Package 'fbar'

January 14, 2017

Type Package

Title An Extensible Approach to Flux Balance Analysis

Version 0.1.23

Date 2017-1-14

Description This is a simple package for Flux Balance Analysis and related metabolic modelling techniques. Functions are provided for: parsing models in tabular format, converting parsed metabolic models to input formats for common linear programming solvers, and evaluating and applying gene-protein-reaction mappings. In addition, there are wrappers to parse a model, select a solver, find the metabolic fluxes, and return the results applied to the original model. Compared to other packages in this field, this package puts a much heavier focus on providing reusable components that can be used in the design of new implementation of new techniques, in particular those that involve large parameter sweeps.

License GPL-3

Depends R (>= 3.0.0)

Imports assertthat, Matrix, dplyr, tidyr, purrr, tibble, stringr, magrittr, ROI

Suggests knitr, rmarkdown, testthat, tidyverse, Rglpk, ROI.plugin.ecos, ROI.plugin.glpk

VignetteBuilder knitr

RoxygenNote 5.0.1

NeedsCompilation no

Author Max Conway [aut, cre]

Maintainer Max Conway <conway.max1@gmail.com>

Repository CRAN

Date/Publication 2017-01-14 12:46:35

R topics documented:

`ecoli_core` *A small ecoli model, created from a number of sources.*

Description

A small ecoli model, created from a number of sources.

Usage

```
ecoli_core
```

Format

A data frame with 95 rows and 7 columns:

abbreviation an abbreviated reaction name, acts as the reaction id

officialName full reaction name

equation reaction equation

lowbnd lower bound on the reaction rate

uppbnd upper bound on the reaction rate

obj_coef identifies a reaction (or reactions) for which the maximum possible rate should be found

geneAssociation A boolean combination of genes which control the reaction

Source

`http://gcrg.ucsd.edu/Downloads/EcoliCore`, Reconstruction and Use of Microbial Metabolic Networks: the Core Escherichia coli Metabolic Model as an Educational Guide, A comprehensive genome-scale reconstruction of Escherichia coli metabolism–2011.

expanded_to_glpk *Parse a long format metabolic model to a glpk model*

Description

This parses the long format produced by `reactiontbl_to_expanded` to a glpk model.

Usage

```
expanded_to_glpk(reactions_expanded)
```

Arguments

`reactions_expanded`	A list of data frames as output by `reactiontbl_to_expanded`

Details

To install the Rglpk package in linux, run `sudo apt-get install libglpk-dev` in a terminal, and then run `install.packages('Rglpk')` in R.

The `reaction_table` must have columns:

- `abbreviation`,
- `equation`,
- `uppbnd`,
- `lowbnd`, and
- `obj_coef`.

Value

A list suitable for input to Rglpk

See Also

Other parsing_and_conversion: `expanded_to_ROI`, `expanded_to_gurobi`, `reactiontbl_to_expanded`, `reactiontbl_to_gurobi`

Examples

```
data(ecoli_core)
library(dplyr)

glpk_model <- ecoli_core %>%
  reactiontbl_to_expanded %>%
  expanded_to_glpk

if(requireNamespace('Rglpk', quietly=TRUE)){
```

```
  glpk_result <- purrr::lift_dl(Rglpk::Rglpk_solve_LP)(glpk_model)

  ecoli_core_with_flux <- ecoli_core %>%
    mutate(flux = glpk_result[['solution']])
}
```

---

expanded_to_gurobi *Parse a long format metabolic model to a gurobi model*

Description

Used as the second half of reactiontbl_to_gurobi, this parses the long format produced by reactiontbl_to_expanded to a gurobi model

Usage

```
expanded_to_gurobi(reactions_expanded)
```

Arguments

reactions_expanded	A list of data frames as output by expand_reactions

Details

For installation instructions for Gurobi, refer to the Gurobi website: http://www.gurobi.com/.

The reaction_table must have columns:

- abbreviation,
- equation,
- uppbnd,
- lowbnd, and
- obj_coef.

Value

A list suitable for input to Gurobi.

See Also

Other parsing_and_conversion: expanded_to_ROI, expanded_to_glpk, reactiontbl_to_expanded, reactiontbl_to_gurobi

Examples

```
data(ecoli_core)
library(dplyr)

gurobi_model <- ecoli_core %>%
  reactiontbl_to_expanded %>%
  expanded_to_gurobi

## Not run:
if(requireNamespace('gurobi', quietly=TRUE)){
  gurobi <- gurobi(gurobi_model)

  ecoli_core_with_flux <- ecoli_core %>%
    mutate(flux = guorbi_result[['solution']])
}

## End(Not run)
```

`expanded_to_reactiontbl` *Convert intermediate expanded format back to a reaction table*

Description

Useful for saving a new or edited model

Usage

```
expanded_to_reactiontbl(expanded)
```

Arguments

`expanded`	A list of data frames: • `rxns`, which has one row per reaction, • `mets`, which has one row for each metabolite, and • `stoich`, which has one row for each time a metabolite appears in a reaction.

