Flux Variability analysis (FVA)

Flux variability analysis (FVA) is a widely used computational tool for evaluating the minimum and maximum range of each reaction flux that can still satisfy the constraints using two optimisation problems for each reaction of interest[1].

$$\begin{array}{lll} \max\limits_{v} / \min\limits_{v} & & v_j \\ \text{s.t.} & & Sv = 0, \\ & & l \le v \le u, \\ & & c_T v = c_T v^* \end{array}$$

where $v \in R^n$ represents the rate of each biochemical reaction, but typically an infinite set of steady state flux vectors exist can satisfy the same requirement for an optimal objective $c_T v^* = c_T v$. As well as for the flux balance analysis (FBA), there are also many possible variations on flux variability analysis (FVA)[2].

Depending on the size of the model you are using for the analysis, use:

- `fluxVariability()` function - for the low dimensional FVA;
- `fastFVA()` function - for the models with more than 1,000 reactions;
- distributedFBA.jl - for high dimensional FVA, models larger than 10,000 reactions[2];

EQUIPMENT SETUP

If necessary, initialize the cobra toolbox with

```
% initCobraToolbox
warning('off', 'MATLAB:subscripting:noSubscriptsSpecified');
```

For solving linear programming problems in FBA and FVA analysis, certain solvers are required:

```
% solverOK = changeCobraSolver(solverName, solverType)
```

The present tutorial can run with glpk package, which does not require additional installation and configuration. Although, for the analysis of large models is recommended to use the GUROBI package.

```
changeCobraSolver ('gurobi', 'all');
```

```
> Gurobi interface added to MATLAB path.
> Solver for LP problems has been set to gurobi.

> Gurobi interface added to MATLAB path.
> Solver for MILP problems has been set to gurobi.

> Gurobi interface added to MATLAB path.
> Solver for QP problems has been set to gurobi.

> Gurobi interface added to MATLAB path.
```

```
> Solver for MIQP problems has been set to gurobi.
> Solver gurobi not supported for problems of type NLP. Currently used: matlab
```

PROCEDURE

In this tutorial, the provided model is a generic model of the human cellular metabolism, Recon 3D[3] or Recon2.0. Therefore, we assume, that the cellular objectives include energy production or optimisation of uptake rates and by-product secretion for various physiological functions of the human body.

Before proceeding with the simulations, the path for the model needs to be set up:

```
% check if Recon3 exists:
% pathModel = '~/work/sbgCloud/data/models/unpublished/Recon3D_models/';
% filename = '2017_04_28_Recon3d.mat';
% load([pathModel, filename])
% model = modelRecon3model;
% clear modelRecon3model
% and if not
% select your own model, or use Recon2.0model instead filename='Recon3.0model';
global CBTDIR
load([CBTDIR filesep 'test' filesep 'models' filesep 'Recon2.0model.mat']);
model = Recon2model;
model.rxns = strrep(model.rxns, '(', '[');
model.rxns = strrep(model.rxns, ')', ']');
clear Recon2model
```

The metabolites structures and reactions are from the Virtual Metabolic Human database (VMH, http://vmh.life).

TROUBLESHOOTING

If there are multiple energy sources available in the model, specify more constraints.

If we do not do that, we will have additional carbon and oxygen energy sources available in the cell and the maximal ATP production.

To avoid this issue, all external carbon sources need to be closed.

```
% Closing the uptake of all energy and oxygen sources
idx = strmatch('Exchange/demand reaction', model.subSystems);
c = 0;
for i = 1:length(idx)
    if model.lb(idx(i)) ~= 0
        c = c + 1;
        uptakes{c} = model.rxns{idx(i)};
    end
end
% If you use Recon3.0 model, than:
% modelalter = model;
% modelalter = changeRxnBounds(modelalter, uptakes, 0, 'b');
% modelalter = changeRxnBounds(modelalter, 'EX_HC00250[e]', -1000, 'l');

% The alternative way to do that, in case you were using another large model,
% that does not contain defined Subsystem is
% to find uptake exchange reactions with following codes:
% [selExc, selUpt] = findExcRxns(model);
% uptakes = model.rxns(selUpt);
```

```
% Selecting from the exchange uptake reactions those
% which contain at least 1 carbon in the metabolites included in the reaction:
 subuptakeModel = extractSubNetwork(model, uptakes);
 hiCarbonRxns = findCarbonRxns(subuptakeModel,1);
% Closing the uptake of all the carbon sources
 modelalter = model;
 modelalter = changeRxnBounds(modelalter, hiCarbonRxns, 0, 'b');
% Closing other oxygen and energy sources
 exoxygen = {'EX_adp'
    'EX_amp[e]'
    'EX_atp[e]'
    'EX_co2[e]'
    'EX_coa[e]'
    'EX_fad[e]'
    'EX_fe2[e]'
    'EX_fe3[e]'
    'EX_gdp[e]'
    'EX_gmp[e]'
    'EX_gtp[e]'
    'EX_h[e]'
    'EX_h2o[e]'
    'EX_h2o2[e]'
    'EX_nad[e]'
    'EX_nadp[e]'
    'EX_no[e]'
    'EX_no2[e]'
    'EX_o2s[e]'};
modelalter = changeRxnBounds (modelalter, exoxygen, 0, 'l');
```

