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].

$$\begin{aligned} \max_{v} / \min_{v} \quad & v_j \\ \text{s.t.} \quad & Sv = 0, \\ & l \le v \le u, \\ & c_T v = c_T v * \end{aligned}$$

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:

```
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: /opt/gurobi702/linux64/matlab
  - [----] TOMLAB_PATH :  --> set this path manually after installing the solver ( see instructions )
  - [----] MOSEK_PATH :  --> set this path manually after installing the solver ( see instructions )
  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              0      0      0      -      -
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         -                 8      3      4      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' - 'pdco' - '
> You can solve MILP problems using: 'glpk' - 'gurobi' - 'ibm_cplex'
> You can solve QP problems using: 'gurobi' - 'ibm_cplex' - 'pdco' - 'qpng'
> You can solve MIQP problems using: 'gurobi'
> You can solve NLP problems using: 'matlab' - 'quadMinos'

> Checking for available updates ...
> The COBRA Toolbox is up-to-date.
```

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

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

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', 1);
```

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

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

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

> Gurobi interface added to MATLAB path.
> Solver for MIQPproblems 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]. 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:

```
pathModel = '~/work/sbgCloud/data/models/unpublished/Recon3D_models/';
filename = '2017_04_28_Recon3d.mat';
load([pathModel, filename])
model = modelRecon3model;
clear modelRecon3model
```

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

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, '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;
modelfva1 = changeRxnBounds (modelfva1, 'EX_glc_D[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'
    'HMR_2087'
    'LALDO'
    'LCADim'
    'ME2m'
    'AKGDm'
    'PGI'
    'PGM'
    'r0062'
    'HMR_2088'};
```

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)
```

```
Starting parallel pool (parpool) using the 'local' profile ... connected to 4 workers.
minFlux1 =

           0
           0
           0
       -1000
       -1000
           0
           0
           0
       -1000
       -1000

maxFlux1 =

   1.0e+03
    0.9960
    0.4980
    1.0000
    1.0000
    0.8983
    0.8543
    0.9554
    0.5460
    1.0000
    0.6669

Vmin1 =

         0         0         0   75.0906         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  -75.0906         0         0         0         0
 -131.2953  -98.0480    3.0043  -31.3200 -244.1706 -158.9278  -44.5171 -140.3465
         0         0         0         0         0         0         0         0
   -0.0000         0         0         0         0         0         0         0

Vmax1 =

         0         0         0   75.0906         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  -75.0906         0         0         0         0
 -131.2953  -98.0480    3.0043  -31.3200 -244.1706 -158.9278  -44.5171 -140.3465
         0         0         0         0         0         0         0         0
   -0.0000         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
   -0.5000
   -1.0000
         0
         0
         0
   -1.0000
   -0.3200

maxFlux2 =

   1.0e+03
    0.0400
    0.0200
    1.0000
    0.5800
    0.0200
    0.0200
    0.0400
    0.0308
    1.0000
         0

Vmin2 =

         0         0         0   13.3333         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  -13.3333         0         0         0         0
   -6.1524   -6.1773    3.0405   -6.1524   -5.7676   -2.5516   -3.0597  -24.7238
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0

Vmax2 =

         0         0         0   13.3333         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  -13.3333         0         0         0         0
   -6.1524   -6.1773    3.0405   -6.1524   -5.7676   -2.5516   -3.0597  -24.7238
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
```

The additional n × k output matrices `Vmin` and `Vmax` return the flux vector for each of the k ≤ 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
    ______    ______

    995.97        40
    497.98        20
      1000      1000
      1000       580
    898.31        20
    854.29        20
    955.36        40
    546.03    30.769
      1000      1000
    666.92         0
    497.98        20
      1000       580
```

