|  |
| --- |
| Script Files:  All script files within the package have a script prefix in their names.  Example: 1. File for runnin the ecoli model is named: script\_ecoli.m  2. File for running the toy model is named: script\_toy.m |
| modelgen.m |
| Model generation from text files:  Function call:  [model,parameter,variable,nrxn,nmetab] = modelgen(rxfname)  Input:  rxnfname: Full path name of tab delimited text file containing model information (see below for more information)  Outputs:  model: MATLAB structure containing all fields required for kinetic model and FBA analysis of netabolic networks  parameter: MATLAB Structure containg all kinetic model parameters (given in the input file only) as sparce matrices  variable: not utilized (do not remove - required for future functionality)  nrxn: Number of reactions in metabolic network  nmetab: Number of metabolites in metabolic network  The package allows the creation of an FBA style model from an excel file that is saved in a tab delimited format as a text file. This file is generic for all types of models. An example of a tab delimited model file with all requisite columns can be found in examplemodel.xlsx and the corresponding tab delimited text file examplemodel.txt  The creation of the model requires the tab delimited file to have the exact number and designations of columns as specified in examplemodel. Any deviations will result in an error  Torubleshooting:  If error occurs despite following the above instructions, make sure the tab delimited file does not have extra lines appended to it after the last line of text in the corresponding excel file. Appended lines should be removed manually from the tab delimited file and the file is to be saved.  Dependencies:  ToColumnVector.m  defparval.m  extract\_par.m  fluxIndex.m  isemptyr.m |
| FBAfluxes.m |
| Evaluate FBA or pFBA steady state flux distribution for any given MATLAB model structure  Function Call:  model = FBAfluxes(model,option,ess\_rxn,Vup\_struct,prxnid)  Inputs:  model: MATLAB model structure with fields vl, vu, c and b  Optional Inputs:  option: {'fba'} or 'pFBA' for FBA or pFBA analysis based flux determination respectively  ess\_rxn: cell array of exchnage reaction names that cannot have a zero flux lower bound i.e. flux bounds should be reversible. All other exchange reaction lower bounds are fixed to 0.  Example: ess\_rxn = {'exO2','exH', 'exH2O'}  Vup\_struct: MATLAB structure of fixed uptake fluxes for calculation of FBA flux distributions with reaction names as field names and fluxes as field values.  Example: Vup\_struct.exGLC = 20  prxnid: index of target reaction that is the objective for maximization and/or minimization. The biomass reaction is the target by default.  Output:  model: MATLAB structure with new field values from 'fba' or 'pfba' for the steady state fluxes in Vss  Dependencies:  solveLP.m  run\_pFBA.m  changebounds.m  fixUptake.m  cplexlp.m and cplexlp.p  convertIrreversible.m  fluxIndex.m  columnVector.m |
| parallel\_sampling.m |
| Wrapper for ACHR based sampling of metabolite concentrations using reaction equailibrium calculations  Function Call:  [mc,pvec,smp] = parallel\_sampling(model,parameter,input,setupfun,nsample)  Inputs  model: MATLAB structure with fields as required by other dependencies and wrapping functions  parameter: MATLAB structure with fields K (sparse vector of Michaelis binding constants of size number of metabolites x number of reactions) Klb (Lower bounds for K values), Kub (upper bounds for K values), KIact (sparse array of activation constants of size number of metabolites x number of reactions), KIihb (sparse array of inhibition constants of size number of metabolites x number of reactions), Vmax (vector of enzyme concentration values in Vmax = kcat\*ei of size number of reactions), kact\_fwd (vector of enzyme turnover numbers for reactions in the forward direction), kcat\_bkw (vector of enzyme turnover numbers for reactions in the reverse direction. It has a value of zero for irreversible reactions.)  input: MATLAB structure with field names set to external metabolite names and field values set to corresponding concentrations. Typically used to specify constant external metabolite concentrations.  Example: met.h2o\_c = 55.0 for [h2o[c]] = 55.0 M  met.o2\_e = 0.05 for [o2[e]] = 0.05 M  setupfun: function name/handle to be called to set the linear programming problem to solve for ACHR  nsample: number of metabolite concentration samples to be obtained  Outputs:  mc: vector of metabolite concentrations obtained using ACHR sampling  parameter: MATLAB structure with additional field delGr (vector of Gibb’s free energy of all reactions within the network for concentrations in mc)  smp: cell array of nsample x 3 for nsample metabolite concentrations. Each of the nsample rows has the mc vector and the modified parameter structure in 2 columns respectively.  Dependencies  getiConEstimate.m  iconcentration.m  ACHRmetSampling.m  assignConc.m  assignRxns.m  getdelGr.m  separate\_slack.m  setupSlackVariables.m  solvemetLP.m  cplexlp.m and cplexlp.p |