In this example, we are analysing the variability of several reactions from the human cellular metabolism in the aerobic and anaerobic state.

For each simulation, the original model will be copied to a new variable. This preserves the constraints of the original model and allows to perform simulations with new constraints. Additionally, this method of renaming the model avoids confusion while performing multiple simulations at the same time.

```
% modelfva1 represents aerobic condition
modelfva1 = modelalter;
% For Recon3.0 model
% modelfva1 = changeRxnBounds (modelfva1, 'EX_glc_D[e]', -20, 'l');
modelfva1 = changeRxnBounds(modelfva1, 'EX_glc[e]', -20, 'l');
modelfva1 = changeRxnBounds(modelfva1, 'EX_o2[e]', -1000, 'l');
% modelfva2 represents anaerobic condition
modelfva2 = modelfva1;
modelfva2 = changeRxnBounds(modelfva2, 'EX_o2[e]',  0, 'l');
```

Standard FVA

The full spectrum of flux variability analysis options can be accessed using the command:

```
% [minFlux, maxFlux, Vmin, Vmax] = fluxVariability(model, optPercentage,...
%     osenseStr, rxnNameList, verbFlag, allowLoops, method);
```

The `optPercentage` parameter allows one to choose whether to consider solutions that give at least a certain percentage of the optimal solution.

Setting the parameters `osenseStr = 'min'` or `osenseStr = 'max'` determines whether the flux balance analysis problem is first solved with minimization or maximisation.

The `rxnNameList` accepts a cell array list of reactions to selectively perform flux variability upon. This is useful for high-dimensional models where computation of a flux variability for all reactions is more time consuming:

```
% Selecting several reactions from the model that we want to analyse with FVA
rxnsList = {'DM_atp_c_'
    'ACOAHi'
    'ALCD21_D'
    'LALDO'
    'ME2m'
    'AKGDm'
    'PGI'
    'PGM'
    'r0062'};
```

The `verbFlag` input determines how much output to print.

`allowLoops==0` invokes a mixed integer linear programming implementation of thermodynamically constrained flux variability analysis for each minimization or maximisation of a reaction rate.

The `method` parameter input determines whether are the output flux vectors also minimise the `0-norm`, `1-norm` or `2-norm` whilst maximising or minimising the flux through one reaction.

Running `fluxVariability()` on both models (`modelfva1`, `modelfva2`) will generate the minimum and maximum flux ranges of selected reactions, from rxnsList, in the network.

