Metabotools tutorial II - Integration of quantitative metabolomic data

Maike Aurich

In this tutorial we ...

Clear workspace and initialize the COBRA Toolboxf

```
clear
initCobraToolbox
```

```
      _____   _____   _____   _____     _____     |
     /  ___| /  _  \ |  _  \ |  _  \   / ___ \    |   COnstraint-Based Reconstruction and Analysis
     | |     | | | | | |_| | | |_| |  | |___| |   |   The COBRA Toolbox - 2017
     | |     | | | | |  _  { |  _  /  |  ___  |   |
     | |___  | |_| | | |_| | | | \ \  | |   | |   |   Documentation:
     \_____| \_____/ |_____/ |_|  \_\ |_|   |_|   |   http://opencobra.github.io/cobratoolbox
                                                  |

 > Checking if git is installed ...  Done.
 > Checking if the repository is tracked using git ...  Done.
 > Checking if curl is installed ...  Done.
 > Checking if remote can be reached ...  Done.
 > Initializing and updating submodules ... Done.
 > Adding all the files of The COBRA Toolbox ...  Done.
 > Define CB map output... set to svg.
 > Retrieving models ...   Done.
 > TranslateSBML is installed and working properly.
 > Configuring solver environment variables ...
   - [----] ILOG_CPLEX_PATH: /opt/ibm/ILOG/CPLEX_Studio1271/cplex/matlab/x86-64_linux
   - [----] GUROBI_PATH: /home/syarra/Dropbox/software/gurobi/gurobi652/linux64/matlab
   - [----] TOMLAB_PATH :  --> set this path manually after installing the solver ( see instructions )
   - [----] MOSEK_PATH: /home/syarra/Dropbox/software/mosek/linux/8/
   Done.
 > Checking available solvers and solver interfaces ... Done.
 > Setting default solvers ... Done.
 > Saving the MATLAB path ... Done.
   - The MATLAB path was saved as ~/pathdef.m.

 > Summary of available solvers and solver interfaces

   Support     LP   MILP     QP   MIQP    NLP
 ----------------------------------------------------------------------
 cplex_direct  full              0      0      0      0      -
 dqqMinos      full              1      -      -      -      -
 glpk          full              1      1      -      -      -
 gurobi        full              1      1      1      1      -
 ibm_cplex     full              1      1      1      -      -
 matlab        full              1      -      -      -      1
 mosek         full              1      1      1      -      -
 pdco          full              1      -      1      -      -
 quadMinos     full              1      -      -      -      1
 tomlab_cplex  full              0      0      0      0      -
 qpng          experimental      -      -      1      -      -
 tomlab_snopt  experimental      -      -      -      -      0
 gurobi_mex    legacy            0      0      0      0      -
 lindo_old     legacy            0      -      -      -      -
 lindo_legacy  legacy            0      -      -      -      -
 lp_solve      legacy            1      -      -      -      -
 opti          legacy              0      0      0      0      0
 ----------------------------------------------------------------------
 Total         -                   9      4      5      1      2

 + Legend: - = not applicable, 0 = solver not compatible or not installed, 1 = solver installed.


 > You can solve LP problems using: 'dqqMinos' - 'glpk' - 'gurobi' - 'ibm_cplex' - 'matlab' - 'mosek' -
 > You can solve MILP problems using: 'glpk' - 'gurobi' - 'ibm_cplex' - 'mosek'
 > You can solve QP problems using: 'gurobi' - 'ibm_cplex' - 'mosek' - 'pdco' - 'qpng'
 > You can solve MIQP problems using: 'gurobi'
 > You can solve NLP problems using: 'matlab' - 'quadMinos'

 > Checking for available updates ...
ssh: /usr/local/MATLAB/R2016a/bin/glnxa64/libcrypto.so.1.0.0: no version information available (requirec
ssh: /usr/local/MATLAB/R2016a/bin/glnxa64/libcrypto.so.1.0.0: no version information available (requirec
OpenSSL version mismatch. Built against 1000207f, you have 100010bf
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
 > The changes of The COBRA Toolbox could not be fetched. > There are 169 new commit(s) on <master> and
 > You can update The COBRA Toolbox by running updateCobraToolbox() (from within MATLAB).
```

```
global CBTDIR

tol = 1e-6;
```

set and check solver

```
solver = 'gurobi';  % can be gurobi or tomlab_cplex
solverQuant = 'ibm_cplex';
outputPath = pwd;  % output is saved to this location, can be the same as pathToCOBRA 'C: ...
```

