TN-CORE: INTERPRETATION OF TN-SEQ DATA USING METABOLIC NETWORK RECONSTRUCTION AND MODELLING

George C diCenzo, Alessio Mengoni, Marco Fondi

TABLE OF CONTENTS

OVERVIEW

Tn-Core was developed with the intent of assisting in the interpretation of Tn-seq data using metabolic network reconstruction and constraint-based metabolic modelling. Tn-Core provides an user-friendly workflow for the generation of context-specific core metabolic models through the integration Tn-seq data, and optionally also RNA-seq data. Tn-Core is further designed to assist in the analysis of redundancy in core metabolic networks, and to assist in the curation of genome-scale metabolic network reconstructions. The output returned by Tn-Core should be beneficial to experimental biologists wishing to use metabolic modelling to better understand their Tn-seq data, as well as to computational biologists wishing to integrate Tn-seq data with their metabolic modelling pipelines and/or to refine genome-scale metabolic reconstructions.

The current version of Tn-Core consists of 21 functions, one of which is an overall analysis workflow. These functions are written in MATLAB, and make use of COBRA formatted models, the COBRA Toolbox, and the TIGER Toolbox. The most recent version of Tn-Core is available in the following GitHub repository: https://github.com/diCenzo-GC/Tn-Core.

A more detailed description of the functionality of Tn-Core can be found in the associated publication that describes its development, available here:
diCenzo GC, Mengoni A, Fondi M (2018) Tn-Core: a toolbox for integrating Tn-seq gene essentiality data and constraint-based metabolic modelling. *ACS Synthetic Biology*. **8:**158-169.

CONTACT INFORMATION

If you have question related to the usage of Tn-Core, please contact either George diCenzo at georgecolin.dicenzo@unifi.it and/or Marco Fondi at marco.fondi@unifi.it.

CITATION

If you make use of any of the scripts of Tn-Core, any of its output files, or the webserver, please cite the dependencies listed on the next and the following article describing the Tn-Core Toolbox:

diCenzo GC, Mengoni A, Fondi M (2018) Tn-Core: a toolbox for integrating Tn-seq gene essentiality data and constraint-based metabolic modelling. *ACS Synthetic Biology*. **8:**158-169.

DEPENDANCIES

Tn-Core is dependent upon the following three tools:

COBRA Toolbox
(opencobra.github.io):
Schellenberger, J, Que R, Fleming RM, Thiele I, Orth JD, Feist AM, Zielinski DC, Bordar A, Lewis NE, Rahmanian S, Kang J, Hyduke DR, Palsson BØ (2011) Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox v2.0. *Nat Protoc.* **6:**1290-1307.

TIGER Toolbox
(bme.virginia.edu/csbl/Downloads1-tiger.html):
Jensen PA, Lutz KA, Papin JA (2011) TIGER: Toolbox for integrating genome-scale metabolic models, expression data, and transcriptional regulatory networks. *BMC Syst Biol*. **5:**147.

FASTCORE
(en.uni.lu/recherche/fstc/life_sciences_research_unit/research_areas/systems_biology/software/fastcore):
Vlassis N, Pacheco MP, Sauter T (2014) Fast reconstruction of compact context-specific metabolic network models. *PLOS Comput Biol*. **5:**e1003424.

VERSION HISTORY

Version 1.0: The initial release of Tn-Core consisting of the following 8 functions: i) tncore_main, ii) tncore_randomize, iii) tncore_matrix, iv) tncore_refine, v) tncore_compare, vi) tncore_delete, vii) tncore_remove, viii) tncore_reconstruct.

Version 1.1: Several new functions and modifications have been added.

i) The previous tncore_main function has been renamed as tncore_redundancy.
ii) A new script for context-specific core model generation called tncore_core has been added, as has the associated script tncore_gimme.
iii) New functions for initializing the workspace (tncore_init), importing the necessary data (tncore_import), and exporting the output (tncore_export) have been added.
iv) An overall workflow function (tncore_workflow) for performing the complete analysis pipeline has been added.
v) The tncore_refine function was updated. Previously this function updated the grRules field in the genome-scale model, but did not properly update the rules field or the rxnGeneMat field. This has been fixed. Additionally, it has been updated to work with the new tncore_core function.
vi) The tncore_redundancy function now contains options to define the essentiality threshold and the essentiality category bins.
vii) The tncore_randomize function now contains an option to define the essentiality category bins.
viii) The tncore_randomize function has been updated to ensure that metabolites are correctly removed from the model during removal of dead-end producing reactions.
ix) The tncore_delete function was modified in order to ensure all grRules are correctly updated.

Version 1.2: One function has been modified.

i) Several modifications were made to the tncore_refine function. This function now exports a table summarizing the changes made during the refinement. The script was modified to ensure that it correctly identifies putative 'or' to 'and' replacements in the GPRs. It was also modified so that putative 'or' to 'and' replacements are no longer made during the refinement, but they are instead only listed in the output table as putative changes that could be made.

Version 2.0: Contains seven new functions added since the previous version. Additionally, several updates to the existing functions have been made.

i) New function: the function tncore_duplicates for removal of duplicate genes from a model.
ii) New function: the function tncore_fix to add grRules, rxnGeneMat, or rules fields if missing in a model.
iii) New function: the function tncore_expand to enlarge a core (or any other) metabolic model based on reactions present in a second genome-scale metabolic model, including GPR relationships.
iv) New function: the function tncore_rna for generation of a context-specific model based on RNA-seq data, optionally using essentiality data or a pre-defined core gene (an adaptation

of tncore_core that makes a core model based on Tn-seq data, optionally using RNA-seq data).

v) New function: the function tncore_multi_gimme for preparation of context-specific models from a multi-species starting model, using RNA-seq data.

vi) New function: the function tncore_deadends to iteratively remove reactions containing deadend metabolites, resulting in a model containing no deadends.

vii) New function: the function tncore_remove_reactions to remove a list of reactions from the model, ensuring that there are no issues if the number of reactions and metabolites are equal.

viii) Updated function: the tncore_core function was modified to limit the likelihood of it failing to finish.

ix) Updated function: The tncore_core function was updated to remove the 'essThresh' input variable as it was not used in the script. Functions calling tncore_core have also been updated accordingly.

x) Updated functions: the tncore_core function was updated to fix an error in the removal of lowly expressed protein complexes.

xi) Updated function: the tncore_delete function was updated to avoid inadvertently allowing a reaction to be active without all essential genes (i.e., with half of a protein complex), and two other errors were fixed. A bug resulting in the accidental deletion of most to all genes associated with a reaction has also been fixed. Further modified to ensure compatibility with newer versions of the COBRA Toolbox.

xii) Updated function: the tncore_redundancy function was updated to properly remove the unnecessary unknowns from the rxnGeneMat field of the input model.

xiii) Updated function: the tncore_remove function was updated to change the code used for updating the rxnGeneMat.

xiv) Updated functions: all functions previously using the removeRxns function have been updated to use the tncore_remove_reactions function.

xv) Updated functions: all functions using the detectDeadEnds function have been updated to set the removeExternalMets option to true.

Version 2.1: Contains fixes for two of the existing functions.

i) Updated function: an error in the code for the function tncore_randomize was fixed.

ii) Updated function: the function tncore_remove_reactions was updated to ensure it does not occasionally fail, and an error in the help text was fixed.

