Package 'CoRegFlux'

October 16, 2019

Type Package

Title CoRegFlux

Version 1.0.0

Author Pauline Trébulle, Daniel Trejo-Banos, Mohamed Elati

Maintainer Pauline Trébulle and Mohamed Elati <coregflux@gmail.com>

Description CoRegFlux aims at providing tools to integrate reverse engineered gene regulatory networks and gene-expression into metabolic models to improve prediction of phenotypes, both for metabolic engineering, through transcription factor or gene (TF) knockout or overexpression in various conditions as well as to improve our understanding of the interactions and cell inner-working.

License GPL-3

Encoding UTF-8

LazyData false

SystemRequirements GLPK (>= 4.42)

Depends R (>= 3.6)

Imports CoRegNet, sybil

RoxygenNote 6.1.1

Suggests glpkAPI, testthat, knitr, rmarkdown, digest, R.cache, ggplot2, plyr, igraph, methods, latex2exp, rBayesianOptimization

biocViews GeneRegulation,Network,SystemsBiology,GeneExpression,Transcription,GenePrediction

VignetteBuilder knitr

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

git_branch RELEASE_3_9

git_last_commit d3bd471

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

R topics documented:

`adjust_constraints_to_observed_rates`
Adjust the constraint of the model to observed rates

Description

Adjust the constraint of the model to observed rates

Usage

```
adjust_constraints_to_observed_rates(model, metabolites_with_rates,
  exchange_met = build_exchange_met(model), backward_fluxes = "_b",
  forward_fluxes = "_f")
```

Arguments

`model`	a genome-scale metabolic model of class modelorg
`metabolites_with_rates`	is a data.frame consisting of the name of the metabolites, their concentrations and rates in mmol/gDW/h. The column name must be "name", "concentrations","rates"
`exchange_met`	Optional. a data.frame as given by build_exchange_met
`backward_fluxes`	Optional. Useful for irreversible model
`forward_fluxes`	Optional. Useful for irreversible model

Value

Return the model with updated bounds corresponding to the observed rates provided

Examples

```
data("iMM904")
metabolites_rates<-data.frame(name=c("D-Glucose","Glycerol"),
                          concentrations=c(16,0),rates=c(-2.81,-8.01))
iMM904_adjusted<-adjust_constraints_to_observed_rates(iMM904,
metabolites_rates)
```

---

aliases_SC *aliases_SC data*

Description

A data.frame containing the gene ID used in the metabolic model and their common name, used in the gene regulatory network

Usage

```
aliases_SC
```

Format

a two colums data.frame which first columns correspond to the name used in the model and the second to the ID used in the GRN (common name). Those columns should be named geneName_model and geneName_GRN respectively.

geneName_model Aliases or gene names used in the gene-association field in the genome-scale metabolic model

geneName_GRN Aliases or gene names used in the gene regulatory network

---

build_exchange_met *Build the exchange metabolite data.frame*

Description

Build the exchange metabolite data.frame

Usage

```
build_exchange_met(model)
```

Arguments

model	An object of class modelOrg, the genome scale metabolic model

Value

a data.frame containing the exchange metabolite model id and the equivalent name

Examples

```
data("iMM904")
exchanged_met<-build_exchange_met(iMM904)
head(exchanged_met)
```

`coregflux_static` *Update the model using the provided gene regulatory network and expression*

Description

`coregflux_static()` uses the gene states to update the fluxes bounds from the metabolic model.