| parallel\_ensemble.m |
| Sample kinetic model parameters for a given metabolic network with known metabolite concentrations. Sampling of parameters follows the method of ORACLE using a Monte Carlo approach. More information on the methods can be obtained from the reference.  Function Call:  ensemble = parallel\_ensemble(model,mc,parameter,rxn\_add,rxn\_excep,nmodels,smp)  Input:  model: MATLAB model structure  mc: vector of (initial or otherwise) metabolite concentrations for kinetic model analysis  parameter: MATLAB structure with fields K (sparse vector of Michaelis binding constants of size number of metabolites x number of reactions) Klb (Lower bounds for K values), Kub (upper bounds for K values), KIact (sparse array of activation constants of size number of metabolites x number of reactions), KIihb (sparse array of inhibition constants of size number of metabolites x number of reactions), Vmax (vector of enzyme concentration values in Vmax = kcat\*ei of size number of reactions), kact\_fwd (vector of enzyme turnover numbers for reactions in the forward direction), kcat\_bkw (vector of enzyme turnover numbers for reactions in the reverse direction. It has a value of zero for irreversible reactions.)  Optional Inputs:  rxn\_add: Cell array of reactions whose kinetics is to be considered for calculation of fluxes in addition to intracellular reactions accounted in index vector Vind  rxn\_excep: reactions not to be considered for kinetic flux determination even if described otherwise by the model  nmodels: number of models whose fluxes are to be analyzed kinetically  smp: number of metabolite concentration samples Outputs:  ensemble: Cell of array sampled models using Monte Carlo techniques. The number of rows is same as the number of models requested in the input nmodels. Each row has 2 columns for the model’s corresponding metabolite concentration and the model itself. Does not support multiple metabolite concentration samples yet.  Dependencies:  buildmodels.m  estimateKm.m  iflux.m  CKinetics.m  TKinetics.m  EKinetics.m  addToVind.m  samplekcat.m  scale\_flux.m |
| buildmodels.m |
| Build ensemble of models based on given metabolite concentration. Only sample parameters that are unknown. All known/specified parameters are left as is  Function Call:  parameter = buildmodels(model,parameter,mc,rxn\_add,rxn\_excep)  Inputs:  model: MATLAB model structure  parameter: MATLAB structure with fields K (sparse vector of Michaelis binding constants of size number of metabolites x number of reactions) Klb (Lower bounds for K values), Kub (upper bounds for K values), KIact (sparse array of activation constants of size number of metabolites x number of reactions), KIihb (sparse array of inhibition constants of size number of metabolites x number of reactions), Vmax (vector of enzyme concentration values in Vmax = kcat\*ei of size number of reactions), kact\_fwd (vector of enzyme turnover numbers for reactions in the forward direction), kcat\_bkw (vector of enzyme turnover numbers for reactions in the reverse direction. It has a value of zero for irreversible reactions.)  mc: vector of (initial or otherwise) metabolite concentrations for kinetic model generation through Monte Carlo methods and subsequent analysis  Optional Inputs:  rxn\_add: Cell array of reactions whose kinetics is to be considered for calculation of fluxes in addition to intracellular reactions accounted in index vector Vind  rxn\_excep: reactions not to be considered for kinetic flux determination even if described otherwise by the model  Outputs:  parameter: MATLAB structure with fields K (sparse vector of Michaelis binding constants of size number of metabolites x number of reactions) Klb (Lower bounds for K values), Kub (upper bounds for K values), KIact (sparse array of activation constants of size number of metabolites x number of reactions), KIihb (sparse array of inhibition constants of size number of metabolites x number of reactions), Vmax (vector of enzyme concentration values in Vmax = kcat\*ei of size number of reactions), kact\_fwd (vector of enzyme turnover numbers for reactions in the forward direction), kcat\_bkw (vector of enzyme turnover numbers for reactions in the reverse direction. It has a value of zero for irreversible reactions.)  Dependencies:  addToVind.m  estimateKm.m  samplekcat.m  CKinetics.m  Tkinetics.m  iflux.m |