Version 2.2: Updated three of the existing functions.

i) Updated function: the code to identify genes no longer present in a model was updated in the function tncore_remove.

ii) Updated function: the function tncore_core was updated to ensure input parameters are properly parsed.

iii) Updated function: the function tncore_import was updated to allow the function to still work even if the ExRxns.txt file is empty.

WORKFLOW FUNCTIONS

TNCORE_OVERALL_WORKFLOW

Description

An overall pipeline for running the Tn-Core Toolbox, from initializing the workspace and importing the data, through to exporting of the output. Running this script will: i) import the necessary data from the current working directory; ii) if desired, produce a context-specific core model based on the Tn-seq data and optionally the RNA-seq data; iii) if desired, evaluate redundancy within core metabolic pathways; and iv) export the output files.

Usage

tncore_overall_workflow(varargin)

Basic set ups

Below are two of the most basic set-ups of this function, one for generating of a core model, and one for generating a core model and examining redundancy.

The following code will generate a core model based on the provided Tn-seq data (without considering RNA-seq data), without providing an analysis of redundancy:
tncore_overall_workflow()

The following code will generate a core model based on the provided Tn-seq data (without considering RNA-seq data), and will providing an analysis of redundancy:
tncore_overall_workflow('testRedun', true)

Optional inputs

solver: The solver to be used. If no value is given, the default is the Gurobi solver.

prepCore: A true/false variable indicating if a context-specific core metabolic model should be prepared. If no value is given, the default is to prepare a context-specific core metabolic model.

testRedun: A true/false variable indicating if an analysis of redundancy in core metabolic pathways should be performed. If no value is given, the default is to not examine redundancy.

tnseqData: A true/false variable indicating if Tn-seq data is available in the current working directory for core model generation. If no value is given, the default is to assume that Tn-seq data is available in the working directory, and to use it to during generation of a context-specific core metabolic model.

rnaseqData: A true/false variable indicating if RNA-seq data is available in the current working directory for core model generation. If no value is given, the default is to assume that no RNA-seq data is available, and to not use RNA-seq data during generation of a context-specific core metabolic model.

epsilon: The epsilon value (the minimum flux rate that is considered non-zero) to be used during the running of FASTCC and FASTCORE during the preparation of a context-specific core model. If not value is given, the default value is 1.1e-6.

binThresh: An array containing three values (number of standard deviations away from the mean of the log of the Tn-seq data) to use for setting the limits of each bin. This variable is used both in core model generation, and in analysis of redundancy. If no value is given, the default array is: {'3.5'; '2.5'; '1.5'} (i.e., the bin boundaries are 3.5, 2.5, and 1.5 standard deviations below the mean value).

essThresh: The threshold (in number of standard deviations from the mean of the log of the Tn-seq data) for a gene to be considered essential. This variable is used both in core model generation, and in analysis of redundancy. Genes considered to be essential are protected during core model generation and during analysis of redundancy. However, an essential gene may still be removed from the core model if its associated reaction that is constrained due to the deletion of another gene. If no value is given, the default is 3.5 (i.e., genes with an essentiality value more than 3.5 standard deviations below the mean value are considered essential).

growthFrac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. This variable is used both in core model generation, and in analysis of redundancy. If no value is given, the default is 0.5 (i.e., half the growth of the full input model.

expressThresh: The threshold for a gene to be considered expressed. This variable is used both in core model generation, and in analysis of redundancy. Genes above the expression threshold are protected during core model generation and during analysis of redundancy. If no value is given, the default is set to 0.02% of the sum of all expression values.

keepHigh: A true/false variable indicating if high expressed genes should be added back into the context-specific model, after the context specific model has otherwise been prepared. In other words, once all genes to be kept in the core model is determined, should highly expressed genes be added to this list. Use 1 for yes, and 0 for no. If no value is given, the default is to not add highly expressed genes to the core model gene list.

deadends: A true/false variable indicating if reactions producing dead-end metabolites should be removed from the core model. If no value is given, the default is to remove the reactions producing dead-end metabolites.

iters: The number of random core metabolic models to be produced during analysis of redundancy. If no value is provided, the default value is 1,000.

redunMethod: Indicates if the FBA or MOMA algorithms should be used when calculating growth rates of gene deletion mutants during analysis of redundancy. Use 1 for FBA and 2 for MOMA. If no value is given, the default is to use FBA.

tnseqData2: A true/false variable indicating if a Tn-seq data is available and should be used during analysis of redundancy. If no value is given, the default is to assume that Tn-seq data is available in the working directory, and to use it to during the analysis of redundancy.

rnaseqData: A true/false variable indicating if RNA-seq data should be considered during the analysis of redundancy. If no value is given, the default is to not use RNA-seq data during the analysis of redundancy.

Output files if preparing a core model

exportedModel.xlsx: The core metabolic model in an Excel file. One sheet contains information on the reactions, one sheet contains information on the genes, and one sheet contains information on the metabolites.

Output files if testing redundancy

geneCoocurrenceTable.txt: A table listing each pair of genes present in at least one randomized core model, indicating the number of randomized core models containing each gene, the number of randomized core models containing both genes, the expected number of randomized core models containing both genes based on their individual occurrence, and a Chi-squared statistic indicating the strength of the difference between the observed and expected co-occurrence.

geneCoocurrenceMatrix.txt: A gene co-occurrence matrix indicating the number of randomized core models that contained both genes.

geneCoocurrenceAdjustedMatrix.txt: A gene co-occurrence matrix containing the Chi-squared statistics indicating the strength of the difference between the observed and expected co-occurrence of the genes in the randomized core models.

geneCountTable.txt: A gene occurrence table listing each gene, and the number and the percent of randomized core models that contain the gene.

rxnCoocurrenceTable.txt: A table listing each pair of reactions present in at least one randomized core model, indicating the number of randomized core models containing each reactions, the number of randomized core models containing both reactions, the expected number of randomized core models containing both reactions based on their individual occurrence, and a Chi-squared statistic indicating the strength of the difference between the observed and expected co-occurrence.

rxnCoocurrenceMatrix.txt: A reactions co-occurrence matrix indicating the number of randomized core models that contained both reactions.

rxnCoocurrenceAdjustedMatrix.txt: A reactions co-occurrence matrix containing the Chi-squared statistics indicating the strength of the difference between the observed and expected co-occurrence of the reactions in the randomized core models.

rxnCountTable.txt: A reactions occurrence table listing each reactions, and the number and the percent of randomized core models that contain the reactions.

genePresenceMatrix.txt: A matrix indicating which genes are present in each of the randomized core models.

rxnPresenceMatrix.txt: A matrix indicating which reactions are present in each of the randomized core models.

Notes

Unlike all other functions included with Tn-Core, the *tncore_overall_workflow* function uses parameter parsing in the input. Defaults are provided for all parameters. If you wish to modify a parameter, you must include the parameter name followed by the desired value. The order that the parameters are listed is not important. For example, to indicate that RNA-seq data should be used in extracting a context-specific core metabolic model and to not use binning, the following command would be used:
tncore_overall_workflow('binning', false, 'rnaseqData', true)

The workflow can be used to make a core model without evaluating redundancy, to evaluate redundancy without making a core model, or to both make a core model and to evaluate redundancy.

If *tncore_overall_workflow* encounters an error while calling one of the other functions in the Tn-Core Toolbox, please refer to the user guide section related to that function for further information.

TNCORE_INIT

Description

An optional, simple script to initialize the workspace for running Tn-Core. Initializes the COBRA Toolbox and the TIGER Toolbox, using the Gurobi solver by default.