Run FVA analysis for the model with the constraints that simulates aerobic conditions:

```
[minFlux1, maxFlux1, Vmin1, Vmax1] = fluxVariability(modelfva1, [], [], rxnsList)
```

```
CPXPARAM_QPMethod                                1
CPXPARAM_QPMethod                                1
CPXPARAM_Read_APIEncoding                        "*"
CPXPARAM_Output_CloneLog                         1
Tried aggregator 1 time.
QP Presolve eliminated 2892 rows and 3361 columns.
Reduced QP has 2172 rows, 4079 columns, and 17776 nonzeros.
Reduced QP objective Q matrix has 4079 nonzeros.
Presolve time = 0.02 sec. (4.41 ticks)

Using LP solver to compute a starting basis.

Iteration log . . .
Iteration:     1    Scaled infeas =        809006.348953
Iteration:   235    Scaled infeas =        254528.194928
Iteration:   358    Scaled infeas =        190568.748674
Iteration:   474    Scaled infeas =        156230.914947
Iteration:   583    Scaled infeas =        125053.416035
Iteration:   702    Scaled infeas =        106759.924122
Iteration:   808    Scaled infeas =         94441.176259
Iteration:   906    Scaled infeas =         83251.115047
Iteration:  1016    Scaled infeas =         71134.562056
Iteration:  1125    Scaled infeas =         60839.020346
Iteration:  1234    Scaled infeas =         54158.413809
Iteration:  1349    Scaled infeas =         45646.114719
Iteration:  1460    Scaled infeas =         34548.836668
Iteration:  1582    Scaled infeas =         26693.506750
```

```
Iteration:  1696    Scaled infeas =         19674.935362
Iteration:  1809    Scaled infeas =         17581.715845
Iteration:  1930    Scaled infeas =         11610.198404
Iteration:  2039    Scaled infeas =          9429.052929
Switched to devex.
Iteration:  2151    Scaled infeas =          7522.605903
Iteration:  2269    Scaled infeas =          5130.865665
Iteration:  2381    Scaled infeas =          4020.124498
Iteration:  2496    Scaled infeas =          2659.911564
Iteration:  2599    Scaled infeas =          1794.637821
Iteration:  2708    Scaled infeas =          1005.051615
CPXPARAM_Read_APIEncoding                        "*"
CPXPARAM_Output_CloneLog                         1
Tried aggregator 1 time.
QP Presolve eliminated 2892 rows and 3361 columns.
Reduced QP has 2172 rows, 4079 columns, and 17776 nonzeros.
Reduced QP objective Q matrix has 4079 nonzeros.
Presolve time = 0.02 sec. (4.41 ticks)

Using LP solver to compute a starting basis.

Iteration log . . .
Iteration:     1    Scaled infeas =        802405.239581
Iteration:   282    Scaled infeas =        156833.639986
Iteration:   457    Scaled infeas =        100519.803430
Iteration:   605    Scaled infeas =         83926.160690
Iteration:   744    Scaled infeas =         69816.485921
Iteration:   867    Scaled infeas =         55500.964495
Iteration:   991    Scaled infeas =         41537.977213
Iteration:  1112    Scaled infeas =         30808.845465
Iteration:  1234    Scaled infeas =         24678.227136
Iteration:  1367    Scaled infeas =         14974.665183
Iteration:  1499    Scaled infeas =          9344.215496
Iteration:  1614    Scaled infeas =          5934.754899
Iteration:  1731    Scaled infeas =          4008.302145
Iteration:  1868    Scaled infeas =          3273.618701
Iteration:  1990    Scaled infeas =          2089.243196
Switched to devex.
Iteration:  2153    Scaled infeas =          1669.457231
Iteration:  2283    Scaled infeas =           549.399429
Iteration:  2416    Scaled infeas =           182.630752
Iteration:  2526    Scaled infeas =           108.749837
Iteration:  2636    Scaled infeas =            47.367193
Iteration:  2764    Scaled infeas =             7.093565
Iteration:  2887    Scaled infeas =             1.740681

Iteration log . . .
Iteration:     1    Objective     =     387517414.656292
Iteration:   117    Objective     =     387517414.656292
Iteration:  2813    Scaled infeas =           334.105203
Iteration:  2941    Scaled infeas =           191.755419
Switched to steepest-edge.
Iteration:  3059    Scaled infeas =           117.659241
Iteration:  3156    Scaled infeas =            48.942211
Iteration:  3238    Scaled infeas =            28.760935
Iteration:  3342    Scaled infeas =             2.828016
Iteration:  3435    Scaled infeas =             0.001006

Iteration log . . .
Iteration:     1    Objective     =     380708315.270927
Iteration:   154    Objective     =     377838445.499805
Iteration:   255    Objective     =     347833952.262181
Iteration:   438    Objective     =     210049514.692313
Iteration:   224    Objective     =     387517414.656292
Iteration:   244    Objective     =     387517414.656292
Iteration:   344    Objective     =     385757278.247892
Iteration:   392    Objective     =     385757278.247892
Iteration:   428    Objective     =     382240654.295706
```

```
Iteration:   589    Objective     =     380716468.141767
Iteration:   611    Objective     =     380609788.131670
Iteration:   772    Objective     =     377008129.694392
Iteration:   920    Objective     =     376292552.800659
Iteration:  1079    Objective     =     369502332.041310
Iteration:   588    Objective     =     134823717.478052
Iteration:   611    Objective     =     126919040.022167
Iteration:   734    Objective     =      72295508.628015
Markowitz threshold set to 0.1
Iteration:   877    Objective     =      18981138.573552
Removing shift (41).
Iteration:   966    Phase I obj   =      18890059.907669