Usage

```
coregflux_static(model, predicted_gene_expression, gene_parameter = 0,
  tol = 1e-10, aliases = NULL)
```

Arguments

`model`	A genome-scale metabolic model of class modelorg
`predicted_gene_expression`	The vector of predicted gene expression for the genes present in the metabolic model as given by `predict_linear_model_influence()`
`gene_parameter`	Parameter of the softplus function
`tol`	Fluxes values below this threshold will be ignored.
`aliases`	a data.frame containing the gene names currently used in the network under the colname "geneName" and the alias under the colnames "alias"

Value

list containing:

`model`	the metabolic model with the coregflux constraints added
`softplus_positive`	the results of evaluating ln(1+exp(gpr(x+theta))) where gpr() are the continuous version of the gpr rules applied to a set of gene expression x
`softplus_negative`	the results of evaluating ln(1+exp(gpr(x+theta))) where gpr() are the continuous version of the gpr rules applied to a set of gene expression x

Examples

```
data("SC_GRN_1")
data("SC_experiment_influence")
data("SC_EXP_DATA")
data("aliases_SC")
data(iMM904)
data(PredictedGeneState)
static_list<-coregflux_static(iMM904,PredictedGeneState)
```

get_biomass_flux_position *Get biomass flux position*

Description

Get biomass flux position

Usage

```
get_biomass_flux_position(model, biomass_reaction_id = "biomass",
  biomass_reaction_name = NULL)
```

Arguments

model	An object of class modelOrg, the genome scale metabolic model
biomass_reaction_id	Default value "biomass"
biomass_reaction_name	Optional, the react_name in the modelOrg under which the biomass function can be found, such as "growth"

Value

the position of the biomass generating reaction according the the objective in our case we had the biomass reactions for models iMM904 and iTO977

Examples

```
data("iMM904")
get_biomass_flux_position(iMM904)
```

get_fba_fluxes_from_observations *Get fluxes balance from an observed growth rate*

Description

Get fluxes balance from an observed growth rate

Usage

```
get_fba_fluxes_from_observations(model, observed_growth_rate,
  metabolites_rates = NULL,
  biomass_flux_index = get_biomass_flux_position(model),
  backward_fluxes = "_b", forward_fluxes = "_f")
```

Arguments

model	a genome-scale metabolic model of class modelorg
observed_growth_rate	a numerical value for the observed growth rate
metabolites_rates	Optional, a data.frame consisting of the name of the metabolites, their concentrations and rates in mmol/gDW/h to adjust the model uptake rates. The column name must be "name", "concentrations","rates"
biomass_flux_index	Optional. Index of the biomass flux as returned by `get_biomass_flux_position()`
backward_fluxes	Optional, only relevant for irreversible model
forward_fluxes	Optional, only relevant for irreversible model

Value

Return fluxes values compatible with the observed growth rate through flux balance analysis

Examples

```
data("iMM904")
metabolites_rates<-data.frame(name=c("D-Glucose","Glycerol"),
                         concentrations=c(16,0),rates=c(-2.81,-8.01))

FluxesFromObs<-get_fba_fluxes_from_observations(iMM904,0.205,
metabolites_rates = metabolites_rates)
```

---

get_fva_intervals_from_observations

Get intervals of flux variability (FVA) from an observed growth rate

---

Description

Get intervals of flux variability (FVA) from an observed growth rate

Usage

```
get_fva_intervals_from_observations(model, observed_growth_rate,
  metabolites_rates = NULL,
  biomass_flux_index = get_biomass_flux_position(model))
```

Arguments

model	a genome-scale metabolic model of class modelorg
observed_growth_rate	a numerical value for the observed growth rate
metabolites_rates	Optional. a data.frame consisting of the name of the metabolites, their concentrations and rates in mmol/gDW/h to adjust the model uptake rates. The column name must be "name", "concentrations","rates"
biomass_flux_index	Optional. Index of the biomass flux as returned by `get_biomass_flux_position()`

Value

Return the interval of fluxes values compatible with the observed growth rate through flux variability analysis

Examples

```
data("iMM904")
metabolites_rates<-data.frame(name=c("D-Glucose","Glycerol"),
                         concentrations=c(16,0),rates=c(-2.81,-8.01))

FluxesVarFromObs<-get_fva_intervals_from_observations(iMM904,0.205,
metabolites_rates=metabolites_rates)
```

---

get_linear_model *Train a linear model*

Description

Here we train a linear regression model of the form x= alpha + beta*I where x is the gene expression of the metabolic genes of the train data set train_expression, alpha is an intercept, I is the influence of the regulators of the training data set and beta are the coefficients.