Usage

tncore_init(solver)

Optional inputs

solver: The solver to be used. If no value is given, the default is the Gurobi solver.

TNCORE_IMPORT

Description

An optional function to import the data to be used with Tn-Core, and to update the model to be in the correct format. Data to be imported must be present in the current working directory. Files to import should be named as indicated below. Only the files for model import are required; the others are optional.

Usage

[model, tnseq, rnaseq] = tncore_import()

Input files for model import

inputModel.xml: A SBML formatted metabolic model, saved as an XML file.

ExRxns.txt: A tab-delimited file indicating the exchange reactions to be on (first column), and the lower boundary (second column). Exchange reactions not listed will have a lower bound of 0, unless it is determined to be essential for flux through the objective function. The upper bounds are left at the defaults. This file can be empty, although this is generally not recommended.

ObjectiveRxn.txt: A tab-delimited file indicating the reaction to be set as the objective function (first column) and the objective value (second column).

Input files for importing Tn-seq data

TnSeqData.txt: A tab-delimited file with the gene essentiality index in the first column (e.g., insertions per nucleotide), and the gene name in the second column.

Input files for importing RNA-seq data

RnaSeqData.txt: A tab-delimited file with the gene expression data (e.g., RPKM) in the first column, and the gene name in the second column.

Outputs

model: A COBRA formatted metabolic model with the objective function set, the exchange bounds set, and with the necessary fields to work with Tn-Core.

tnseq: The Tn-seq data formatted for use with Tn-Core.

rnaseq: The RNA-seq data formatted for use with Tn-Core.

Notes

This function can be used to import the model, and optionally the Tn-seq data and/or the RNA-seq data. All files must be located in the working directory and named as indicated above. If the imported metabolic model lacks a grRules field and/or a rxnGeneMat field, these fields will be built and added to the model. The list of exchange reactions will be used to set the boundary conditions; any exchange reaction not listed will have a lower bound of zero. If not flux through the objective function occurs under these conditions, a minimum set of additional exchange reactions necessary for flux through the objective function will be identified and turned on with a lower bound of -1000.

TNCORE_EXPORT

Description

A function to export the output of Tn-Core. Exports one or more of the following items:

1) The core metabolic model as an Excel file.
2) Several files related to the occurrence and co-occurrence of each gene, as text files. Includes only genes variably present in the core models
    a) A gene co-occurrence table. Lists each pair of genes, the number of randomized core models containing each gene, the number of randomized core models containing both genes, the expected number of randomized core models containing both genes based on their individual occurrence, and a Chi-squared statistic indicating the strength of the difference between the observed and expected co-occurrence.
    b) A gene co-occurrence matrix indicating the number of randomized core models that contained both genes.
    c) A gene co-occurrence matrix containing the Chi-squared statistics indicating the strength of the difference between the observed and expected co-occurrence of the genes in the randomized core models.
3) Several files related to the occurrence and co-occurrence of each reaction, as text files. Includes only reactions variably present in the core models.
    a) A reaction co-occurrence table. Lists each pair of reaction, the number of randomized core models containing each reaction, the number of randomized core models containing both reaction, the expected number of randomized core models containing both reaction based on their individual occurrence, and a Chi-squared statistic indicating the strength of the difference between the observed and expected co-occurrence.
    b) A reaction co-occurrence matrix indicating the number of randomized core models that contained both reaction.
    c) A reaction co-occurrence matrix containing the Chi-squared statistics indicating the strength of the difference between the observed and expected co-occurrence of the reaction in the randomized core models.
4) Two files related to the presence or absence of each gene in at least one core model.
    a) A gene presence matrix as a text file. A matrix indicating which genes are present in each of the randomized core models.
    b) A gene occurrence table listing each gene, and the number and the percent of randomized core models that contain the gene.
5) Two files related to the presence or absence of each gene in at least one core model.
    a) A reaction presence matrix as a text file. A matrix indicating which reactions are present in each of the randomized core models.
    b) A reaction occurrence table listing each reaction, and the number and the percent of randomized core models that contain the reaction.

Usage

tncore_export(model, reducedModel, tnseq, rnaseq, coMatGenes, coMatGenesAdj, … geneGrowMat, iters, coMatRxns, coMatRxnsAdj, rxnGrowMat, genePres, rxnPres)

Inputs for model export

model: The core metabolic model returned by the *tncore_main* function.
reducedModel: The reduced genome-scale metabolic model returned by the *tncore_main* function.

Optional inputs for model export

tnseq: The tnseq data used during core model preparation. If provided, the Tn-seq data will be listed next to each of the model genes in the 'Genes' sheet of the output model.

rnaseq: The RNA-seq data used during core model preparation. If provided, the RNA-seq data will be listed next to each of the model genes in the 'Genes' sheet of the output model.

Inputs for gene or reaction (co-)occurrence table and matrix export

iters: The number of iterations used in the *tncore_redundancy* function (i.e., the number of core models generated during the running of *tncore_redundancy*.

Inputs for gene (co-)occurrence table and matrix export

coMatGenes: The genes co-occurrence matrix produced by the *tncore_matrix* function.

coMatGenesAdj: The adjusted gene co-occurrence matrix (with the chi-squared statistics) produced by the *tncore_matrix* function.

geneGrowMat: The genes growth matrix produced by the *tncore_matrix* function.

Inputs for reaction (co-)occurrence table and matrix export

coMatRxns: The reaction co-occurrence matrix produced by the *tncore_matrix* function.

coMatRxnsAdj: The adjusted reaction co-occurrence matrix (with the chi-squared statistics) produced by the *tncore_matrix* function.

rxnGrowMat: The reaction growth matrix produced by the *tncore_matrix* function.

Inputs for gene presence table export

genePres: The gene presence variable returned by the *tncore_matrix* function.

Inputs for reaction presence table export

rxnPres: The reaction presence variable returned by the *tncore_matrix* function.

Output files if exporting a core model

exportedModel.xlsx: The core metabolic model in an Excel file. One sheet contains information on the reactions, one sheet contains information on the genes, and one sheet contains information on the metabolites.

Output files if exporting gene co-occurrence data

geneCoocurrenceTable.txt: A table listing each pair of genes present in at least one randomized core model, indicating the number of randomized core models containing each gene, the number of randomized core models containing both genes, the expected number of randomized core models containing both genes based on their individual occurrence, and a Chi-squared statistic indicating the strength of the difference between the observed and expected co-occurrence.

geneCoocurrenceMatrix.txt: A gene co-occurrence matrix indicating the number of randomized core models that contained both genes.

geneCoocurrenceAdjustedMatrix.txt: A gene co-occurrence matrix containing the Chi-squared statistics indicating the strength of the difference between the observed and expected co-occurrence of the genes in the randomized core models.

geneCountTable.txt: A gene occurrence table listing each gene, and the number and the percent of randomized core models that contain the gene.

Output files if exporting reaction co-occurrence data

rxnCoocurrenceTable.txt: A table listing each pair of reactions present in at least one randomized core model, indicating the number of randomized core models containing each reactions, the number of randomized core models containing both reactions, the expected number of randomized core models containing both reactions based on their individual occurrence, and a Chi-squared statistic indicating the strength of the difference between the observed and expected co-occurrence.

rxnCoocurrenceMatrix.txt: A reactions co-occurrence matrix indicating the number of randomized core models that contained both reactions.

rxnCoocurrenceAdjustedMatrix.txt: A reactions co-occurrence matrix containing the Chi-squared statistics indicating the strength of the difference between the observed and expected co-occurrence of the reactions in the randomized core models.

rxnCountTable.txt: A reactions occurrence table listing each reactions, and the number and the percent of randomized core models that contain the reactions.