Iteration:   969    Objective     =     126525027.068087
Iteration:  1084    Objective     =      75538318.240691
Iteration:  1222    Objective     =     367293498.964907
Iteration:  1389    Objective     =     362773328.170021
Iteration:  1557    Objective     =     340922675.511429
Iteration:  1714    Objective     =     294051018.746023
Iteration:  1832    Objective     =     282125805.754080
Iteration:  1996    Objective     =     273752350.884846
Iteration:  2150    Objective     =     253469459.846934
Iteration:  1195    Objective     =      22126560.377221
Iteration:  1307    Objective     =      19693605.042544
Iteration:  1335    Objective     =      19693369.556368
Removing shift (7).
Scaled reduced cost of dropped variable 'x3953' = -1572.33
Attempting to reinclude dropped variables.
Iteration:  1338    Objective     =      19543343.478506
Removing shift (7).
Iteration:  1358    Phase I obj   =      18890063.315057
Markowitz threshold set to 0.6
Iteration:  2296    Objective     =     207659002.616543
Iteration:  2442    Objective     =     199710812.433243
Iteration:  2607    Objective     =     182940875.558113
Iteration:  2798    Objective     =     175104824.857662
Iteration:  2958    Objective     =     165833475.758821
Iteration:  3056    Objective     =     151527657.365892
Iteration:  3194    Objective     =     131024891.002498
CPXPARAM_QPMethod                                1
CPXPARAM_Read_APIEncoding                        "*"
CPXPARAM_Output_CloneLog                         1
Tried aggregator 1 time.
QP Presolve eliminated 2892 rows and 3361 columns.
Reduced QP has 2172 rows, 4079 columns, and 17776 nonzeros.
Reduced QP objective Q matrix has 4079 nonzeros.
Presolve time = 0.10 sec. (4.41 ticks)

Using LP solver to compute a starting basis.

Iteration log . . .
Iteration:     1    Scaled infeas =        809006.348953
Iteration:   235    Scaled infeas =        254528.194928
Iteration:   358    Scaled infeas =        190568.748674
Iteration:   474    Scaled infeas =        156230.914947
Iteration:   583    Scaled infeas =        125053.416035
Iteration:   702    Scaled infeas =        106759.924122
Iteration:   808    Scaled infeas =         94441.176259
Iteration:   906    Scaled infeas =         83251.115047
Iteration:  1016    Scaled infeas =         71134.562056
Iteration:  1125    Scaled infeas =         60839.020346
Iteration:  3229    Objective     =     130481039.172317
Iteration:  3254    Objective     =     129954782.647019
Iteration:  3406    Objective     =     128001708.161898
Iteration:  3537    Objective     =      93428453.665306
Iteration:  3663    Objective     =      45342818.338836
Iteration:  3674    Objective     =      39840967.415091
Iteration:  3895    Objective     =      35440481.474279
Iteration:  4110    Objective     =      20911807.740116
```

```
Iteration:  1234    Scaled infeas =         54158.413809
Iteration:  1349    Scaled infeas =         45646.114719
Iteration:  1460    Scaled infeas =         34548.836668
Iteration:  1582    Scaled infeas =         26693.506750
Iteration:  1696    Scaled infeas =         19674.935362
Iteration:  1809    Scaled infeas =         17581.715845
Iteration:  1930    Scaled infeas =         11610.198404
Iteration:  2039    Scaled infeas =          9429.052929
Switched to devex.
Iteration:  2151    Scaled infeas =          7522.605903
Iteration:  2269    Scaled infeas =          5130.865665
Iteration:  2381    Scaled infeas =          4020.124498
Iteration:  2496    Scaled infeas =          2659.911564
Iteration:  2599    Scaled infeas =          1794.637821
Iteration:  2708    Scaled infeas =          1005.051615
Iteration:  4254    Objective     =        777006.212855
Iteration:  4284    Objective     =        777005.300826
Iteration:  4303    Objective     =        776986.858524
Iteration:  4430    Objective     =        767315.054346
Iteration:  4570    Objective     =        767313.657904
Iteration:  2813    Scaled infeas =           334.105203
Iteration:  2941    Scaled infeas =           191.755419
Switched to steepest-edge.
Iteration:  3059    Scaled infeas =           117.659241
Iteration:  3156    Scaled infeas =            48.942211
Iteration:  3238    Scaled infeas =            28.760935
Iteration:  3342    Scaled infeas =             2.828016
Iteration:  3435    Scaled infeas =             0.001006

Iteration log . . .
Iteration:     1    Objective     =     380708315.270927
Iteration:   154    Objective     =     377838445.499805
Iteration:   255    Objective     =     347833952.262181
Iteration:   438    Objective     =     210049514.692313
Iteration:   588    Objective     =     134823717.478052
```

```
minFlux1 =

   1.0e+03
         0
         0
         0
   -1.0000
         0
         0
   -1.0000
   -0.0682
         0
maxFlux1 =

   1.0e+03
    1.0000
    1.0000
    1.0000
    1.0000
    1.0000
    0.6571
    0.0200
    1.0000
    1.0000
Vmin1 =

         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
```

```
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
   -9.0328   -0.0000   -0.3575   -1.8017   -9.2028    0.0000   -0.1666  -20.2576
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0

Vmax1 =

         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
   -9.0328   -0.0000   -0.3575   -1.8017   -9.2028    0.0000   -0.1666  -20.2576
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
```