| iconcentration.m |
| Assign user defined concentrations to the concentration vector  Function Call:  [varargout] = iconcentration(model,input,mc\_lb,assignFlag)  Inputs:  model: MATLAB model structure with required fields for all dependencies  input: MATLAB structure with field names set to external metabolite names and field values set to corresponding concentrations. Typically used to specify constant external metabolite concentrations.  Example: met.h2o\_c = 55.0 for [h2o[c]] = 55.0 M  met.o2\_e = 0.05 for [o2[e]] = 0.05 M  Optional Inputs:  mc\_lb: user defined concentration lower bounds  assignFlag: logical vector of size # metabolites x 1 with a TRUE value indicating that the corresponding metabolite already has an assigned/predetermined concentration. FALSE values indicate the correspoding metabolite has not been assigned a value.  Outputs:  mc\_lb: vector of metabolite concentration lower bounds  mc\_ub: vector of metabolite concentration upper bounds. It is set to mc\_lb if metabolite concentration is to be set to a single value as opposed to one between lower and upper bounds.  assignFlag: logical vector of size # metabolites x 1 with a TRUE value indicating that the corresponding metabolite already has an assigned/predetermined concentration. FALSE values indicate the correspoding metabolite has not been assigned a value. |
| getiConEstimate.m |
| Wrapper to obtain one set of metabolite concentrations by solving the linear program for thermodynamic feasibility of reactions within a metabolic network. Distance from thermodynamic equilibrium can be arbitrarily specified using slack variables.  Function Call:  [mc,assignFlag,delGr,model,vCorrectFlag] = getiConEstimate(model,setupfun)  Inputs:  model: MATLAB model structure with all relevant fields  setupfun: function name/handle to be called to set the linear programming problem to solve for ACHR  Outputs:  mc: vector of metabolite concentrations obtained using ACHR sampling  assignFlag: logical vector with TRUE values for all metabolites whose concentrations are assigned and FALSE for all metabolites whose concentrations are unassigned.  delGr: vector of Gibb’s free energy of all reactions within the network for concentrations in mc  model: MATLAB model structure  vCorrectFlag: vector of logicals with TRUE corresponding to all reactions whose directionality matches corresponding values in delGr and FALSE otherwise  Dependencies:  setupSlackVariables.m  solvemetLP.m  separate\_slack.m  assignConc.m  assignRxns.m  getdelGr.m  cplexlp.m and cplexlp.p |
| ACHRmetsampling.m |
| Sample data points from a convex solution space defined using constraints as in a linear programming problem. Function Call:  [pts,assignFlag,delGr,vCorrectFlag] = ACHRmetSampling(model,nFiles,nptsPerFile,stepsPerPnt)  Input:  model: MATLAB model structure with all relevant fields  Optional Inputs:  nFiles:  nptsPerFile: number of points to be sampled from the solution space  stepsPerPnt: number of ACHR steps to be taken within the solution space before a new point is recorded  Outputs:  pts: array of sampled data point vectors  assignFlag: logical vector with TRUE values for all metabolites whose concentrations are assigned and FALSE for all metabolites whose concentrations are unassigned.  delGr: vector of Gibb’s free energy of all reactions within the network for concentrations in pts  vCorrectFlag: vector of logicals with TRUE corresponding to all reactions whose directionality matches corresponding values in delGr and FALSE otherwise  Dependencies:  setupMetLP.m  setupSlackVariables.m  createWarmupPoints.m  separate\_slack.m  assignConc.m  assignRxns.m  cplexlp.m and cplexlp.p  ToColumnVector.m  fluxIndex.m  getdelGr.m  solvemetLP.m |
| run\_pFBA.m |
| Wrapper to run pFBA using the kinetic model MATLAB structure  Function Call:  [model,Vss] = run\_pFBA(model,ess\_rxn,Vup\_struct)  Inputs:  model: MATLAB structure with fields as required by other dependencies and wrapping functions  ess\_rxn: cell array of exchnage reaction names that cannot have a zero flux lower bound i.e. flux bounds should be reversible. All other exchange reaction lower bounds are fixed to 0. Default is an empty cell array.  Example: ess\_rxn = {'exO2','exH', 'exH2O'}  Vup\_struct: MATLAB structure of fixed uptake fluxes for calculation of FBA flux distributions with reaction names as field names and fluxes as field values. Currently not in use. Default is empty.  Example: Vup\_struct.exGLC = 20  Outputs:  model: MATLAB structure with new field values from 'fba' or 'pfba' for the steady state fluxes in field Vss  Vss: model steady state fluxes obtained using pFBA  Dependencies:  covertIrreversible.m  solveLP.m  cplexlp.m and cplexlp.p  changebounds.m  fluxIndex.m  columnVector.m |