Output files if exporting a gene presence matrix

genePresenceMatrix.txt: A matrix indicating which genes are present in each of the randomized core models.

Output files if exporting a reaction presence matrix

rxnPresenceMatrix.txt: A matrix indicating which reactions are present in each of the randomized core models.

Notes

This function can be used to export any combination of the five sets of files listed in the description section. The field *iters* must be provided for the export of gene (co-)occurrence data as well as for reaction (co-)occurrence; if exporting both gene and reaction (co-)occurrence data, only one value for *iters* can be provided (i.e., only gene and reaction data can be properly exported at once if the numbers of iters for both is the same).

CONTEXT-SPECIFIC MODEL GENERATION FUNCTIONS

TNCORE_CORE

Description

A function that uses the gene-centric GIMME implementation of the TIGER Toolbox to extract a context-specific core metabolic model from a genome-scale metabolic model based on the Tn-seq data, and optionally with RNA-seq data as well.

Usage

[contextModel, reducedModel] = tncore_core(model, tnseq, epsilon, binThresh, growthFrac, …
rnaseq, expressThresh, keepHigh, deadends)

Inputs

model: A COBRA formatted, genome-scale metabolic network reconstruction from which the core metabolic model is to be derived. The model must include gene-reaction associations. Exchange bounds must be set to represent the desired growth environment and nutrient uptake rate prior to the use of the model in this script.

tnseq: The Tn-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct essentiality thresholds to be determined. This variable should be a matrix, with the first column containing the Tn-seq data (not log transformed), and the second column containing the gene names.

Optional inputs

epsilon: The epsilon value (the minimum flux rate that is considered non-zero) to be used during the running of FASTCC and FASTCORE during the preparation of a context-specific core model. If not value is given, the default value is 1.1e-6.

binThresh: An array containing three values (number of standard deviations away from the mean of the log of the Tn-seq data) to use for setting the limits of each bin. If no value is given, the default array is: {'3.5'; '2.5'; '1.5'} (i.e., the bin boundaries are 3.5, 2.5, and 1.5 standard deviations below the mean value).

essThresh: The threshold (in number of standard deviations from the mean of the log of the Tn-seq data) for a gene to be considered essential. Genes considered to be essential are protected during core model generation. However, an essential gene may still be removed from the model if its associated reaction that is constrained due to the deletion of another gene. If no value is given, the default is 3.5 (i.e., genes with an essentiality value more than 3.5 standard deviations below the mean value are considered essential).

growthFrac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. If no value is given, the default is 0.5 (i.e., half the growth of the full input model.

rnaseq: The RNA-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct expression thresholds to be determined if not set by the user. This variable should be a matrix, with the first column containing the RNA-seq data (not log transformed), and the second column containing the gene names. Genes above the expression threshold are protected during core model generation; however, a highly expressed gene may still be removed from the model if its associated reaction is constrained due to the deletion of another gene. RNA-seq data can only be provided if Tn-seq data is also provided.

expressThresh: The threshold for a gene to be considered expressed. Genes above the expression threshold are protected during core model generation. If no value is given, the default is set to 0.02% of the sum of all expression values.

keepHigh: Indicates if high expressed genes should be added back into the context-specific model, after the context specific model has otherwise been prepared. In other words, once all genes to be kept in the core model is determined, should highly expressed genes be added to this list. Use 1 for yes, and 0 for no. If no value is given, the default is to not add highly expressed genes to the core model gene list.

deadends: Indicates if reactions producing dead-end metabolites should be removed from the core model. Use 1 for yes, and 0 for no. If no value is given, the default is to remove the reactions producing dead-end metabolites.

Outputs

contextModel: The context-specific core model based on the Tn-seq data (considering also the RNA-seq data, if provided).

reducedModel: The input model with dead-ends removed and with unnecessary 'Unknown' GPRs removed.

Notes

This pipeline optimizes model gene content based on the Tn-seq essentiality data. If both Tn-seq and RNA-seq data are provided, highly expressed genes are protected from deletion, and then the model is optimized based on the Tn-seq data.

Binning the data groups the Tn-seq data into essentiality classes, and sets the essentiality value of all genes within the same class to be equal. Four classes are prepared, with values of (from most essential to least essential): 1000, 10, 0.1, and 0. While this removes some of the quantitative data from the Tn-seq input, it also stabilizes the results (i.e., minor differences that may reflect technical fluctuations should have less impact). More importantly, it can alter the preference of the GIMME algorithm in cases of redundant pathways (as described further in the accompanying manuscript).

By default, reactions that produce dead-end metabolites (metabolites produced or consumed by only a single reaction) are removed from the core model as they will never carry flux in FBA. However, not removing dead-end reactions may provide additional information to users wishing to use this function to assist in interpretation of Tn-seq; such reactions are associated with genes essential in the Tn-seq study, but for which the entire pathway was not included in the core model. Exploring these reactions may therefore be of interest in some cases.

If RNA-seq data is provided, we recommend running this script at least twice: once with the option *keepHigh* set to 1, and once with the option *keepHigh* set to 0. These two set-ups are expected to often give differing core models, and the one that better represents the metabolism of the organism cannot be determined beforehand. We therefore recommend the script is run both ways, and the user to examine both outputs to determine which one is preferred.

TNCORE_RNA

Description

An adaptation of the *tncore_core* function for production of a context-specific model using RNA-seq data, and optionally also integrating a set of core genes determined via Tn-seq. Whereas *tncore_core* optionally uses expression data (RNA-seq) to first 'protect' genes, and then produces the context-specific core model based on essentiality data (Tn-seq), the *tncore_rna* function described here first 'protects' genes based on essentiality data (Tn-seq, or a user provided list of core genes), and the produces the context-specific model based on expression data (RNA-seq).

Usage

[contextModel, reducedModel] = tncore_rna(model, rnaseq, expressThresh, growthFrac, … coreGenes, tnseq, essThresh, forceHigh, epsilon)

Inputs

model: A COBRA formatted, genome-scale metabolic network reconstruction from which the context specific metabolic model is to be derived. The model must include gene-reaction associations. Exchange bounds must be set to represent the desired growth environment and nutrient uptake rate prior to the use of the model in this script.

rnaseq: The RNA-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct expression thresholds to be determined if not set by the user. This variable should be a matrix, with the first column containing the RNA-seq data (not log transformed), and the second column containing the gene names. Genes above the expression threshold are protected during core model generation; however, a highly expressed gene may still be removed from the model if its associated reaction is constrained due to the deletion of another gene.