set and check solver

```
solverOK = changeCobraSolver(solverQuant, 'LP');
```

```
 > IBM ILOG CPLEX interface added to MATLAB path.
```

```
if solverOK == 1
    display('The solverQuant is set.');
else
    error('The solverQuant is not set.')
end
```

```
The solverQuant is set.
```

```

solverOK = changeCobraSolver(solver, 'LP');
```

```
 > Gurobi interface added to MATLAB path.
```

```
if solverOK == 1
    display('The LP solver is set.');
else
    error('The LP solver is not set.')
```

```
end
```

```
The LP solver is set.
```

```

solverOK = changeCobraSolver(solver, 'QP');
```

```
 > Gurobi interface added to MATLAB path.
```

```
if solverOK == 1
    display('The QP solver is set.');
else
    error('The QP solver is not set.')
end
```

```
The QP solver is set.
```

load and check tutorial input is loaded correctly

```
tutorialPath = [CBTDIR filesep 'tutorials' filesep 'metabotools' filesep 'tutorial_II'];
if exist([tutorialPath filesep 'starting_model.mat'], 'file') == 2  % 2 means it's a file.
    load([tutorialPath filesep 'starting_model.mat']);
    display('The model is loaded.');
else
    error('The ''starting_model'' could not be loaded.');
end
```

```
The model is loaded.
```

```

% Check output path and writing permission
if ~exist(outputPath) == 7
    error('Output directory in ''outputPath'' does not exist. Verify that you type it correctl
end

% make and save dummy file to test writing to output directory
A = rand(1);
try
    save([outputPath filesep 'A']);
catch ME
    error('Files cannot be saved to the provided location: %s\nObtain rights to write into %s
end
```

Section 1 - Define the model bounds using setMediumConstraints

```
set_inf = 2000;
current_inf = 1000;
medium_composition = {};
met_Conc_mM = [];
cellConc = [];
t = [];
cellWeight = [];

mediumCompounds = {'EX_h(e)', 'EX_h2o(e)', 'EX_hco3(e)', 'EX_nh4(e)', 'EX_o2(e)', 'EX_pi(e)',
ions = {'EX_ca2(e)', 'EX_cl(e)', 'EX_co(e)', 'EX_fe2(e)', 'EX_fe3(e)', 'EX_k(e)', 'EX_na1(e)',
mediumCompounds = [ions mediumCompounds];
mediumCompounds_lb = -100;

customizedConstraints = {'EX_co2(e)', 'EX_o2(e)', 'EX_his_L(e)', 'EX_ile_L(e)', 'EX_leu_L(e)',
                         'EX_lys_L(e)', 'EX_phe_L(e)', 'EX_thr_L(e)', 'EX_trp_L(e)', 'EX_val_L
                         'EX_met_L(e)', 'EX_ascb_L(e)', 'EX_btn(e)', 'EX_chol(e)', 'EX_fol(e)'
                         'EX_pnto_R(e)', 'EX_retn(e)', 'EX_thm(e)', 'EX_vitd2(e)', 'EX_vitd3(e
customizedConstraints_ub = [2000, 0, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 200
                            2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000];
customizedConstraints_lb = [-100, -1000, -10, -10, -10, -10, -10, -10, -10, -10, -10, -1, -1,
                            -1, -1, -1, -1, -1, -1, -1];

close_exchanges = 0;

[modelMedium, basisMedium] = setMediumConstraints(starting_model, set_inf, current_inf, medium
                                                  cellConc, t, cellWeight, mediumCompounds, me
                                                  customizedConstraints_ub, customizedConstrai

clearvars -EXCEPT modelMedium tol solver outputPath tutorialPath solverQuant
```

Section 2 - Generate an individual exchange profiles for each sample

```
load([tutorialPath filesep 'tutorial_II_data.mat']);
model = modelMedium;
test_max = 500;
test_min = 0.0001;
variation = 20;

prepIntegrationQuant(model, metData, exchanges, samples, test_max, test_min, outputPath, tol,

clearvars -EXCEPT modelMedium samples tol solver outputPath tutorialPath solverQuant
```

Section 2B - Prepare table to check exchange profiles

```
nmets = 70;
[mapped_exchanges, minMax, mapped_uptake, mapped_secretion] = checkExchangeProfiles(samples, o

clearvars -EXCEPT modelMedium samples tol solver mapped_exchanges  outputPath tutorialPath sol

save([outputPath 'Result_checkExchangeProfiles']);
```