Run FVA analysis for the model with the constraints that simulates anaerobic conditions:

```
[minFlux2, maxFlux2, Vmin2, Vmax2] = fluxVariability(modelfva2, [], [], rxnsList)
```

```
minFlux2 =

   1.0e+03
         0
         0
         0
   -1.0000
         0
         0
   -0.2644
   -0.0402
         0
maxFlux2 =

   1.0e+03
    0.0826
    0.1652
    1.0000
    1.0000
    1.0000
    0.0280
    0.0200
    0.0542
    0.1652
Vmin2 =

         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
  -19.3226  -26.1787   -1.7640  -13.9937  -11.0973   -2.5152   -2.4127  -19.5854
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0

Vmax2 =

         0         0         0         0         0         0         0         0
```

```
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
  -19.3226   -26.1787    -1.7640   -13.9937   -11.0973    -2.5152    -2.4127   -19.5854
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
```

The additional $n \times k$ output matrices `Vmin` and `Vmax` return the flux vector for each of the $k \leq n$ fluxes selected for flux variability.

Further, plot and compare the FVA results from the both models:

```
ymax1 = maxFlux1;
ymin1 = minFlux1;
ymax2 = maxFlux2;
ymin2 = minFlux2;

maxf = table(ymax1, ymax2)
```

```
maxf =
    ymax1     ymax2
    ______    ______

      1000    82.618
      1000    165.24
      1000      1000
      1000      1000
      1000      1000
    657.07        28
        20        20
      1000    54.206
      1000    165.24
```

```
minf = table(ymin1, ymin2)
```