Optional inputs

expressThresh: The threshold for a gene to be considered expressed. Genes above the expression threshold are protected during core model generation. If no value is given, the default is set to 0.02% of the sum of all expression values.

growthFrac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. If no value is given, the default is 0.5 (i.e., half the growth of the full input model.

coreGenes: A list of genes to be protected (i.e., not directly deleted) during generation of the core metabolic models. Note that a gene in this list may still be removed from the model if its associated reaction is constrained due to the deletion of another gene. This field can either be empty (no core genes are set prior to core model generation), contain a list of genes (genes to be

protected), or set to {1} (core genes are automatically determined by the script on the basis of the Tn-seq data as those with a log-transformed value < the median - 3.5 * the standard deviation). If set to {1}, Tn-seq data must also be provided.

tnseq: The Tn-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct essentiality thresholds to be determined. This variable should be a matrix, with the first column containing the Tn-seq data (not log transformed), and the second column containing the gene names. Must be provided if coreGenes is set to {1}.

essThresh: The threshold (in number of standard deviations from the mean of the log of the Tn-seq data) for a gene to be considered essential. Genes considered to be essential are protected during core model generation. However, an essential gene may still be removed from the model if its associated reaction that is constrained due to the deletion of another gene. If no value is given, the default is 3.5 (i.e., genes with an essentiality value more than 3.5 standard deviations below the mean value are considered essential).

forceHigh: An option to choose a workflow that favours a pathway/complex with a highly expressed gene over an alternative pathway/complex with no highly expressed gene but with a higher average expression. Use 1 to force inclusion of the highly expressed gene over alternate pathways, and 0 to not do this. If no input is given, the default is to not use this option.

epsilon: The epsilon value (the minimum flux rate that is considered non-zero) to be used during the running of FASTCC and FASTCORE during the preparation of a context-specific core model. Only relevant if *forceHigh* is set to 1. If not value is given, the default value is 1.1e-6.

Output

contextModel: The context-specific model based on the RNA-seq data (considering also the Tn-seq data, if provided).

reducedModel: The input model with dead-ends removed and with unnecessary 'Unknown' GPRs removed.

Notes

This pipeline optimizes model gene content based on the RNA-seq expression data. If both Tn-seq and RNA-seq data are provided, essential genes are protected from deletion, and then the model is optimized based on the RNA-seq data.

The *forceHigh* option is a workflow to favour the inclusion of a pathway/complex highly expressed gene over a pathway/complex with no highly expressed gene but an overall higher expression. To illustrate this, consider the following. Pathway One includes Gene A and Gene B, with expression levels of 500 and 100, respectively. Pathway Two includes Gene C and Gene D with expression levels of 300 and 300, respectively. Using an expression threshold of 400, the default algorithm is expected to favour Pathway Two, whereas the *forceHigh* workflow is expected to favour Pathway One.

TNCORE_GIMME

Description

A slight modification of the TIGER Toolbox (Jensen PA, Lutz KA, Papin JA. 2011. BMC Syst Biol. 5:147) implementation of GIMME (Becker SA, Palsson BØ. 2008. PLOS Comput Biol. 4:e1000082). After solving the MIP problem, instead of rounding the gene states, all gene states above zero are set to one, and the rest are set to zero. Requires the TIGER Toolbox to work.

Usage

[gene_states, genes, sol, tiger] = tncore_gimme(tiger, express, thresh, gene_names, obj_frac)

Inputs

tiger: The TIGER model from which the context-specific model is to be extracted.

express: Matrix containing expression data, or properly modified essentiality data, for all model genes. Should be provided in the same order as the genes in gene_names (or if gene_names is not provided, the same order as the genes in the tiger.genes field).

Optional inputs

thresh: The threshold for a gene to be considered expressed. Genes above the expression threshold are protected during core model generation. If no value is given, the default is set to 0.02% of the sum of all expression values. If using this function with essentiality data instead of expression data, this should be the threshold for a gene to be considered expressed, and should be manually set in the inputs.

gene_names: Cell array of the gene names in the same order as their expression/essentiality data in the express variable.

obj_frac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. If no value is given, the default is 0.5 (i.e., half the growth of the full input model.

Outputs

gene_states: Binary expression states calculated by GIMME.

genes: Cell array of the gene names corresponding to gene_states.

sol: The solution structure with details from the MILP solver.

tiger: TIGER model with gene_states applied.

TNCORE_MULTI_GIMME

Description

An adaptation of tncore_gimme (which is a slight modification of the TIGER Toolbox [Jensen PA, Lutz KA, Papin JA. 2011. BMC Syst Biol. 5:147] implementation of GIMME [Becker SA, Palsson BØ. 2008. PLOS Comput Biol. 4:e1000082]) to use with a multi-species model. Takes separate gene lists, expression data, and expression thresholds for each species in the model. Requires the TIGER Toolbox to work.

Usage

[gene_states, genes, sol, tiger, weightStruct] = tncore_multi_gumme(tiger, fields, … expressStruct, genesStruct, threshStruct, frac)

Inputs

tiger: The TIGER model from which the context-specific model is to be extracted.

fields: A cell array containing the names of each of the fields present in the expressStruct, geneStruct, and threshStruct fields.

expressStruct: A structure with one matrix per species included in the model. Each matrix should have expression data for all genes of that species present in the model, and should be provided in the same order as the corresponding gene list in the genesStruct structure.

genesStruct: A structure with one cell array per species included in the model. Each cell array should have the names of the genes of that species that are included in the model, and should be in the same order as the corresponding expression data in the expressStruct structure.

Optional inputs

threshStruct: A structure with one matrix per species included in the model. Each matrix should indicate the threshold for a gene of that species to be considered expressed. Genes above the expression threshold are protected during context-specific model generation. If no structure is given, the default is set to the mean of the expression values in the corresponding matrix of the expressStruct structure.

obj_frac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. If no value is given, the default is 0.5 (i.e., half the growth of the full input model.

Outputs

gene_states: Binary expression states calculated by GIMME.

genes: Cell array of the gene names corresponding to gene_states.

sol: The solution structure with details from the MILP solver.

tiger: TIGER model with gene_states applied.

weightStruct: A structure with one matrix per species included in the model. Each matrix indicates the weighting applied to the expression values of the corresponding species.

Notes

The main modification of this function compared to tncore_gimme is that a weighting is applied between species to normalize the expression data. The purpose of this is to ensure that the output model is not artificially biased towards one of the species in the model. Consider for example a model containing two interacting species, and the presence of a RNA-seq dataset generated for these interacting partners. For various reasons, we may expect that the mean expression level of genes from the two species may differ, for example, mean expression in species A may be 100 and mean expression in species B may be 200. If using a single threshold without normalizing the RNA-seq data, the output model is likely to be incorrectly biased towards species B. To account for this, this function applies a weighting based on the species-specific expression thresholds. So if mean expression was used as the expression threshold, the weighting for species A would be 1, and the weighting for species B would be 2. In essence, the expression values are then normalized by dividing the expression by the weighting (in practice, genes above the respective thresholds are first turned 'on', the difference between the 'off' genes and the respective thresholds are calculated, and then these differences are normalized by dividing by the weighting factor). The weightStruct returned by the function contains the weighting values applied to each species.

While a default threshold for highly expressed genes can be calculated by this function, this is not ideal as it will be based solely on the expression level of the genes in the model. It is therefore recommended to first calculate an expression threshold based on all genes in each of the species, and provide this as input in a threshStruct structure.

The fields within each of the input structures must be named the same way.

The following instructions can be used to generate the input structures used in this function. First, generate the expression matrix, gene array, and expression threshold variable as normal (i.e., as would normally be done when running GIMME). Next, prepare empty structures using, for example, "expressStruct = struc()". Finally, populate each structure with the corresponding lists. For example, add expression data for species A to the expression structure using "expressStruct.speciesA = expression_speciesA", where expression_speciesA is the expression matrix for species A.