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

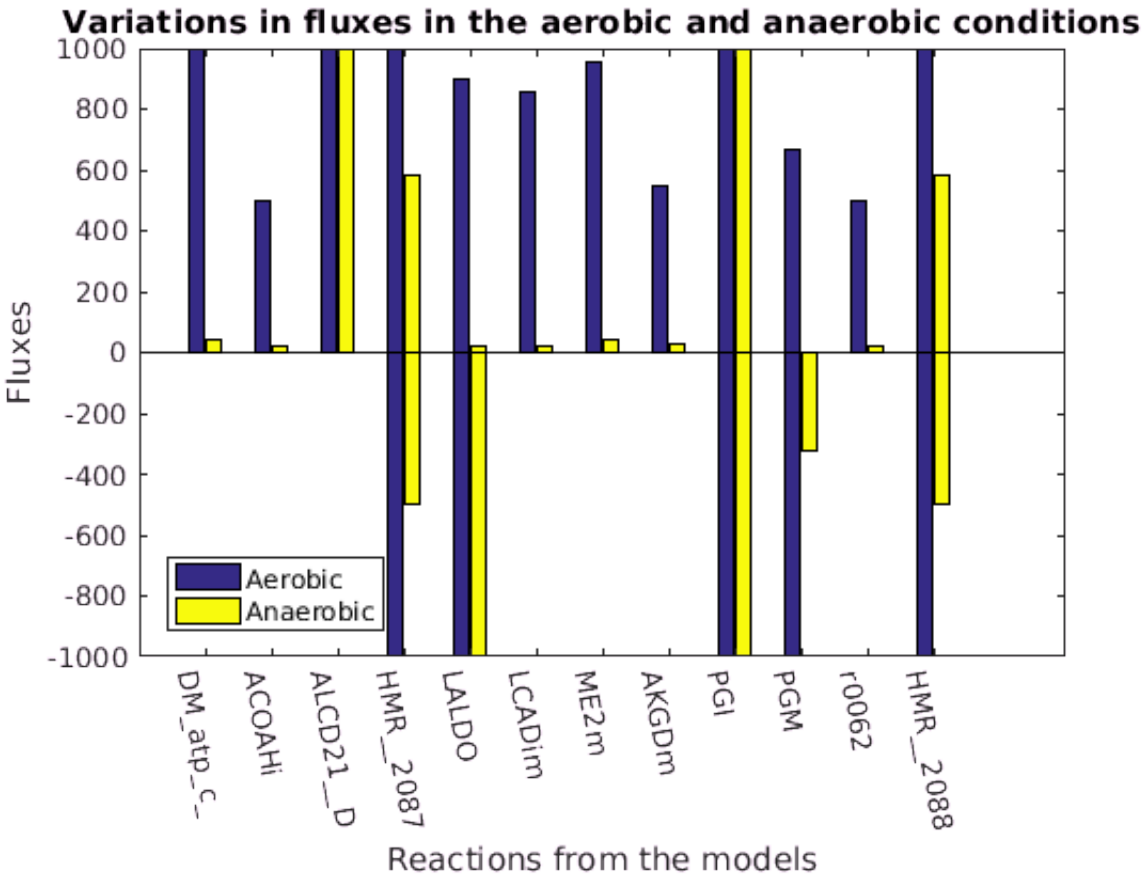
```
minf =
    ymin1    ymin2
    _____    _____

        0        0
        0        0
        0        0
    -1000     -500
    -1000    -1000
        0        0
        0        0
        0        0
    -1000    -1000
    -1000     -320
        0        0
    -1000     -500
```

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

figure
plot1 = bar(cell2mat(maxfxs(1:end,:)));
hold on
plot2 = bar(cell2mat(minfxs(1:end,:)));

xticklabels({'DM_atp_c_', 'ACOAHi', 'ALCD21__D', 'HMR__2087',...
    'LALDO', 'LCADim', 'ME2m', 'AKGDm', 'PGI', 'PGM', 'r0062', 'HMR__2088'})
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 LPproblems has been set to ibm_cplex.

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

> IBM ILOG CPLEX interface added to MATLAB path.
> Solver for QPproblems 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: (5835,10600)
 >> All reactions are solved (10600 reactions - 100%).
 >> 0 reactions out of 10600 are minimized (0.00%).
 >> 0 reactions out of 10600 are maximized (0.00%).
 >> 10600 reactions out of 10600 are minimized and maximized (100.00%).