Section 3 - Generate contextualized models

```
changeCobraSolver(solverQuant, 'LP');
```

```
> IBM ILOG CPLEX interface added to MATLAB path.
```

```

minGrowth = 0.008;
obj = 'biomass_reaction2';
no_secretion = {'EX_o2(e)'};
no_uptake = {'EX_o2s(e)', 'EX_h2o2(e)'};
medium = {};
tol = 1e-6;
```

```
model = modelMedium;
epsilon = 1e-4;

addExtraExch = {'EX_tdchola(e)', 'Ex_5hoxindoa[e]'};
addExtraExch_value = 1;
[ResultsAllCellLines, OverViewResults] = setQuantConstraints(model, samples, tol, minGrowth, o
                                                                    no_uptake, medium, addExtraExch,
```

```
j = 1
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
Starting parallel pool (parpool) using the 'local' profile ... connected to 12 workers.
LO = 446
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 13
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 5
  3935 Total reactions
  1231 Reversible reactions.
  2704 Irreversible reactions.
  2161 Flux consistent reactions, without flipping.
  1270 Flux inconsistent irreversible reactions, without flipping.
   504 Flux inconsistent reactions, without flipping.
  2176 Flux consistent reactions.
   489 Flux inconsistent reversible reactions left to flip.
j = 2
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 445
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 11
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 8
  3935 Total reactions
  1231 Reversible reactions.
  2704 Irreversible reactions.
  2156 Flux consistent reactions, without flipping.
  1272 Flux inconsistent irreversible reactions, without flipping.
   507 Flux inconsistent reactions, without flipping.
  2174 Flux consistent reactions.
   489 Flux inconsistent reversible reactions left to flip.
j = 3
solution =
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 447
solution =
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 16
solution =
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 1
  3935 Total reactions
  1231 Reversible reactions.
  2704 Irreversible reactions.
  2164 Flux consistent reactions, without flipping.
  1267 Flux inconsistent irreversible reactions, without flipping.
   504 Flux inconsistent reactions, without flipping.
  2179 Flux consistent reactions.
   489 Flux inconsistent reversible reactions left to flip.
j = 4
solution =
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 446
solution =
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 17
solution =
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 1
  3935 Total reactions
  1231 Reversible reactions.
  2704 Irreversible reactions.
  2159 Flux consistent reactions, without flipping.
  1267 Flux inconsistent irreversible reactions, without flipping.
```

```
   509 Flux inconsistent reactions, without flipping.
  2180 Flux consistent reactions.
   488 Flux inconsistent reversible reactions left to flip.
j = 5
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 444
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 9
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 0
  3935 Total reactions
  1231 Reversible reactions.
  2704 Irreversible reactions.
  2152 Flux consistent reactions, without flipping.
  1272 Flux inconsistent irreversible reactions, without flipping.
   511 Flux inconsistent reactions, without flipping.
  2169 Flux consistent reactions.
   494 Flux inconsistent reversible reactions left to flip.
j = 6
secretion_rxns = 
    'EX_5hoxindoa[e]'

secretion_rxns = 
    'EX_glyc3p[e]'

solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 445
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 10
solution = 
    obj: 0.0080

This objective corresponds to a flux with minimum Euclidean norm.
The largest weighting for minimising the norm was 1.
Check that the objective is the same without minimising the norm.
LO = 0
  3935 Total reactions
  1231 Reversible reactions.
  2704 Irreversible reactions.
  2153 Flux consistent reactions, without flipping.
  1281 Flux inconsistent irreversible reactions, without flipping.
   501 Flux inconsistent reactions, without flipping.
  2169 Flux consistent reactions.
   485 Flux inconsistent reversible reactions left to flip.
```

```
clearvars -EXCEPT modelMedium samples ResultsAllCellLines OverViewResults tol solver mapped_ex
```

Section 4 - Analyze added exchanges

```
changeCobraSolver(solver, 'LP');
```

```
 > Gurobi interface added to MATLAB path.
```

```

[Ex_added_all_unique] = statisticsAddedExchanges(ResultsAllCellLines, samples);
```

```
i = 1
i = 2
i = 3
i = 4
i = 5
i = 6
```

```
clearvars -EXCEPT modelMedium samples ResultsAllCellLines OverViewResults Ex_added_all_unique


[Added_all] = mkTableOfAddedExchanges(ResultsAllCellLines, samples, Ex_added_all_unique);

save([outputPath filesep 'statistics']);

clearvars -EXCEPT modelMedium samples ResultsAllCellLines OverViewResults tol solver mapped_ex
```