CORE METABOLISM REDUNDANCY FUNCTIONS

TNCORE_REDUNDANCY

Description

A function to generate a population of randomized core metabolic models, with or without the presence of Tn-seq data. If Tn-seq data is included, the script will return two COBRA formatted core models, the one most consistent with the Tn-seq data, and the one with the highest objective flux. Regardless of if Tn-seq data is provided, several files will be returned for use with the *tncore_matrix* script to generate summary matrixes representing the variation and redundancy within all core model that were produced.

Usage

[coreGeneVar, coreRxnVar, coreGrowthVar, coreModel, coreModelFast, reducedModel, … coreGeneCat, genePresence, rxnPresence] = tncore_redundancy(model, iters, … growthFrac, tnseq, coreGenes, essThresh, binThresh, rnaseq, expressThresh, method)

Inputs

model: A COBRA formatted, genome-scale metabolic network reconstruction from which the core metabolic models are to be derived. The model must include gene-reaction associations. Exchange bounds must be set to represent the desired growth environment and nutrient uptake rate prior to the use of the model in this script.

Optional inputs

iters: The number of random core metabolic models to be produced. If no value is provided, the default value is 1,000.

growthFrac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. (Default = 0.5)

tnseq: The Tn-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct essentiality thresholds to be determined. This variable should be a matrix, with the first column containing the Tn-seq data (not log transformed), and the second column containing the gene names. If no Tn-seq data is provided, the *coreModel* and *coreModelFast* outputs are not returned.

coreGenes: A list of genes to be protected (i.e., not directly deleted) during generation of the core metabolic models. Note that a gene in this list may still be removed from the model if its associated reaction is constrained due to the deletion of another gene. This field can either be empty (no core genes are set prior to core model generation), contain a list of genes (genes to be protected), or set to {1} (core genes are automatically determined by the script on the basis of the Tn-seq data as those with a log-transformed value < the median - 3.5 * the standard deviation).

essThresh: The threshold (in number of standard deviations from the mean of the log of the Tn-seq data) for a gene to be considered essential. Used if *coreGenes* is set to {1}. If no value is given, the default is 3.5 (i.e., genes with an essentiality value more than 3.5 standard deviations below the mean value are considered essential).

binThresh: An array containing three values (number of standard deviations away from the mean of the log of the Tn-seq data) to use for setting the limits of each bin. If no value is given, the default array is: {'3.5'; '2.5'; '1.5'} (i.e., the bin boundaries are 3.5, 2.5, and 1.5 standard deviations below the mean value).

rnaseq: The RNA-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct expression thresholds to be determined if not set by the user. This variable should be a matrix, with the first column containing the RNA-seq data (not log transformed), and the second column containing the gene names. Genes above the expression threshold are included in the core gene list, and protected during core model generation; however, a highly expressed gene may still be removed from the model if its associated reaction is constrained due to the deletion of another gene. RNA-seq data can only be provided if Tn-seq data is also provided.

expressThresh: The threshold for a gene to be considered expressed. If no value is provided, the default is set to 0.02% the sum of all expression values.

method: Set the script to use either the FBA or MOMA algorithms when calculating growth rates of gene deletion mutants. Use 1 for FBA and 2 for MOMA. (Default = 1)

Outputs

coreGeneVar: A matrix where each column represents one core model, and each row represents a gene from the reduced input model. If the gene was included as part of the core model, the gene name is given. If the gene was not included as part of the core model, the gene is given but appended with '_deleted'.

coreRxnVar: A matrix where each column represents one core model, and each row represents a reaction from the reduced input model. If the reaction was included as part of the core model, the reaction ID is given. If the reaction was not included as part of the core model, the reaction ID is given but appended with '_deleted'. Models (i.e., columns) are in the same order as those in the *coreGeneVar* output.

coreGrowthVar: A matrix consisting of one row, with each column representing one core model. The objective flux rate for each of the core models is indicated. The values are provided in the same order as the models in the *coreGeneVar* output; i.e., the objective flux in the first cell corresponds to the model of the first column of the *coreGeneVar* variable.

coreModel: The COBRA formatted core model derived from the input model that was most consistent with the Tn-seq data. This output is not provided if no Tn-seq data is given as input.

coreModelFast: The COBRA formatted core model derived from the input model that had the highest rate of flux through the objective reaction. This output is not provided if no Tn-seq data is given as input.

reducedModel: A reduced version of the input model. This model is the input model, but with all reactions involving dead-ends metabolites removed from the model, and with all 'or Unknown' GPRs removed.

coreGeneCat: Contains four rows corresponding to the different gene fitness categories, with each column corresponding to one model in the same order as the *coreGeneVar* variable. Each cell contains the number of genes in the model that was grouped into that fitness category based on the Tn-seq data. The top row are non-essential genes, the second row are genes whose deletion results in a moderate growth impairment, the third row are genes whose deletion results in a severe growth impairment, and the fourth row are the essential genes.

genePresence: The same as *coreGeneVar* except that gene presence is indicated by a '1' and gene absence is indicated by a '0'.

rxnPresence: The same as *coreRxnVar* except that reaction presence is indicated by a '1' and reaction absence is indicated by a '0'.

Notes

This function can be used to produce context-specific core metabolic models, instead of using *tncore_core*. However, this is generally not recommended. Although our tests suggest that the core models returned through this process are of high-quality, so are those produced through the *tncore_core* script. However, the large number of core models that must be produced by *tncore_redundancy* in order to ensure the optimal model is produced is high, meaning large computational resources are required; *tncore_core* on the other hand is very quick.

The use of the MOMA algorithm is generally not recommended. While the output may differ, this algorithm is significantly slower, and the core models produced using the MOMA algorithm may contain no flux through the objective function if solved using FBA.

When many iterations are run, the output files *coreGeneVar*, *coreRxnVar*, *genePresence*, and *rxnPresence* are too big to be saved as a standard '.mat' file. In this case, it is recommended to convert the cell arrays to tables, and to export the tables as text files.

TNCORE_RANDOMIZE

Description

A function to generate a randomized core metabolic model. It is called during the running of *tncore_redudancy*, but can also be used as a stand alone script.

Usage

```
[coreModel, genesTnseq, geneGrouping, solution, reactions, rxnPresence, genePresence] = …
    tncore_randomize(model, growthThresh, nonProtected, modelTnseq, medianValue, …
    stdev, binThresh, solver, method)
```

Inputs

model: A COBRA formatted, genome-scale metabolic network reconstruction from which the core metabolic models are to be derived. The model must include gene-reaction associations. Exchange bounds must be set to represent the desired growth environment and nutrient uptake rate prior to the use of the model in this script.