Usage

```
get_linear_model(train_expression,
  train_influence = regulatorInfluence(network, train_expression, minTarg
  = 10), network)
```

Arguments

train_expression	Gene expression of the training data set, not necessary if train_influence is supplied. Should be numerical matrix corresponding to the gene expression. Rownames should contain gene names/ids while samples should be in columns.
train_influence	Optional. Regulator influence scores computed using the function CoRegNet::regulatorInfluence for the training data set, default minTarg = 10
network	CoRegNet object use to build the linear model and to compute the influence.

Details

train_expression Gene expression of the training data set, not necessary if train_influence is supplied. Should be numerical matrix corresponding to the gene expression. Rownames should contain gene names/ids while samples should be in columns.

Value

A linear model

See Also

predict_linear_model_influence

get_metabolites_exchange_fluxes *Get metabolites exchange fluxes*

Description

Get metabolites exchange fluxes

Usage

```
get_metabolites_exchange_fluxes(model, metabolites,
  exchange_met = build_exchange_met(model), backward_fluxes = "_b",
  forward_fluxes = "_f")
```

Arguments

model	An object of class modelOrg, the genome scale metabolic model
metabolites	A data.frame containing the names and concentrations of metabolites
exchange_met	A data.frame as build by the function build_exchange_met
backward_fluxes	Optional parameter for irreversible model to indicate backward fluxes
forward_fluxes	Optional parameter for irreversible model to indicate forward fluxes

Value

a data.frame containing the exchange metabolite model id and the equivalent name

Examples

```
data("iMM904")
metabolites<-data.frame("name"=c("D-Glucose","Glycerol"),
                        "concentrations"=c(16,0))
get_metabolites_exchange_fluxes(iMM904,metabolites)
```

iMM904 *iMM904*

Description

A _S. cerevisiae_ genome-scale metabolic model as a modelOrg object

Usage

```
iMM904
```

Format

a modelOrg object as required by _sybil_. See _sybilSBML_ for more information on how to load other model.

ODCurveToFluxCurves *ODCurveToFluxCurves*

Description

This function takes measured ODs and turn them into a FluxCurves object to be visualize using visFluxCurves(). It relies on flux variability analysis to highlight the flux value interval required to meet the specified OD.

Usage

```
ODCurveToFluxCurves(model, ODs, times, metabolites_rates = NULL,
  biomass_flux_index = get_biomass_flux_position(model))
```

Arguments

model	An object of class modelOrg, the metabolic model.
ODs	A vector of measured ODs
times	A vector of timepoints at which the flux balance analysis solution will be evaluated.
metabolites_rates	A data.frame containing the extraneous metabolites, their initial concentrations and their uptake rates. Columns must be named "names","concentrations" and "rates".
biomass_flux_index	Optional. index of the flux corresponding to the biomass reaction.

Value

An object FluxCurves to visualize using the function visFluxCurves

See Also

visFluxCurves, ODCurveToMetabolicGeneCurves, visMetabolicGeneCurves

Examples

```
data(iMM904)
ODs<-seq.int(0.099,1.8,length.out = 5)
times = seq(0.5,2,by=0.5)

metabolites_rates <- data.frame("name"=c("D-Glucose"),
"concentrations"=c(16.6),"rates"=c(-2.81))

ODtoflux<-ODCurveToFluxCurves(model = iMM904,
ODs = ODs,times = times, metabolites_rates = metabolites_rates)

visFluxCurves(ODtoflux)
```

ODCurveToMetabolicGeneCurves

ODCurveToMetabolicGeneCurves

Description

This function takes measured ODs and turn them into a ODcurveToMetCurve object to be visualize using `visMetabolicGeneCurves()`. It relies on flux variability analysis to highlight the flux value interval required to meet the specified OD and to map it on the metabolic genes.