 -- Starting to loop through the 4 workers. --

 -- The splitting strategy is 0. --

-------------------------------------------------------------------------------
--  Task Launched // TaskID: 2 / 4 (LoopID = 3) <> [5301, 7950] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:34:11 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: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:40:18 2017
 >> Time spent in FVAc: 367.4 seconds.
-------------------------------------------------------------------------------
 ==> 25.0% done. Please wait ...

-------------------------------------------------------------------------------
--  Task Launched // TaskID: 3 / 4 (LoopID = 4) <> [7951, 10600] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:34:11 2017
 >> #Task.ID = 3; logfile: cplexint_logfile_3.log

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

        -- Minimization (iRound = 0). Number of reactions: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:40:59 2017
 >> Time spent in FVAc: 407.9 seconds.
-------------------------------------------------------------------------------
 ==> 50.0% done. Please wait ...

-------------------------------------------------------------------------------
--  Task Launched // TaskID: 1 / 4 (LoopID = 1) <> [1, 2650] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:34:11 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: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:41:59 2017
 >> Time spent in FVAc: 468.0 seconds.
-------------------------------------------------------------------------------
 ==> 75.0% done. Please wait ...
```

```
--------------------------------------------------------------------------------
--  Task Launched // TaskID: 4 / 4 (LoopID = 2) <> [2651, 5300] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:34:11 2017
 >> #Task.ID = 4; logfile: cplexint_logfile_4.log

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

        -- Minimization (iRound = 0). Number of reactions: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:42:11 2017
 >> Time spent in FVAc: 480.6 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: (5835,10600)
 >> All reactions are solved (10600 reactions - 100%).
 >> 0 reactions out of 10600 are minimized (0.00%).
 >> 0 reactions out of 10600 are maximized (0.00%).
 >> 10600 reactions out of 10600 are minimized and maximized (100.00%).

 -- Starting to loop through the 4 workers. --

 -- The splitting strategy is 0. --

--------------------------------------------------------------------------------
--  Task Launched // TaskID: 2 / 4 (LoopID = 3) <> [5301, 7950] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:42:21 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: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:49:08 2017
 >> Time spent in FVAc: 406.4 seconds.
--------------------------------------------------------------------------------
 ==> 25.0% done. Please wait ...

--------------------------------------------------------------------------------
--  Task Launched // TaskID: 3 / 4 (LoopID = 4) <> [7951, 10600] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:42:21 2017
 >> #Task.ID = 3; logfile: cplexint_logfile_3.log

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

        -- Minimization (iRound = 0). Number of reactions: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:50:16 2017
 >> Time spent in FVAc: 474.4 seconds.
--------------------------------------------------------------------------------
 ==> 50.0% done. Please wait ...
```

```
------------------------------------------------------------------------------
--  Task Launched // TaskID: 1 / 4 (LoopID = 1) <> [1, 2650] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:42:21 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: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:51:18 2017
 >> Time spent in FVAc: 536.6 seconds.
------------------------------------------------------------------------------
 ==> 75.0% done. Please wait ...

------------------------------------------------------------------------------
--  Task Launched // TaskID: 4 / 4 (LoopID = 2) <> [2651, 5300] / [5835, 10600].
 >> The number of reactions retrieved is 2650
 -- Start time:     Thu Jul  6 18:42:21 2017
 >> #Task.ID = 4; logfile: cplexint_logfile_4.log