Optional inputs

growthThresh: The minimum allowable objective flux in the generated core models. If no value is provided, the default threshold is set to 10% the objective flux of the input model.

nonProtected: A list of genes that can be the script can attempt to delete during construction of the core metabolic model. If no genes are provided, a default list is prepared consisting of genes whose deletion results in an objective flux below the growth threshold.

modelTnseq: The Tn-seq data. Data should be provided only for the genes present in the model. This variable should be a matrix, with the first column containing the log-transformed Tn-seq data, and the second column containing the gene names.

medianValue: The median of the log-transformed Tn-seq data for all genes in the genome, following the removal of outlier values. If no Tn-seq data is provided, this field is left empty.

stdev: The standard deviation of the log-transformed Tn-seq data for all genes in the genome, following the removal of outlier values. If no Tn-seq data is provided, this field is left empty.

binThresh: An array containing three values (number of standard deviations away from the mean of the log of the Tn-seq data) to use for setting the limits of each bin. If no value is given, the default array is: {'3.5'; '2.5'; '1.5'} (i.e., the bin boundaries are 3.5, 2.5, and 1.5 standard deviations below the mean value).

solver: The Cobra LP solver to be used. If no input is given, the default is set to the currently set Cobra LP solver.

method: Set the script to use either the FBA or MOMA algorithms when calculating growth rates of gene deletion mutants. Use 1 for FBA and 2 for MOMA (Default = 1)

Outputs

coreModel: The COBRA formatted core model produced from the input model.

geneTnseq: A list of the Tn-seq values for the genes included in the core model. Values are provided in the order of genes in the input model genes field.

geneGrouping: Contains four cells corresponding to the different gene fitness categories. Each cell contains the number of genes in the model that was grouped into that fitness category based on the Tn-seq data. The top row are non-essential genes, the second row are genes whose deletion results in a moderate growth impairment, the third row are genes whose deletion results in a severe growth impairment, and the fourth row are the essential genes.

solution: The objective flux rate for the core models is indicated, determined with the *optimizeCbModel* function of the COBRA Toolbox.

reactions: A list indicating which reactions were included in the core model. If the reaction was included as part of the core model, the reaction ID is given. If the reaction was not included as part of the core model, the reaction ID is given but appended with '_deleted'.

rxnPresence: A binary matrix indicating if each reaction present in the input model was included in the core model (1) or not included in the core model (0). Numbers correspond to the reaction in the same position of the input model's rxn field.

genePresence: A binary matrix indicating if each gene present in the input model was included in the core model (1) or not included in the core model (0). Numbers correspond to the gene in the same position of the input model's genes field.

TNCORE_MATRIX

Description

A function to generate a few matrixes summarizing the variation between the population of core models generated with the *tncore_core* script. The output matrixes can be directly used for the generation of figures.

Usage

[coocurMatrix, coocurMatrixAdj, growthMatrix, varPresenceLabel, uniqueModels] = …
tncore_matrix(model, variable, growth, varPresence, type, minPresence)

Inputs

model: The COBRA formatted model from which the core metabolic models was generated. If the core models were produced with the *tncore_core* script, use the *reducedModel* output from the *tncore_ core* script.

variable: The *coreGeneVar* or the *coreRxnVar* output variable from the *tncore_core* script.

growth: The *coreGrowthVar* output variable from the *tncore_core* script.

varPresence: The *genePresence* or *rxnPresence* output variable from the *tncore_core* script.

Optional inputs

type: The type of matrixes to produce. If gene matrixes are to be produced, use 1. If reaction matrixes are to be produced, use 2. Ensure that all input variables correspond to the correct type of matrixes to be produced. If no value in given, the default is 1.

minPresence: The minimum number of models a gene or reaction must be present in to be included within the output matrixes. If no value is given, the default is one model.

Outputs

coocurMatrix: A matrix indicating the total number of times each pair of genes or reactions occurred in the same core model. Each column and each row represents a gene, and each cell indicates the number of times those two genes were in the same model. Only genes or reactions included in at least as many models indicated in the *minPresence* input. The first row/column contains the gene name.

coocurMatrixAdj: A matrix that provides a Chi-squared statistic to indicate the significance of observed over- or under-representation of each pair of genes or reactions co-occurring in the same core model. Each column and each row represents a gene, and each cell contains a Chi-squared statistic indicating if the genes or reactions are more likely than expected to occur in the

same core model (a positive number) or more likely than expected to not occur in the same core model (a negative number). Only genes or reactions included in at least as many models indicated in the *minPresence* input. The first row/column contains the gene name.

growthMatrix: A matrix indicating the average growth rate of core models containing the indicated gene, the average growth rate of core models not containing the indicated gene, and the number of core models that the gene is included within.

varPresenceLabel: The *varPresence* input variable modified to row and column labels, and with genes or reactions that are not present in at least the number of models specified by the *minPresence* field removed.

uniqueModels: Indicates how many unique models were present in the core model population, based either on the genes present in the models (type = 1) or the reactions present in the models (type = 2).

TNCORE_COMPARE

Description

A function to compare the frequency that genes or reactions occur in core model populations generated from independent runs of the *tncore_redundancy* script. For example, if *tncore_redundancy* is run multiple times using different growth thresholds, this script can take the output from the *tncore_redundancy* runs, and indicate the frequency that each gene or each reaction is found in the core models for each growth threshold.

Usage

[varPresenceMatrix] = tncore_compare(varNames, varPresenceArray, headers)

Inputs

varNames: A cell array containing the gene names or the reaction IDs in the same order as the genes or reactions are given in the *genePresence* or *rxnPresence* variables.

varPresenceArray: A cell array containing the names of the *genePresence* or *rxnPresence* outputs from the multiple *tncore_redundancy* runs. For example, if *tncore_redundancy* was run three times, there should be three *genePresence* variables (e.g., genePres1, genePres2, genPres3). In this case, this variable should contain the following: {'genePres1', 'genePres2', 'genePres3'}.

Optional inputs

headers: A cell array containing the names to use as the column labels in the *varPresenceMatrix* output variable. If this is empty, the default is to use the *varPresenceArray* as the column labels.

Outputs

varPresenceMatrix: A matrix indicating the percent of core models in each core model population that contained the gene or reaction. Each row corresponds to a gene or reaction, and each column corresponds to a different set of core model populations. Column and row labels are included.

TNCORE_RECONSTRUCT

Description

A function to reproduce a core metabolic model based on a binary gene presence/absence list. Developed in order to allow reconstruction of any of the core models produced during random model core generation by the *tncore_redundancy* script using the *reducedModel* output and one column of the *genePresence* output table.

Usage

[coreModel] = tncore_reconstruct(model, genePresence, trim)

Inputs

model: The COBRA formatted metabolic model from which the core model is derived. If reconstructing a core model generated by the *tncore_redundancy* function, use the *reducedModel* exported by the *tncore_redundancy* script.

genePresence: A binary array indicating which genes in the input model are presence (1) or absence (0) in the core model to be produced. Can use one column of the *genePresence* output of the *tncore_redundancy* script.

Optional inputs.

trim: Indicates if non-essential reaction that are either non associated with any gene or have only an 'Uknown' GPR should be removed from the core model. Use 1 for trim, and 0 for do not trim. If no value is given, the default is to remove these reactions.

Outputs

coreModel: The core model produced from the input model and containing only the genes and associated reactions of the *genePresence* input.

MODEL MANIPULATION AND REFINEMENT FUNCTIONS

TNCORE_REFINE

Description

This function is an extension of the *tncore_core* script that first generates a core model consistent with Tn-seq data, and then uses this core model to refine the GPRs of the input genome-scale metabolic model. In essence, the script first generates a core model consistent with the Tn-seq. Once the core model is produced, any genes essential in the Tn-seq data but not the core model are identified. At this point, for any reaction in the core model that contains an essential genes based on the Tn-seq data, the GPRs for these reactions in the input model are replaced with the corresponding GPRs of the core model. Additionally, if two or more of these genes are associated with the same, and only the same, reaction(s), for any reaction that has only 'or' statements in the GPR, it is suggested to consider replacing the 'or' statements to 'and' statements. The function also provides a summary table.