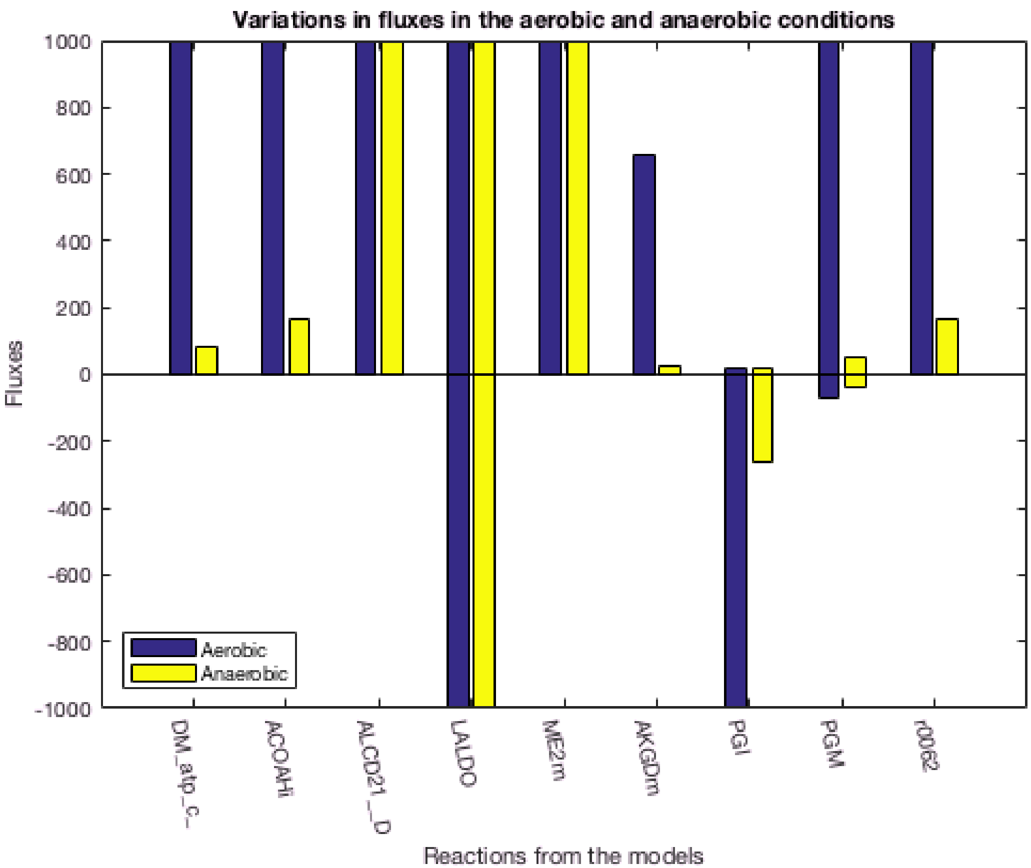
```
minf =
     ymin1      ymin2
    _______    _______

          0          0
          0          0
          0          0
      -1000      -1000
          0          0
          0          0
      -1000    -264.38
    -68.167     -40.21
          0          0
```

```
maxfxs = table2cell(maxf);
minfxs = table2cell(minf);

figure
plot1 = bar(cell2mat(maxfxs(1:end, :)));
hold on
```

```
plot2 = bar(cell2mat(minfxs(1:end, :)));
hold off
xticklabels({'DM_atp_c_', 'ACOAHi', 'ALCD21__D', 'LALDO',...
             'ME2m', 'AKGDm', 'PGI', 'PGM', 'r0062'})
set(gca, 'XTickLabelRotation', -80);
yticks([-1000 -800 -600 -400 -200 0 200 400 600 800 1000])
xlabel('Reactions from the models')
ylabel('Fluxes')
legend({'Aerobic', 'Anaerobic'}, 'Location', 'southwest')
title('Variations in fluxes in the aerobic and anaerobic conditions')
```



Fast FVA

The code is as follows-

```
% [minFlux, maxFlux, optsol, ret, fbasol, fvamin, fvamax, statussolmin,...
% statussolmax] = fastFVA(model, optPercentage, objective, solverName,...
% rxnsList, matrixAS, cpxControl, strategy, rxnsOptMode)
```

The `fastFVA()` function returns vectors for the initial FBA in `fbasol` together with matrices `fvamin` and `fvamax` containing the flux values for each individual min/max problem.

TROUBLESHOOTING

Note that for large models the memory requirements may become prohibitive.

The `fastFVA()` function only supports the CPLX solver. For detail information, refer to the solver installation guide.

```
changeCobraSolver ('ibm_cplex', 'all', 1);
```

```
> IBM ILOG CPLEX interface added to MATLAB path.
> Solver for LP problems has been set to ibm_cplex.

> IBM ILOG CPLEX interface added to MATLAB path.
> Solver for MILP problems has been set to ibm_cplex.

> IBM ILOG CPLEX interface added to MATLAB path.
> Solver for QP problems has been set to ibm_cplex.
> Solver ibm_cplex not supported for problems of type MIQP. Currently used: gurobi
> Solver ibm_cplex not supported for problems of type NLP. Currently used: matlab
```