Usage

```
ODCurveToMetabolicGeneCurves(times, ODs, metabolites_rates = NULL, model,
  softplusParam = 0, singlePointFluxEstimate = FALSE,
  biomass_flux_index = get_biomass_flux_position(model),
  aliases = NULL)
```

Arguments

`times`	A vector of timepoints at which the flux balance analysis solution will be evaluated.
`ODs`	vector of measured ODs.
`metabolites_rates`	A data.frame containing the extraneous metabolites, their initial concentrations and their uptake rates. Columns must be named "names","concentrations" and "rates".
`model`	An object of class modelOrg, the metabolic model.
`softplusParam`	Softplus parameter identify through calibration.
`singlePointFluxEstimate`	Optional, logical.
`biomass_flux_index`	index of the flux corresponding to the biomass reaction.
`aliases`	Optional. A data.frame containing the gene names used in the metabolic model and the aliases to use to match the regulatory network.

Value

Metabolic genes curves to visualize using the function `visMetabolicGeneCurves`

Examples

```
ODs<-c(0.4500000,0.5322392,0.6295079,0.7445529)
data("aliases_SC","iMM904")
ODcurveToMetCurve<-ODCurveToMetabolicGeneCurves(times = seq(0.5,2,by=0.5),
ODs = ODs,model = iMM904,aliases = aliases_SC)
visMetabolicGeneCurves(ODcurveToMetCurve,genes="YJR077C")
```

ODcurveToMetCurve *ODcurveToMetCurve data*

Description

List as obtained by the function ODCurveToMetabolicGeneCurves

Usage

```
ODcurveToMetCurve
```

Format

List as obtained by the function ODCurveToMetabolicGeneCurves

ODtoflux *ODtoflux data*

Description

List as obtained by the function ODCurveToFluxCurves

Usage

```
ODtoflux
```

Format

List as obtained by the function ODCurveToFluxCurves

ODToFluxBounds *ODToFluxBounds*

Description

ODToFluxBounds

Usage

```
ODToFluxBounds(odRate, model, metabolites_rates = NULL,
  biomass_flux_index = get_biomass_flux_position(model))
```

Arguments

odRate	The values of OD measured over time
model	An object of class modelOrg, the metabolic model.
metabolites_rates	A data.frame containing the extraneous metabolites, their initial concentrations and their uptake rates. Columns must be named "names","concentrations" and "rates".
biomass_flux_index	Optional. index of the flux corresponding to the biomass reaction.

Value

Flux bounds from OD

PredictedGeneState *PredictedGeneState data*

Description

Predicted gene states as obtained by the function `predict_linear_model_influence`

Usage

```
PredictedGeneState
```

Format

a named vector containing the gene name and its associated predicted gene state.

---

predict_linear_model_influence *Predict the gene expression level based on condition-specific influence*

Description

Build a linear model and use it to predict the gene expression level from the influence of an experiment

Usage

```
predict_linear_model_influence(network, model,
  train_influence = regulatorInfluence(network, train_expression,
  min_Target), experiment_influence, train_expression, min_Target = 10,
  tol = 1e-10, aliases = NULL, verbose = 0)
```

Arguments

network	a coregnet object
model	A genome-scale metabolic model from a class modelOrg.
train_influence	Optional, if is train_expression is provided. An influence matrix as computed by the function regulatorInfluence() from CoRegNet
experiment_influence	Regulator influence scores for the condition of interest as a named vector with the TF as names.
train_expression	Gene expression of the training data set, not necessary if train_influence is supplied. Should be numerical matrix corresponding to the gene expression. Rownames should contain gene names/ids while samples should be in columns.
min_Target	Optional. Use in case train_influence is not provided. Default value = 10. See regulatorInfluence for more information.
tol	Fluxes values below this threshold will be ignored. Default
aliases	Optional, A two columns data.frame containing the name used in the gene regulatory network and their equivalent in the genome-scale metabolic model to allow the mapping of the GRN onto the GEM. The colnames should be geneName_model and geneName_GRN
verbose	Default to 0. Give informations about the process status

