Tutorial 2 – Construct a Functional Small Model

This exercise deals with a small glycolysis model in RAVEN compatible Excel format and shows the most basic aspects of the stoichiometric modelling. It is shown how to build a simple model from scratch, set parameters and perform simple simulations.

Open tutorial2.m in MATLAB to begin this exercise. This script contains all the functions needed to complete this exercise. The solutions to this exercise can be found in tutorial2_solutions.m file. This script is nearly identical to tutorial2.m, just the different model file (small.xlsx) is imported here. During this exercise, the user is supposed to modify empty.xlsx file for it to be nearly identical to small.xlsx file, that the same results would be obtained in tutorial2.m and tutorial2_solutions.m scripts.

Figure 1 shows a somewhat simplified version of glycolysis.

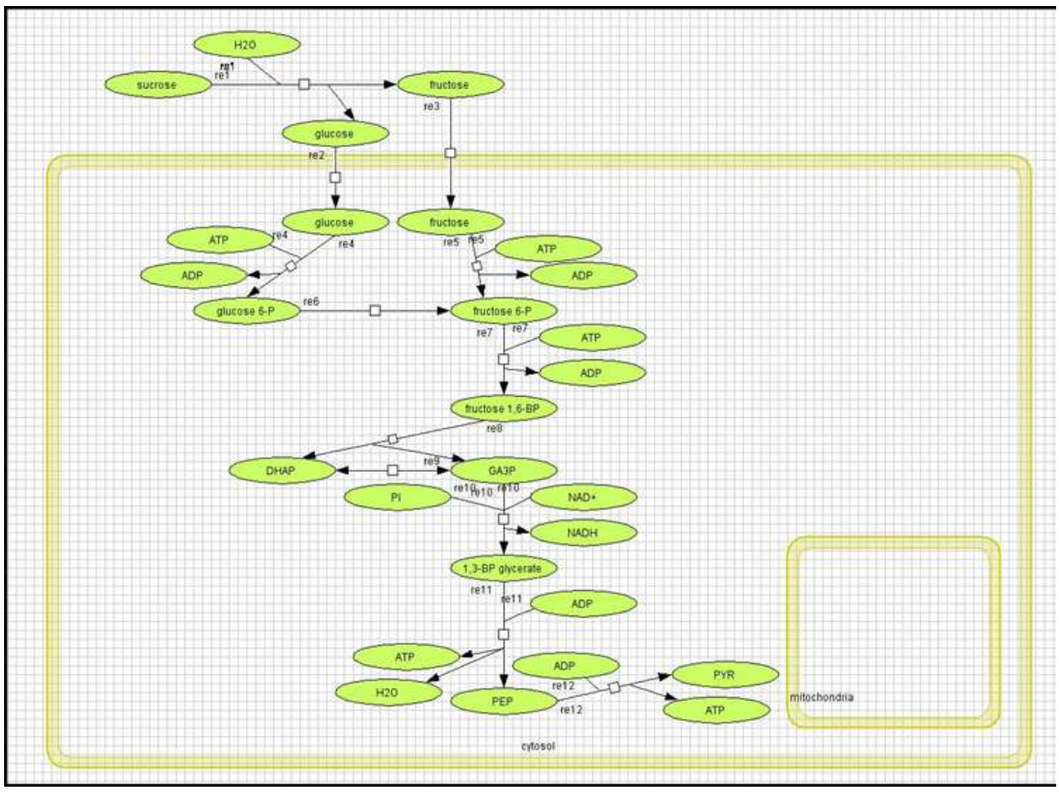

Figure 1. Glycolysis (some reactions are lumped together).

The task here is to make a model for this pathway to answer how many units of ATP could be generated through glycolysis. Even though such information can be retrieved from Google in 15 seconds, the goal here is to get such information through the stoichiometric modelling.

In [ ]:
```
setRavenSolver('gurobi')
```

In [ ]:
```
smallModel=importExcelModel('./tutorial_data/empty.xlsx');
printModelStats(smallModel);
```

```
NOTE: DEFAULT LOWER not supplied. Uses -1000
NOTE: DEFAULT UPPER not supplied. Uses 1000
Network statistics for empty: Empty model structure
NOTE: DEFAULT UPPER not supplied. Uses 1000
Network statistics for empty: Empty model structure
Genes*                          1
        extracellular   1

Reactions*                      1
        extracellular   1
Unique reactions**      1

Metabolites                     4
        extracellular   4
Unique metabolites      4

* Genes and reactions are counted for each compartment if any of the corresponding m
etabolites are in that compartment. The sum may therefore not add up to the total nu
mber.
** Unique reactions are defined as being biochemically unique (no compartmentalizati
on)
```

The task here is to make a model for this pathway to answer how many units of ATP could be generated through glycolysis. Even though such information can be retrieved from Google in 15 seconds, the goal here is to get such information through the stoichiometric modelling.