Run fast FVA analysis for the whole model with the constraints that simulates aerobic conditions:

```
[minFluxF1, maxFluxF1, optsol, ret, fbasol, fvamin, fvamax,...
    statussolmin, statussolmax] = fastFVA(modelfva1);
```

```
> The CPLEX version has been determined as 1271.
>> Solving Model.S. (uncoupled)
>> The number of arguments is: input: 1, output 9.
>> Size of stoichiometric matrix: (5063,7440)
>> All reactions are solved (7440 reactions - 100%).
>> 0 reactions out of 7440 are minimized (0.00%).
>> 0 reactions out of 7440 are maximized (0.00%).
>> 7440 reactions out of 7440 are minimized and maximized (100.00%).

-- Starting to loop through the 2 workers. --

-- The splitting strategy is 0. --

--------------------------------------------------------------------------------
--  Task Launched // TaskID: 1 / 2 (LoopID = 2) <> [3721, 7440] / [5063, 7440].
>> The number of reactions retrieved is 3720
>> Log files will be stored at /Users/syarra/Dropbox/uni.lu/github/opencobra/cobratoolbox/src/analysis/
-- Start time:     Thu Jul 13 10:58:50 2017
>> #Task.ID = 1; logfile: cplexint_logfile_1.log

-- Warning:: The optPercentage is higher than 90. The solution process might take longer than you expec

       -- Minimization (iRound = 0). Number of reactions: 3720.
       -- Maximization (iRound = 1). Number of reactions: 3720.
-- End time:     Thu Jul 13 11:09:04 2017
>> Time spent in FVAc: 613.6 seconds.
--------------------------------------------------------------------------------
==> 50.0% done. Please wait ...

--------------------------------------------------------------------------------
--  Task Launched // TaskID: 2 / 2 (LoopID = 1) <> [1, 3720] / [5063, 7440].
>> The number of reactions retrieved is 3720
>> Log files will be stored at /Users/syarra/Dropbox/uni.lu/github/opencobra/cobratoolbox/src/analysis/
-- Start time:     Thu Jul 13 10:58:50 2017
>> #Task.ID = 2; logfile: cplexint_logfile_2.log

-- Warning:: The optPercentage is higher than 90. The solution process might take longer than you expec

       -- Minimization (iRound = 0). Number of reactions: 3720.
       -- Maximization (iRound = 1). Number of reactions: 3720.
-- End time:     Thu Jul 13 11:10:09 2017
>> Time spent in FVAc: 678.5 seconds.
--------------------------------------------------------------------------------
==> 100% done. Analysis completed.
```

Run fast FVA analysis for the whole model with the constraints that simulates anaerobic conditions:

```
[minFluxF2, maxFluxF2, optsol2, ret2, fbasol2, fvamin2, fvamax2,...
    statussolmin2, statussolmax2] = fastFVA(modelfva2);
```

```
 > The CPLEX version has been determined as 1271.
 >> Solving Model.S. (uncoupled)
 >> The number of arguments is: input: 1, output 9.
 >> Size of stoichiometric matrix: (5063,7440)
 >> All reactions are solved (7440 reactions - 100%).
 >> 0 reactions out of 7440 are minimized (0.00%).
 >> 0 reactions out of 7440 are maximized (0.00%).
 >> 7440 reactions out of 7440 are minimized and maximized (100.00%).

 -- Starting to loop through the 2 workers. --

 -- The splitting strategy is 0. --

------------------------------------------------------------------------------------
--  Task Launched // TaskID: 1 / 2 (LoopID = 2) <> [3721, 7440] / [5063, 7440].
 >> The number of reactions retrieved is 3720
 >> Log files will be stored at /Users/syarra/Dropbox/uni.lu/github/opencobra/cobratoolbox/src/analysis/
 -- Start time:     Thu Jul 13 11:10:29 2017
 >> #Task.ID = 1; logfile: cplexint_logfile_1.log

 -- Warning:: The optPercentage is higher than 90. The solution process might take longer than you expec

        -- Minimization (iRound = 0). Number of reactions: 3720.
        -- Maximization (iRound = 1). Number of reactions: 3720.
 -- End time:     Thu Jul 13 11:20:08 2017
 >> Time spent in FVAc: 578.9 seconds.
------------------------------------------------------------------------------------
 ==> 50.0% done. Please wait ...

------------------------------------------------------------------------------------
--  Task Launched // TaskID: 2 / 2 (LoopID = 1) <> [1, 3720] / [5063, 7440].
 >> The number of reactions retrieved is 3720
 >> Log files will be stored at /Users/syarra/Dropbox/uni.lu/github/opencobra/cobratoolbox/src/analysis/
 -- Start time:     Thu Jul 13 11:10:28 2017
 >> #Task.ID = 2; logfile: cplexint_logfile_2.log

 -- Warning:: The optPercentage is higher than 90. The solution process might take longer than you expec

        -- Minimization (iRound = 0). Number of reactions: 3720.
        -- Maximization (iRound = 1). Number of reactions: 3720.
 -- End time:     Thu Jul 13 11:21:42 2017
 >> Time spent in FVAc: 673.3 seconds.
------------------------------------------------------------------------------------
 ==> 100% done. Analysis completed.
```

Plotting the results of the fast FVA and comparing them between the aerobic and anaerobic models:

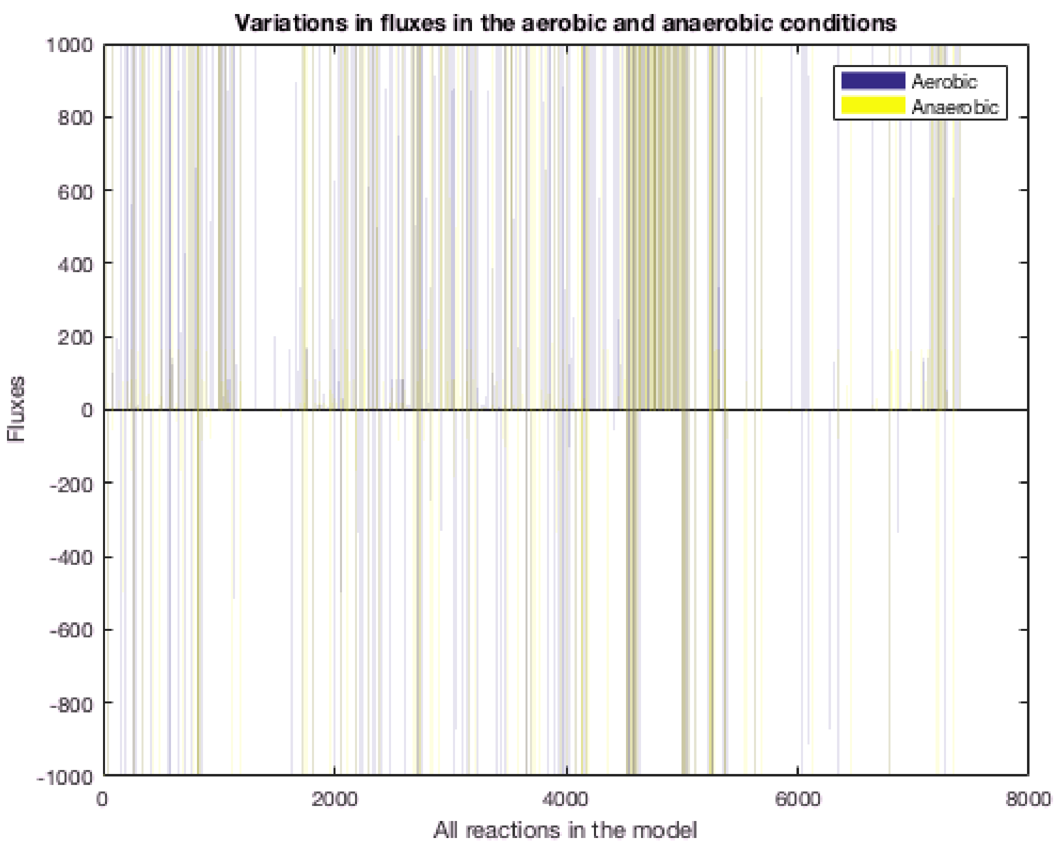
```
ymaxf1 = maxFluxF1;
yminf1 = minFluxF1;
ymaxf2 = maxFluxF2;
yminf2 = minFluxF2;

maxf =table(ymaxf1, ymaxf2);
minf =table(yminf1, yminf2);

maxf = table2cell(maxf);
```

```
minf = table2cell(minf);

figure
plot3 = bar(cell2mat(maxf(1:end, :)));
hold on
plot4 = bar(cell2mat(minf(1:end, :)));
hold off
xticks([0 2000 4000 6000 8000 10600])
yticks([-1000 -800 -600 -400 -200 0 200 400 600 800 1000])
xlabel('All reactions in the model')
ylabel('Fluxes')
legend({'Aerobic', 'Anaerobic'})
title('Variations in fluxes in the aerobic and anaerobic conditions')
```



REFERENCES

[1] Gudmundsson, S., Thiele, I. Computationally efficient flux variability analysis. *BMC Bioinformatics.* 11, 489 (2010).

[2] Heirendt, L., Thiele, I., Fleming, R.M. DistributedFBA.jl: high-level, high-performance flux balance analysis in Julia. *Bioinformatics.* 33 (9), 1421-1423 (2017).

[3] Thiele, I., Price, N.D., Vo, T.D., Palsson B. Ø. Candidate Metabolic Network States in Human Mitochondria. Impact of diabetes, ischemia and diet. *J Bio Chem.* 280 (12), 11683–11695 (2005).