Section 5 - Analyze the sets of essential genes

```
cutoff = 0.05;

[genes, ResultsAllCellLines, OverViewResults] = analyzeSingleGeneDeletion(ResultsAllCellLines,
```

```
grRateWT = 13.5607
Single gene deletion analysis in progress ...
     ⋮
grRateWT = 13.6830
Single gene deletion analysis in progress ...
     ⋮
grRateWT = 15.4836
Single gene deletion analysis in progress ...
     ⋮
grRateWT = 15.3528
Single gene deletion analysis in progress ...
     ⋮
grRateWT = 0.6278
```

```
Single gene deletion analysis in progress ...
     :
grRateWT = 0.4804
Single gene deletion analysis in progress ...
     :
```

```
clearvars -EXCEPT modelMedium samples ResultsAllCellLines OverViewResults Ex_added_all_unique
```

Section 6 - Check which individual gene-associated reaction makes the model infeasible

```
samples_to_test = samples;
fill = 'NAN';
genes_to_test = {'55293.1'};

[FBA_Rxns_KO, ListResults] = checkEffectRxnKO(samples_to_test, fill, genes_to_test, samples, R
```

```
Warning: 3rd argument is numericFlag, currently redundant, will be depreciated
```

```
clearvars -EXCEPT modelMedium samples ResultsAllCellLines OverViewResults Ex_added_all_unique
```

Section 7 - Make intersect and union model

```
mk_union = 1;
mk_intersect = 1;
mk_reactionDiff = 1;
load([tutorialPath filesep 'starting_model.mat']);
model = starting_model;

[unionModel, intersectModel, diffRxns, diffExRxns] = makeSummaryModels(ResultsAllCellLines, sa
clearvars -EXCEPT modelMedium samples ResultsAllCellLines OverViewResults Ex_added_all_unique

save([outputPath filesep 'summary']);
```

Section 8 - Predict differences in metabolite production or consumption

Section 8A ATP production

```
obj = 'DM_atp_c_';
carbon_source = {'EX_glc(e)'};
samples = samples(1:4, 1);
dir = 1;

% ATP production
% exclude transport reactions from flux split analysis
transportRxns = {'ATPtm'; 'ATPtn'; 'ATPtx'; 'ATP1ter'; 'ATP2ter'; 'EX_atp(e)'; 'DNDPt13m';...
                 'DNDPt2m'; 'DNDPt31m'; 'DNDPt56m'; 'DNDPt32m'; 'DNDPt57m'; 'DNDPt20m';...
                 'DNDPt44m'; 'DNDPt19m'; 'DNDPt43m'; 'ADK1'; 'ADK1m'};

ATPprod = {'ATPS4m'; 'PGK'; 'PYK'; 'SUCOASm'};
```

```
met2test = {'atp[c]', 'atp[m]', 'atp[n]', 'atp[r]', 'atp[x]'};
[BMall, ResultsAllCellLines, metRsall, maximum_contributing_rxn, maximum_contributing_flux, AT


PHs = [samples maximum_contributing_rxn(:, 1)];

maximum_contributing_flux_ATP = maximum_contributing_flux;

clear ATPprod transportRxns met2test maximum_contributing_rxn
```

Section 8B NADH production

```
met2test = {'nadh[c]', 'nadh[m]', 'nadh[n]', 'nadh[x]', 'nadh[r]'};

transportRxns = {'NADHtpu'; 'NADHtru'; 'NADtpu'};

[BMall, ResultsAllCellLines, metRsall, maximum_contributing_rxn, maximum_contributing_flux_NAD
PHs = [PHs maximum_contributing_rxn(:, 1)];

clear transportRxns met2test maximum_contributing_rxn
```

Section 8C FADH2 production

```
transportRxns = {'FADH2tru'; 'FADH2tx'};

met2test = {'fadh2[c]', 'fadh2[m]', 'fadh2[n]', 'fadh2[x]', 'fadh2[r]'};
[BMall, ResultsAllCellLines, metRsall, maximum_contributing_rxn, maximum_contributing_flux_FAD

clear transportRxns met2test

PHs = [PHs maximum_contributing_rxn(:, 1)];
```