Value

A data frame describing the metabolic model.

expanded_to_ROI *Parse a long format metabolic model to an ROI model*

Description

This parses the long format produced by `reactiontbl_to_expanded` to an ROI model.

Usage

```
expanded_to_ROI(reactions_expanded)
```

Arguments

`reactions_expanded`	A list of data frames as output by `reactiontbl_to_expanded`

Details

To solve models using ROI, you will need a solver plugin for ROI. Probably the easiest one to install is ROI.plugin.glpk. To install this in linux, run `sudo apt-get install libglpk-dev` in a terminal, and then run `install.packages('ROI.plugin.glpk')` in R.

The `reaction_table` must have columns:

- `abbreviation`,
- `equation`,
- `uppbnd`,
- `lowbnd`, and
- `obj_coef`.

Value

A list suitable for input to ROI.

See Also

Other parsing_and_conversion: `expanded_to_glpk`, `expanded_to_gurobi`, `reactiontbl_to_expanded`, `reactiontbl_to_gurobi`

Examples

```
## Not run:
data(ecoli_core)
library(dplyr)
try(library(ROI.plugin.ecos)) # make a solver available to ROI

roi_model <- ecoli_core %>%
```

```
  reactiontbl_to_expanded %>%
  expanded_to_ROI

if(length(ROI::ROI_applicable_solvers(roi_model))>=1){
  roi_result <- ROI::ROI_solve(roi_model)

  ecoli_core_with_flux <- ecoli_core %>%
    mutate(flux = roi_result[['solution']])
}

## End(Not run)
```

`fbar` *fbar: Flux Balance Analysis in R with a tidy data approach*

Description

fbar is a simple, easy to use Flux Balance Analysis package with a tidy data approach. Just `data_frames` and the occasional `list`, no new classes to learn. The focus is on simplicity and speed. Models are expected as a flat table, and results can be simply appended to the table. This makes this package very suitable for useage in pipelines with pre- and post- processing of models and results, so that it works well as a backbone for customized methods. Loading, parsing and evaluating a model takes around 0.1s, which, together with the straightforward data structures used, makes this library very suitable for large parameter sweeps.

Details

For a list of functions in the package, see `vignette('Introduction','fbar')`

`find_fluxes_df` *Given a metabolic model as a data frame, return a new data frame with fluxes*

Description

Given a metabolic model as a data frame, return a new data frame with fluxes

Usage

```
find_fluxes_df(reaction_table, do_minimization = TRUE)
```

Arguments

`reaction_table`	a data frame representing the metabolic model
`do_minimization`	toggle to uniformly minimize all non-objective fluxes after finding the objective

Details

This function uses ROI, so to solve models, you will need a solver plugin for ROI. Probably the easiest one to install is ROI.plugin.glpk. To install this in linux, run `sudo apt-get install libglpk-dev` in a terminal, and then run `install.packages('ROI.plugin.glpk')` in R.

Value

The input data frame with a new numeric column, "`flux`".

See Also

find_fluxes_vector

Examples

```
## Not run:
data(ecoli_core)
ecoli_core_with_flux <- find_fluxes_df(ecoli_core)

## End(Not run)
```

`find_flux_variability_df` *Given a metabolic model as a data frame, return a new data frame with fluxes and variability*

Description

This function calculates fluxes `folds` times with shuffled versions of the metabolic model. This is designed to detect and quantify underdetermined fluxes.

Usage

```
find_flux_variability_df(reaction_table, folds = 10, do_minimization = TRUE)
```

Arguments

`reaction_table`	a data frame representing the metabolic model
`folds`	number of times to calculate fluxes
`do_minimization`	toggle to uniformly minimize all non-objective fluxes after finding the objective

Details

This function uses ROI, so to solve models, you will need a solver plugin for ROI. Probably the easiest one to install is ROI.plugin.glpk. To install this in linux, run `sudo apt-get install libglpk-dev` in a terminal, and then run `install.packages('ROI.plugin.glpk')` in R.

Value

reaction_table with two added columns: sd (the standard deviation of fluxes found) and flux (a typical flux) from this distribution

---

`gene_associate` *Apply gene expressions to reaction table*

Description

A convenience function that uses `gene_eval` and a custom function to apply new upper and lower bounds.

Usage

```
gene_associate(reaction_table, gene_table,
  expression_flux_function = function(x) {     (1 +
  log(x)/stats::sd(x)^2)^sign(x - 1) })
```

Arguments

`reaction_table`	A data frame describing the metabolic model.
`gene_table`	A data frame showing gene presence
`expression_flux_function`	a function to convert from gene set expression to flux

Value

the reaction_table, with a new column, present, and altered upper and lower bounds

Warning

This function relies on `gene_eval`, which uses `eval` to evaluate gene expression sets. This gives flexibility, but means that malicious code in the `gene_sets` argument could get evaluated. `gene_sets` is evaluated in a restricted environment, but there might be a way around this, so you might want to check for anything suspicious in this argument manually. For more information, read the code.