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

        -- Minimization (iRound = 0). Number of reactions: 2650.
        -- Maximization (iRound = 1). Number of reactions: 2650.
 -- End time:     Thu Jul  6 18:51:24 2017
 >> Time spent in FVAc: 542.8 seconds.
------------------------------------------------------------------------------
 ==> 100% done. Analysis completed.
```

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

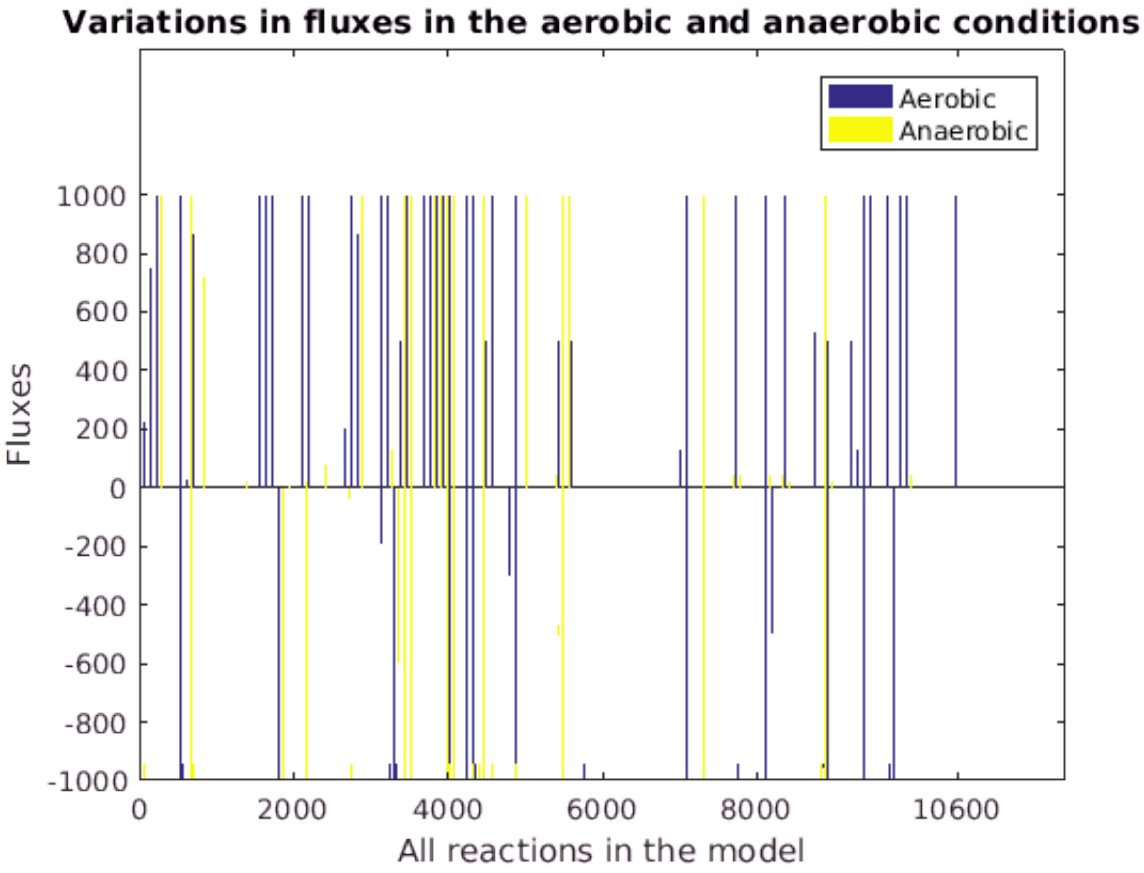
```
ymaxf1 = maxFluxF1;
yminf1 = minFluxF1;
ymaxf2 = maxFluxF2;
yminf2 = minFluxF2;

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,:)));

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., Ines Thiele; Computationally efficient flux variability analysis. *BMC Bioinformatics.* 11, 489 (2010).

[2] Laurent Heirendt, Ines Thiele, Ronan M. T. Fleming; DistributedFBA.jl: high-level, high-performance flux balance analysis in Julia. *Bioinformatics.* 33 (9), 1421-1423 (2017).

[3] Ines Thiele, Nathan D. Price, Thuy D. Vo, Bernhard O. Palsson; Candidate Metabolic Network States in Human Mitochondria. Impact of diabetes, ischemia and diet. *J Bio Chem.* 280 (12), 11683–11695 (2005).