Value

The predicted genes expressions/states

Examples

```
data("SC_GRN_1")
data("SC_experiment_influence")
data("SC_EXP_DATA")
data("iMM904")
data("aliases_SC")
PredictedGeneState <- predict_linear_model_influence(network = SC_GRN_1,
                    experiment_influence = SC_experiment_influence,
                    train_expression = SC_EXP_DATA,
                    min_Target = 4,
                    model = iMM904,
                    aliases= aliases_SC)

GeneState<-data.frame("Name"=names(PredictedGeneState),
                    "State"=unname(PredictedGeneState))
```

SC_experiment_influence

SC_experiment_influence data

Description

A vector of influence computed from the first sample of SC_Test_data

Usage

```
SC_experiment_influence
```

Format

a named numerical vector

SC_EXP_DATA *SC_EXP_DATA data*

Description

A matrix of S. cerevisiae gene expression in various experimental designs, derived from the m3d dataset to infer _S. cerevisiae_ gene regulatory network. The dataset was shorten to 3600 genes in order to limit the size of the object

Usage

```
SC_EXP_DATA
```

Format

a matrix of 3600 genes by 247 samples

Source

subset of m3d dataset available at <http://m3d.mssm.edu/>

SC_GRN_1 *SC_GRN_1 data*

Description

A coregnet object infered from the m3d dataset describing the gene regulatory network for S. cerevisiae as described in Banos, D. T., Trébulle, P., & Elati, M. (2017). Integrating transcriptional activity in genome-scale models of metabolism. BMC systems biology, 11(7), 134.

Usage

```
SC_GRN_1
```

Format

a coregnet object inferred using the package _CoRegNet_

Number of transcription factor 200

Number of targets genes 3748

Evidences TRUE ...

SC_Test_data *SC_Test_data data*

Description

A matrix of S. cerevisiae gene expression during diauxic shift (Brauer and al.)

Usage

```
SC_Test_data
```

Format

a matrix of 6028 genes by 13 samples during diauxic shift

Source

E-GEOD-4398 (Brauer MJ and al.)

Simulation *Simulation using Dynamic Flux balance analysis over time as in varma*

Description

Simulation using Dynamic Flux balance analysis over time as in *varma*

Usage

```
Simulation(model, time = c(0, 1), metabolites, initial_biomass,
  biomass_flux_index = CoRegFlux::get_biomass_flux_position(model),
  coregnet = NULL, regulator_table = NULL, gene_table = NULL,
  gene_state_function = NULL, time_step_fba_bounds = NULL,
  softplus_parameter = 0, aliases = NULL)
```

Arguments

model	An object of class modelOrg, the genome-scale metabolic model (GEM).
time	Timepoints at which the flux balance analysis solution will be evaluated.
metabolites	A data.frame containing the extraneous metabolites and the initial concentrations
initial_biomass	The value of the biomass at the beginning of the simulation
biomass_flux_index	index of the flux corresponding to the biomass reaction.
coregnet	Object of class CoRegNet, containing the regulatory and coregulatory interactions.

`regulator_table`
: A data.frame containing 3 columns: "regulator", "influence","expression" containing respectively the name of a TF present in the CoRegNet object as a string, its influence in the condition of interest as a numerical and an expression factor of 0 for a KO, or an integer >1 for an overexpression

`gene_table`
: A data.frame containing 2 columns: "gene" and "expression" containing respectively the name of a gene present in the modelOrg as a string and an expression factor of 0 for a KO, or an integer >1 for an overexpression

`gene_state_function`
: Function to obtain the gene state for a given subset of gene

`time_step_fba_bounds`
: Bounds for the fba problem at each time point, overrides any other form of constraining for a given flux.