| solveLP.m |
| Wrapper to set up and solve linear programming problems using CPLEX for flux calculation in FBA or pFBA mode.  Function Call:  [vLPmax,vLPmin,model] =...  solveLP(model,bounds,ess\_rxn,prxnid,Vup\_struct,fixgrowth)  Inputs:  model: MATLAB structure with requisite fields for all function dependencies (see below for more information)  Optional Inputs:  bounds: MATLAB structure with fields vl(flux lower bounds), vu(flux upper bounds) and Vuptake(flux vector with non zero values for fixed uptake fluxes in the model). Default field values are set by calling changebounds.m  ess\_rxn: cell array of exchnage reaction names that cannot have a zero flux lower bound i.e. flux bounds should be reversible. All other exchange reaction lower bounds are fixed to 0. Default is an empty cell array.  Example: ess\_rxn = {'exO2','exH', 'exH2O'}  prxnid: index of target reaction that is the objective for maximization and/or minimization. The biomass reaction is the target by default.  Vup\_struct: MATLAB structure of fixed uptake fluxes for calculation of FBA flux distributions with reaction names as field names and fluxes as field values. Currently not in use. Default is empty.  Example: Vup\_struct.exGLC = 20  Outputs:  vLPmax: MATLAB structure with maximization problems with fields v (optimized flux vector), obj (value of the objective function), flag (optimization flag - refer to CPLEX or MATLAB documentation on optimization for more details)  vLPmin: MATLAB structure for minimization problems with the same fields as vLPmax  model: MATLAB model structure with additional fields vl (vector of flux lower bounds), vu (vector of flux upper bounds), c (sparse vector with index of nonzero value(s) corresponding to the target objective reaction and b (rhs constraint vector in Ax = b)  Dependencies:  changebounds.m  cplexlp.m and cplexlp.p |
| convertIrreversible.m |
| Converts all reversible fluxes in a network to a pair of irreversible fluxes. Any given MATLAB model structure can be converted to consist only of irreversible reactions.  Function Call:  [modelIrrev,matchRev,rev2irrev,irrev2rev] = convertIrreversible(model)  Input:  model : MATLAB model structure for converting reversible reactions to irreversible reaction pairs  Outputs:  modelIrrev: MATLAB model containing irreversible reactions  matchRev: matching pair indices of all reversible reactions in their irreversible form  rev2irrev: irreversible reaction index for a reversible reaction  irrev2rev: reversible reaction index for one of the irreversible reaction pair  Dependencies:  columnVector.m  fluxIndex.m |
| cplexlp.m and cplexlp.p |
| Solve linear programming problems using IBM ILOG CPLEX solver for MATLAB. See CPLEX documentation for more information. |
| columnVector.m |
| Performs the same function as ToColumnVector.m. Part of the COBRA Toolbox used for CBMs. Refer to COBRA Toolbox documentation for more information. |
| changebounds.m |
| Initialize and fix bounds for various reactions before performing FBA or pFBA. Allows to change the reversibility and bounds to usedefined values for model reactions whose bounds are predefined.  Function Call:  [model,bounds] = changebounds(model,ess\_rxn,bounds,fixgrowth)  Inputs:  model: MATLAB model structure with additional fields Vind, VFex, Vex  Optional Inputs:  ess\_rxn: cell array of exchnage reaction names that cannot have a zero flux lower bound i.e. flux bounds should be reversible. All other exchange reaction lower bounds are fixed to 0.  bounds: MATLAB structure with fields vl(flux lower bounds), vu(flux upper bounds) and Vuptake(flux vector with non zero values for fixed uptake fluxes in the model)  fixgrowth: if TRUE, fixes the biomass reaction bounds to the desired growth rate in model.gmax. Default value is FALSE.  Outputs:  model: MATLAB model structure with additional fields  bounds: MATLAB structure with fields vl(flux lower bounds), vu(flux upper bounds) and Vuptake(flux vector with non zero values for fixed uptake fluxes in the model) |
| fixUptake.m |