Usage

[refinedModel, report] = tncore_refine(model, tnseq, epsilon, binThresh, essThresh, … growthFrac, deadends)

Inputs

model: A COBRA formatted, genome-scale metabolic network reconstruction from which the core metabolic models are to be derived. The model must include gene-reaction associations. Exchange bounds must be set to represent the desired growth environment and nutrient uptake rate prior to the use of the model in this script.

tnseq: The Tn-seq data. Data should be provided for all genes in the organism, not only the genes in the model, in order to allow correct essentiality thresholds to be determined. This variable should be a matrix, with the first column containing the Tn-seq data (not log transformed), and the second column containing the gene names.

Optional inputs

epsilon: The epsilon value (the minimum flux rate that is considered non-zero) to be used during the running of FASTCC and FASTCORE during the preparation of a context-specific core model. If not value is given, the default value is 1.1e-6.

binThresh: An array containing three values (number of standard deviations away from the mean of the log of the Tn-seq data) to use for setting the limits of each bin. If no value is given, the default array is: {'3.5'; '2.5'; '1.5'} (i.e., the bin boundaries are 3.5, 2.5, and 1.5 standard deviations below the mean value).

essThresh: The threshold (in number of standard deviations from the mean of the log of the Tn-seq data) for a gene to be considered essential. Genes considered to be essential are protected during core model generation. However, an essential gene may still be removed from the model

if its associated reaction that is constrained due to the deletion of another gene. If no value is given, the default is 3.5 (i.e., genes with an essentiality value more than 3.5 standard deviations below the mean value are considered essential).

growthFrac: The minimum allowable objective flux in the generated core model, as a fraction of growth of the input model. If no value is given, the default is 0.5 (i.e., half the growth of the full input model.

deadends: To remove reactions producing dead-ends from the output, refined model, use 1. TO leave reactions producing dead-ends in the output, refined mode, use 0. If no value is provided, the default is to not remove the reactions producing dead-ends.

Outputs

refinedModel: A COBRA formatted model of the original, input genome-scale model, but with the GPRs of reactions associated with essential genes updated based on the GPRs of the core model, and optionally with dead-ends removed.

report: A table displaying the changes made in preparing the refined model. Also includes a list of additional potential changes that are not included in the refined model. The following five columns are provided: 'Reaction' – the reaction ID of the reaction; 'Type of change" – the type of change made. Can be 'GPR modification' (the genes associated with the reaction were modified), 'OR to AND transition' (a suggested to change the OR statements in the GPR to AND statements, or 'Deadend reaction deletion' (the reaction was deleted because it produced a deadend metabolite); 'Original GPR' – the GPR of the reaction in the input model; ' New GPR' – the GPR of the reaction in the output model, or when the change is not made in the model, the suggestion of what the GPR should be modified to; 'Is change made in refined model' – a yes/no column indicating if the output refined model already contains the change.

Notes

By default, the reactions that produce dead-end metabolites are not removed from the core model produced during the running of this function. If you wish to modify the script to remove reactions producing dead-end metabolites from the core model, this can be done on line 87 of this script by changing the '0' to '1'.

For additional notes, refer to the section above on the function *tncore_core*.

TNCORE_EXPAND

Description

A function to prepare an expanded model based on two input models. Takes as input a model to which new reactions should be added (for example, a core model produced using the Tn-Core Toolbox), and a second genome-scale metabolic model of the same organism. The output model is an expanded version of the first model: with all reactions of the first model supplemented with all reactions found uniquely in the second model. GPRs are transferred with the reactions.

Usage

expandedModel = tncore_expand(coreModel, fullModel, deadends)

Inputs

coreModel: The model to serve as the bases of the expanded model, to which the reactions of the second model will be added.

fullModel: A model from which reactions will be identified to be transferred to the core model.

Optional inputs

deadends: Indicates if reactions producing dead-end metabolites should be removed from the expanded model (after transferring all new reactions). Use 1 for yes and 0 for no. If no input is given, the default is to remove the reactions producing dead-end metabolites.

Outputs

expandedModel: The output model, consisting of the core model with the additional reactions of the full model.

Notes

For reactions in common between the two models, the features of the reaction in the expanded model are the same as in the core model; i.e., the GPRs are not updated for common reactions.

For reactions with the same name but different equation (either different metabolites or different stoichiometry), both reactions will be included in the expanded model, but the name of the reaction coming from the full model will be modified.

TNCORE_REMOVE

Description

A function to remove genes in the gene list that are no longer found in the grRules field due to the deletion of reactions from the model. Genes are removed from the genes field, and the rules and rxnGeneMat fields are updated accordingly.

Usage

[modelNew] = tncore_remove(model)

Inputs

model: The model from which to remove the genes. The script will first identify which genes are no longer present in the grRules field, and then remove them from the model.

Outputs

modelNew: The input model, but with the genes not present in the grRules removed from the genesfield, and with the rules and rxnGeneMat fields updated accordingly.

TNCORE_DELETE

Description

A function to remove any gene appended with '_deleted' in the genes field (as is done using the *deleteModelGenes* script of the COBRA Toolbox) from the genes and grRules field, and to update the rules and rxnGeneMat fields accordingly. Note that it does not remove any reactions from the model. Can be useful following the combined use of *deleteModelGenes* and *removeRxns* of the COBRA Toolbox.

Usage

[modelNew] = tncore_delete(model)

Inputs

model: The model from which to delete the genes. At least one gene must be appended with '_deleted'.

Outputs

modelNew: The input model, but with the deleted genes removed from the genes and grRules fields, and with the rules and rxnGeneMat fields updated accordingly.

TNCORE_REMOVE_REACTIONS

Description

A function to remove reactions from the model, and to delete unused metabolites and genes.

Usage

[modelNew] = tncore_remove_reactions(model, reactions)

Inputs

model: The model from which to delete the reactions.

reactions: An array of reactions to delete from the model.

Outputs

modelNew: The input model, but with the reactions of the *reactions* input variable deleted, and with unused metabolites and genes removed.

Notes

This function was developed as at least old versions of the *removeRxns* function of the COBRA Toolbox may have issues if the number of reactions equals the number of metabolites. When the number of metabolites does not equal the number of reactions, this function simply calls the *removeRxns* function as it is significantly faster.

TNCORE_DUPLICATES

Description

A function to remove duplicate genes from a model. If two genes with the same name are found in the gene list, they will be replaced with a single entry. Updates the genes, rules, and rxnGeneMat fields appropriately. Also removes any genes that are not associated with a reaction.

Usage

modelNew = tncore_duplicates(model)

Inputs

model: The model from which to remove the duplicate genes.

Output

modelNew: The input model, but with the duplicated genes replaced with a single gene entry, and with the grRules, rules, and rxnGeneMat fields updated accordingly.

TNCORE_FIX

Description

A function to add the following fields to a model, if they are missing: rxnGenemat, grRules, rules. Many functions of the Tn-Core Toolbox require these fields; however, not all versions of the COBRA Toolbox automatically prepare these fields during import into a COBRA model from SBML format. In order to work correctly, at least one of the grRules or rules fields must be present.

Usage

modelNew = tncore_fix(model)

Inputs

model: The model to which the missing fields should be added.

Output

modelNew: The input model with the missing fields added. Existing fields are not modified.

TNCORE_DEADENDS

Description

A function to iteratively remove all reactions involving deadend metabolites. Results in an output model that lacks deadends.

Usage

reducedModel = tncore_deadends(model)

Inputs

model: The model from which the deadends will be removed.

Output

reducedModel: The input model with deadends removed.