`softplus_parameter`
: the softplus parameter identify through calibration

`aliases`
: Optional. A data.frame containing the gene names currently used in the network under the colname "geneName" and the alias under the colnames "alias"

Details

The simulation function allows the user to run several kind of simulations based on the provided arguments. When providing only the GEM, time, initial biomass and the metabolites, a classical dFBA is carried out. To integrate the gene expression to the GEM, the gene_state_function must be provided while if the user wants to simulate a TF knock-out or overexpression, then a coregnet object and the regulator table should also be provided. See the vignette and quick-user guide for more examples.

Value

Return a list containing the simulation information such as the objective_history, fluxes_history, met_concentration_history, biomass_history

Examples

```
data("SC_GRN_1")
data("SC_EXP_DATA")
data("SC_experiment_influence")
data("iMM904")
data("aliases_SC")
data("PredictedGeneState")

metabolites<-data.frame("name"=c("D-Glucose","Glycerol"),
                        "concentrations"=c(16,0))

result_without_any_constraint<-Simulation(iMM904,time=seq(1,10,by=1),
                   metabolites,
                   initial_biomass=0.45,
                   aliases = aliases_SC)

GeneState<-data.frame("Name"=names(PredictedGeneState),
                    "State"=unname(PredictedGeneState))

result<-Simulation(iMM904,time=seq(1,10,by=1),
                   metabolites,
```

```
                    initial_biomass=0.45,
                    gene_state_function=function(a,b){GeneState},
                    aliases = aliases_SC)

result$biomass_history
```

`Simulation_Step` *Single simulation step*

Description

Single simulation step in which fluxes are reconstrained according to metabolite concentrations, then given the continuous evaluation of the gpr rules and the softplus function of the gene regulatory state.

Usage

```
Simulation_Step(model, coregnet, metabolites, met_concentrations_t0,
  biomass_t0, regulator_table, gene_table, time_step, gene_state,
  softplus_parameter, aliases, biomass_flux_index)
```

Arguments

`model`	An object of class modelOrg, the metabolic model.
`coregnet`	Optional, object of class CoRegNet object containing the information about regulatory and coregulatory relationships
`metabolites`	data frame of metabolites names
`met_concentrations_t0`	data frame of metabolites concentrations at t0, before performing a time step
`biomass_t0`	biomass at t0 before performing a step
`regulator_table`	A data.frame containing 3 columns: "regulator", "influence","expression" containing respectively the name of a TF present in the CoRegNet object as string, its influence in the condition of interest as a numerical and an expression factor of 0 for a KO, or an integer >1 for an overexpression
`gene_table`	A data.frame containing 2 columns: "gene", "expression" containing respectively the name of a gene present in the CoRegNet object as string and an expression factor of 0 for a KO, or an integer >1 for an overexpression
`time_step`	size of the time step to perform; that is t1-t0
`gene_state`	data frame with rows being gene names and columns being the gene expression or any other continuous value representing metabolites activity to be evaluated using the gpr rules
`softplus_parameter`	Softplus parameter identify through calibration. Default to 0.
`aliases`	Optional. A data.frame containing the gene names currently used in the network under the colname "geneName" and the alias under the colnames "alias"
`biomass_flux_index`	index of the flux corresponding to the biomass reaction.

Value

list of: fluxes: fluxes of the resulting fba soulution to the metabolic and genetic constraints biomass_yield: biomass yield that is used as proxy for the growth rate in the dynamic flux balance analysis solution. Corresponds to the flux of the biomass reaction of the model.