| Generates a nonzero vector of size nrxnsx1 whose nonzero values correspond to fixed exchnage fluxes and adds it as a field to the model structure  Function call:  model = fixUptake(model,Vup\_struct)  Inputs:  model: MATLAB structure with fields nt\_rxn (total number of reactions in model), rxns (cell array of all reaction names), Vuptake  Vup\_struct: MATLAB structure of fixed uptake fluxes for calculation of FBA flux distributions with reaction names as field names and fluxes as field values.  Example: Vup\_struct.exGLC = 20  Output:  model: MATLAB structure with additional field Vuptake |
| ToColumnVector.m |
| Convert row vectors to column vectors  Function Call:  columnvec = ToColumnVector(inputvec)  Input:  inputvec: Input vector that is to be converted to a column vector  Output:  columnvec: Output column vector form of input vector |
| defaprval.m |
| Assign default values to model parameters (1 is the default value)  Function call:  [par] = defparval(nterms,par)  Input:  nterms: number of default parameters that should be assigned  par: existing row vector of parameters to which new parameters should be appended to  Output:  par: New parameter row vector of size ntermsx1 or (length(par)+nterms)x1 |
| extract\_par.m |
| Extract parameters or parameter bounds for the kinetic model from a string  Function call:  [parameter,lb,ub] = extract\_par(par\_string)  Input:  par\_string: string of comma separted parameters or string of comma separated parameter bounds  Example: comma separated parameters: p1,p2,p3  comma separated bounds: [p1lb,p1ub],p3,[p3lb,p3ub]  Output:  parameter: row vector of parameters  lb: row vector of parameter lower bounds, if parameter bounds are specified in par\_string  ub: row vector of parameter upper bounds, if parameter bounds are specified in par\_string |
| fluxIndex.m |
| Calculate the index for all intracellular, exchange and trasport fluxes within the metabolic network. The biomass reaction index is also calculated and removed from consideration within other indices.  Function call:  [Vind,VFex,Vex,bmrxn] = fluxIndex(model,nt\_rxn,newS)  Inputs:  model: MATLAB structure with fields S(stoichiomteric matrix) and rxns(list of all reaction names)  nt\_rxn: total number of reactions in model  newS: S or the stoichiomertic matrix of the metabolic network  Output:  Vind: index of all intracellular reactions  VFex: index of all exchange reactions in a CBM  Vex: index of all transport reactions  bmrxn: index of biomass reaction |
| isemptyr.m |
| MATLAB Central file to calculate whether individual cells within a cell array are empty. |
| assignConc.m |
| Assigns concentrations based on metabolite order in MATLAB model structure field mets  Function Call:  [mc,assignFlag,delGr,vCorrectFlag] = assignConc(mc\_in,model,bounds)  Inputs:  mc\_in: vector of metabolite concentrations  model: MATLAB model structure with field mets  bounds: MATLAB structure with field mets corresponding to metabolite concentrations in mc\_in  Outputs:  mc: rearranged vector of metabolite concentrations corresponding to model.mets  assignFlag:  delGr:  vCorrectFlag:  Dependencies:  getdelGr.m |
| assingRxns.m |
| Assigns fluxes based on reaction order in MATLAB model structure field rxns  Function Call:  [vf,assignFlag] = assignRxns(flux\_in,model,bounds)  Inputs:  flux\_in: vector of reaction fluxes  model: MATLAB model structure with field rxns  bounds: MATLAB structure with field rxns corresponding to metabolite concentrations in flux\_in  Outputs:  vf: rearranged vector of reaction fluxes corresponding to model.rxns  assingFlag: |
| solveMetLP.m |
| Wrapper to call cplexlp.m to solve an LP to identify metabolite concentrations satisfying linear constraints  Function Call:  [LPmax,LPmin] = solvemetLP(bounds,prxnid)  Input:  bounds: MATLAB structure containing fields A(array of LP constraint coefficients), b(vector of constraint LHS), lb(metabolite concentration lower bounds, must be greater or equal to zero) and ub(metabolite concentration upper bounds)  prxnid: target metabolite concentration to be maximized or minimized. Should be fixed to zero if solving only a constraint programming problem and not an LP.  Outputs:  LPmax: Refer to solveLP.m description  LPmin: Refer to solveLP.m description |
| getdelGr.m |
| Supplies reaction Gibb’s free energy based on metabolite concentrations and standard reaction Gibb’s free energy  Function Call:  delGr = getdelGr(model,lnmc)  Inputs:  model: MATLAB model structure with fields rxns and Keq  lnmc: vector of logarithm of metabolite concentrations  Output:  delGr: vector of reaction Gibb’s free energies |
| imodel.m |
| Initializes model and solver parameters before solving ODEs using SUNDIALS TB  Function Call:  [model,solverP,saveData] = imodel(model,tmax,ess\_rxn,Vup\_struct)  Inputs:  model:  tmax:  ess\_rxn  Vup\_struct:  Outputs:  model:  solver:  saveData:  Dependencies:  fixUptake  changebounds  solveLP |