Section 8D NADPH production

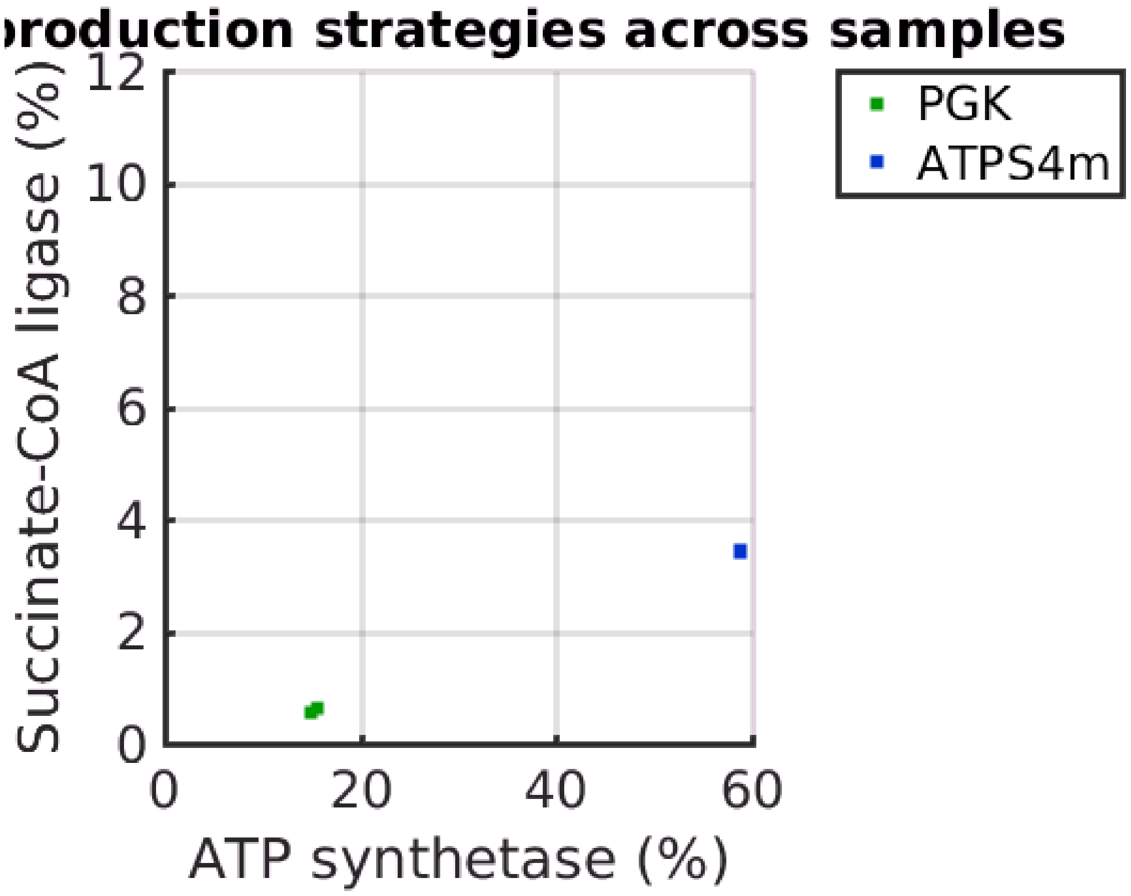
```
transportRxns = {'NADPHtru'; 'NADPHtxu'};

met2test = {'nadph[c]', 'nadph[m]', 'nadph[n]', 'nadph[x]', 'nadph[r]'};
[BMall, ResultsAllCellLines, metRsall, maximum_contributing_rxn, maximum_contributing_flux_NAD
clear transportRxns met2test

PHs = [PHs maximum_contributing_rxn(:, 1)];

save([outputPath filesep 'fluxSplits']);
```

Section 8E illustrate the phenotypes (PHs) on 3Dplot

```
diff_view = 1;
fonts = 18;

make3Dplot(PHs, maximum_contributing_flux_ATP, fonts, outputPath, diff_view);
```

```
x1 = 2x1 double
```

```
   20.5922
   20.4667
y1 = 2x1 double

   58.8195
   58.9126
z1 = 2x1 double

    3.4410
    3.4028
x2 = 2x1 double

   81.3017
   80.6285
y2 = 2x1 double

   15.0569
   15.5873
z2 = 2x1 double

    0.5454
    0.6235
```



Section 9 Perform phase Plane Analysis

```
mets = {'EX_glc(e)', 'EX_o2(e)'; 'EX_gln_L(e)', 'EX_o2(e)'; 'EX_lac_L(e)', 'EX_o2(e)'};
step_size = [40, 40; 20, 40; 40, 40];
step_num = [28, 26; 21, 26; 42, 26];
direct = [-1, -1; -1, -1; 1, -1];
```

