downloads/EcoliCore

References

Bernhard Ø. Palsson (2006). *Systems Biology: Properties of Reconstructed Networks*. Cambridge University Press.

optsol_dynamicFBA-class
Class "optsol_dynamicFBA"

Description

Structure of the class "optsol_dynamicFBA". Objects of that class are returned by the function dynamicFBA. Extends the Class optsol_optimizeProb.

Objects from the Class

Objects can be created by calls of the function optsol_dynamicFBA:

test <-optsol_dynamicFBA(solver = "glpk",method = "simplex").

Slots

solver: Object of class "character" indicating the used solver.

method: Object of class "character" indicating the used method.

num_of_prob: Object of class "integer" indicating the number of optimization problems.

lp_num_cols: Object of class "integer" indicating the number of columns.

lp_num_rows: Object of class "integer" indicating the number of rows.

lp_obj: Object of class "numeric" containing the values of the objective function.

lp_ok: Object of class "integer" containing the exit code of the optimization.

lp_stat: Object of class "integer" containing the solution status of the optimization.

lp_dir: Object of class "character" indicating the direction of optimization.

concentrationMatrix Object of class "matrix" contains concentrations of extracellular metabolite

excRxnNames Object of class "matrix" contains names of exchange reactions for the EC metabolites

fluxdist: Object of class "fluxDistribution" containing the solutons flux distributions.

timeVec Object of class "numeric" Vector of time points

biomassVec Object of class "numeric" Vector of biomass values

all_fluxes Object of class "matrix" contains fluxes of all reactions at all steps

Extends

Class "optsol_optimizeProb", directly. Class "optsol", by class "optsol_optimizeProb", distance 2.

Methods

plot `signature(x = "optsol_dynamicFBA", y = "missing")`:

x An object of class `optsol_dynamicFBA`.

y not used but kept for compitability with generic plot.

plotRxns List of reaction id's to be ploted

... Further arguments passed to `sysBiolAlg`. Argument `solverParm` is a good candidate.

Author(s)

Abdelmoneim Amer Desouki

See Also

`checkOptSol`, `optsol`, `optsol_optimizeProb`

Examples

```
showClass("optsol_dynamicFBA")
```

Index