See Also

Simulation

`update_fluxes_constraints_geneKOOV`
Update the fluxes constraints to simulate gene KO or overexpression

---

Description

Update the constraints of the reactions associated with the knock-out or overexpressed gene

Usage

```
update_fluxes_constraints_geneKOOV(model, gene_table, aliases = NULL)
```

Arguments

`model`	An object of class modelOrg, the metabolic model.
`gene_table`	A data.frame containing 2 columns: "gene", "expression" containing respectively the name of a gene present in the CoRegNet object as string and an expression factor of 0 for a KO, or an integer >1 for an overexpression
`aliases`	Optional. A data.frame containing the gene names used in the metabolic model and the aliases to use to match the regulatory network

Value

Return the model with updated bounds

Examples

```
data("iMM904")
data("aliases_SC")
gene_table <- data.frame("gene" = c("YGL202W","YIL162W"),
"expression" =c(2,0), stringsAsFactors = FALSE)

model_gene_KO_OV_constraints <- update_fluxes_constraints_geneKOOV(
model= iMM904,
gene_table =  gene_table,
aliases = aliases_SC)
```

update_fluxes_constraints_influence *Update the fluxes constraints to simulate TF KO or overexpression*

Description

Update the constraints according to the influence & regulatory network for a single KO or overexpression

Usage

```
update_fluxes_constraints_influence(model, coregnet, regulator_table,
  aliases)
```

Arguments

model	An object of class modelOrg, the metabolic model.
coregnet	Object of class CoRegNet, containing the regulatory and coregulatory interactions.
regulator_table	A data.frame containing 3 columns: "regulator", "influence","expression" containing respectively the name of a TF present in the CoRegNet object as string, its influence in the condition of interest as a numerical and an expression factor of 0 for a KO, or an integer >1 for an overexpression
aliases	Optional, a data.frame containing the gene names used in the metabolic model and the aliases to use to match the regulatory network

Value

Return the model with updated bounds

Examples

```
data("SC_GRN_1")
data("iMM904")
data("aliases_SC")
regulator_table <- data.frame("regulator" = "MET32",
                              "influence" = -1.20322 ,
                              "expression" = 3,
                              stringsAsFactors = FALSE)
model_TF_KO_OV_constraints <- update_fluxes_constraints_influence(
model= iMM904,coregnet = SC_GRN_1,regulator_table =  regulator_table,
aliases = aliases_SC)
```

`update_uptake_fluxes_constraints_metabolites` *Update the fluxes constraints given the metabolite concentrations*

Description

Update the fluxes constraints given the metabolite concentrations

Usage

```
update_uptake_fluxes_constraints_metabolites(model, met_fluxes_indexes,
  met_concentrations_t0, biomass_t0, time_step)
```

Arguments

`model`	An object of class modelOrg, the metabolic model.
`met_fluxes_indexes`	Indexes of the metabolites fluxes
`met_concentrations_t0`	Metabolites concentrations at t0
`biomass_t0`	Biomasss at t0
`time_step`	time_step studied

Value

Return the updated model

`visFluxCurves` *Visualize Fluxes Curves*

Description

Visualize Fluxes Curves

Usage

```
visFluxCurves(fluxCurves, genes = unique(fluxCurves$name)[seq_len(50)],
  ...)
```

Arguments

`fluxCurves`	result table from ODCurveToFluxCurves
`genes`	a vector containing the names of the metabolic genes to plot. Default select the first 50 genes
`...`	Optional others curves

Value

a plot of the curves of the chosen fluxes

See Also

ODCurveToFluxCurves, ODCurveToMetabolicGeneCurves, visMetabolicGeneCurves

Examples

```
data("ODtoflux")
visFluxCurves(ODtoflux,genes ="ADK3")
```

`visMetabolicGeneCurves` *Visualize Metabolic Gene Curves*

Description

Visualize Metabolic Gene Curves

Usage

```
visMetabolicGeneCurves(metabCurves,
  genes = unique(metabCurves$name)[seq_len(50)], ...)
```

Arguments

`metabCurves`	result table from ODCurveToMetabolicGeneCurves
`genes`	a vector containing the names of the metabolic genes to plot. Default select the first 50 genes
`...`	Optional, others curves

Value

a plot of the curves of the chosen metabolic genes

See Also

ODCurveToMetabolicGeneCurves,ODCurveToFluxCurves, visFluxCurves

Examples

```
data("ODcurveToMetCurve")

visMetabolicGeneCurves(ODcurveToMetCurve,genes="YJR077C")
```

Index