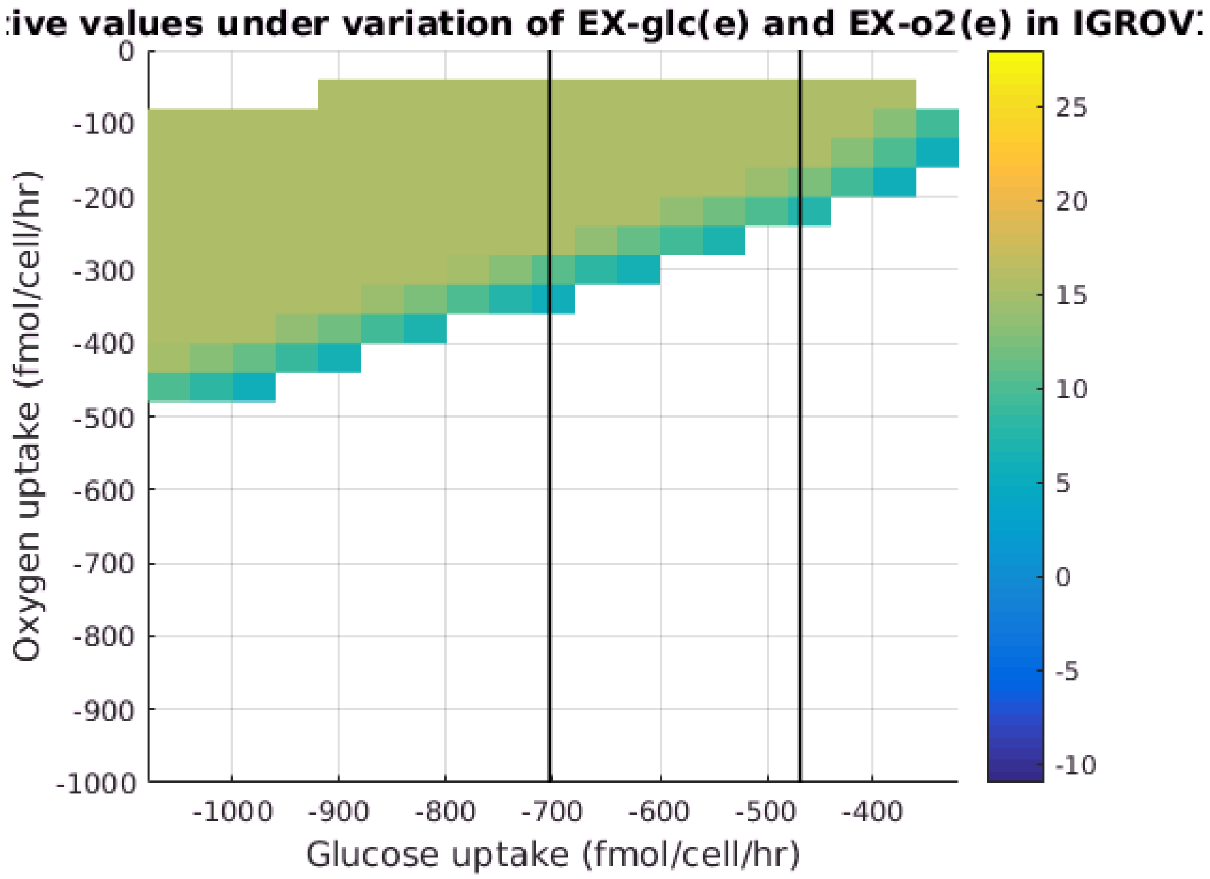
```
[ResultsAllCellLines] = performPPP(ResultsAllCellLines, mets, step_size, samples, step_num, di
```

```
k = 1
k = 2
k = 3
k = 4
```

```
save([outputPath filesep 'PPP']);
```

Section 9b illustrate phase plane analysis results

```
label = {'Glucose uptake (fmol/cell/hr)'; 'Oxygen uptake (fmol/cell/hr)'; 'Growth rate (hr-1)'
mets = {'EX_glc(e)'; 'EX_o2(e)'};
fonts = 12;
samples = {'IGROV1'};
illustrate_ppp(ResultsAllCellLines, mets, outputPath, samples, label, fonts, tol);
```


ive values under variation of EX-glc(e) and EX-o2(e) in IGROV
Oxygen uptake (fmol/cell/hr)
Glucose uptake (fmol/cell/hr)