See Also

gene_eval

gene_eval *Function to estimate the expression levels of gene sets*

Description

Function to estimate the expression levels of gene sets

Usage

```
gene_eval(gene_sets, genes, presences)
```

Arguments

gene_sets	A list of gene set strings: names of genes punctuated with &, | and brackets.
genes	A list of gene names
presences	A list of gene presences, the same length as genes

Value

a vector the same length as gene_sets, with the the calcuated combined gene expression levels.

This function evaluates the gene sets in the context of the gene presences. It can take booleans, or numbers, in which case it associates & with finding the minimum, and | with finding the maximum.

Warning

This function uses eval to evaluate gene expression sets. This gives flexibility, but means that malicious code in the gene_sets argument could get evaluated. gene_sets is evaluated in a restricted environment, but there might be a way around this, so you might want to check for anything suspicious in this argument manually. For more information, read the code.

See Also

gene_associate

iJO1366 *A full size ecoli model.*

Description

A full size ecoli model.

Usage

```
iJO1366
```

Format

A data frame with 95 rows and 8 columns:

abbreviation an abbreviated reaction name, acts as the reaction id

name full reaction name

equation reaction equation

subsystem an indicator of reaction function

GPR A boolean combination of proteins and genes which control the reaction

lowbnd lower bound on the reaction rate

uppbnd upper bound on the reaction rate

obj_coef identifies a reaction (or reactions) for which the maximum possible rate should be found

Source

A comprehensive genome-scale reconstruction of Escherichia coli metabolism–2011.

`parse_met_list` *Internal function: Expand half reaction equations into a long form*

Description

Internal function: Expand half reaction equations into a long form

Usage

```
parse_met_list(mets)
```

Arguments

`mets`	Character vector of halves of reaction equations.

Value

a `date_frame` with columns:

stoich the stoichiometric coefficient

met the metabolite

`reactiontbl_to_expanded` *Parse a reaction table to an intermediate, long format*

Description

The long format can also be suitable for manipulating equations.

Usage

```
reactiontbl_to_expanded(reaction_table, regex_arrow = "<?[-=]+>")
```

Arguments

`reaction_table`	A data frame describing the metabolic model.
`regex_arrow`	Regular expression for the arrow splitting sides of the reaction equation.

Details

The `reaction_table` must have columns:

- `abbreviation`,
- `equation`,
- `uppbnd`,
- `lowbnd`, and
- `obj_coef`.

Value

A list of data frames:

- `rxns`, which has one row per reaction,
- `mets`, which has one row for each metabolite, and
- `stoich`, which has one row for each time a metabolite appears in a reaction.

See Also

Other parsing_and_conversion: `expanded_to_ROI`, `expanded_to_glpk`, `expanded_to_gurobi`, `reactiontbl_to_gurobi`

Examples

```
## Not run:
data(ecoli_core)
library(dplyr)
try(library(ROI.plugin.ecos)) # make a solver available to ROI

roi_model <- ecoli_core %>%
  reactiontbl_to_expanded %>%
  expanded_to_ROI

if(length(ROI::ROI_applicable_solvers(roi_model))>=1){
  roi_result <- ROI::ROI_solve(roi_model)

  ecoli_core_with_flux <- ecoli_core %>%
    mutate(flux = roi_result[['solution']])
}

## End(Not run)
```

`reactiontbl_to_gurobi` *Parse reaction table to Gurobi format*

Description

Parses a reaction table to give a list in Gurobi's input format. This function is a shorthand for `reactiontbl_to_expanded` followed by `expanded_to_gurobi`.

Usage

```
reactiontbl_to_gurobi(reaction_table, regex_arrow = "<?[-=]+>")
```

Arguments

`reaction_table`	A data frame describing the metabolic model.
`regex_arrow`	Regular expression for the arrow splitting sides of the reaction equation.

Details

The `reaction_table` must have columns:

- `abbreviation`,
- `equation`,
- `uppbnd`,
- `lowbnd`, and
- `obj_coef`.

Value

A list suitable for input to Gurobi.

See Also

Other parsing_and_conversion: expanded_to_ROI, expanded_to_glpk, expanded_to_gurobi, reactiontbl_to_expanded

Examples

```
data(ecoli_core)
library(dplyr)

gurobi_model <- ecoli_core %>%
  reactiontbl_to_expanded %>%
  expanded_to_gurobi

## Not run:
if(requireNamespace('gurobi', quietly=TRUE)){
  gurobi <- gurobi(gurobi_model)

  ecoli_core_with_flux <- ecoli_core %>%
    mutate(flux = guorbi_result[['solution']])
}

## End(Not run)
```

---

split_on_arrow *Internal function: Splitting reaction equation into substrate and product*

Description

Internal function: Splitting reaction equation into substrate and product

Usage

```
split_on_arrow(equations, regex_arrow = "<?[-=]+>")
```

Arguments

equations	Character vector of reaction equations.
regex_arrow	Regular expression for the arrow splitting sides of the reaction equation.

Value

a `data_frame`, with columns:

reversible boolean, is reaction reversible

before the left hand side of the reaction string

after the right hand side of the reaction string

Index