1. Use Microsoft Excel to open the model file empty.xlsx, which already contains the first reaction of the glycolysis pathway.
2. In MATLAB run the first command from tutorial2.m (i.e., the one with importExcelModel). This line converts the GEM in Excel format to the MATLAB structure and performs several consistency checks at the same time. The program should warn that there is no information about gene associations in the model and that some metabolites are only used in one reaction. The second warning will be discussed later.
3. Add the remaining 11 reactions to the GEM in Excel format. One can use the arbitrary abbreviations, e.g., “g6p” for “glucose 6-phosphate”. Save often and run importExcelModel to ensure that the model structure is correct.
4. Once all the reactions have been entered, run importExcelModel once again. One should see a warning like:

WARNING: The following internal metabolite(s) are only used in one reaction (zero flux is the only solution):

(m13 [c]) H2O
(m14 [e]) H2O
(m15 [c]) NAD+
(m16 [c]) NADH
(m18 [c]) phosphate
(m19 [c]) pyruvate
(m20 [e]) sucrose

This is a particularly important warning, which must be correctly understood before moving on. Since the modelling is done under the steady state assumption, the production rate of a given metabolite must be the same as the consumption rate. But what if there are no reactions consuming or no reactions producing a given metabolite? Then the production/consumption rate must be zero, which is not a desirable result. The warning simply lists metabolites that only participate in one reaction, which means that it either cannot be consumed or that it cannot be produced. The fact that the warning says, “internal metabolites” indicates that this is not the only kind of metabolite and the solution may lie in something we call “external metabolites”. These are metabolites which do not have to be mass balanced but can be produced or consumed in any amount. Simulations are often centred on choosing which external metabolites are available and how fast they can be

produced or consumed by the GEM. External metabolites are defined by having “true” in the “UNCONSTRAINED” field in the “METS” sheet. They are also by convention placed in a compartment abbreviated “b”, referred as “boundary”. These metabolites are also called “exchange metabolites” and the reactions involving them are called “exchange reactions”.

In [ ]:
```
%Import the Excel model into a RAVEN model structure
smallModel=importExcelModel('./output/solved_empty_a.xlsx');
printModelStats(smallModel);
```

```
Network statistics for hans_tuto2: Me doing tutorial 2
Genes*                          12
        extracellular   3
        cytosol 11
Genes*                          12
        extracellular   3
        cytosol 11

Reactions*                      12
        extracellular   3
        cytosol 11
Unique reactions**      12

Metabolites                     20
        extracellular   4
        cytosol 16
Unique metabolites      17

* Genes and reactions are counted for each compartment if any of the corresponding m
etabolites are in that compartment. The sum may therefore not add up to the total nu
mber.
** Unique reactions are defined as being biochemically unique (no compartmentalizati
on)
```

❗ **Personal note:** importing this model did not raise a warning despite the model having dead-end metabolites. Moreover, I did not see in the code of 'importExcelModel' or 'checkModelStruct implemented the message that the tutorial mentions...

In [ ]:
```
checkProduction(smallModel);
```

```
NOTE: The exchange reactions are assigned to the first compartment
The following metabolites could not be produced:
1,3-Bisphospho-D-glycerate[c]
ADP[c]
The following metabolites could not be produced:
1,3-Bisphospho-D-glycerate[c]
ADP[c]
ATP[c]
D-Fructose 1,6-bisphosphate[c]
D-Fructose 6-phosphate[c]
D-Glyceraldehyde 3-phosphate[c]
D-fructose[c]
D-fructose[e]
D-glucose 6-phosphate[c]
D-glucose[c]
D-glucose[e]
Dihydroxyacetone phosphate[c]
H2O[c]
H2O[e]
NAD+[c]
NADH[c]
Orthophosphate[c]
Phosphoenolpyruvate[c]
Pyruvate[c]
Sucrose[e]
```

In [ ]:
```
sol=solveLP(smallModel);
fprintf('Solve message: "%s". The f value is %5.4f.\n', sol.msg, sol.f)
```

```
Solve message: "Optimal solution found". The f value is Optimal solution found". The
f value is 0.0000.
```

5. In the case for glycolysis, it is essential to allow for uptake of sucrose and production of pyruvate. At this point maybe it is not clear whether there will be net consumption or net synthesis of water, so one should make that exchange reaction reversible. The reaction for taking up sucrose would look like “sucrose[b] => sucrose[e]”. Add the required reactions for sucrose, pyruvate and H2O. Usually, it is only allowed for uptake/excretion from the “extracellular” compartment so add transport reactions when needed. The warning should now read:

WARNING: The following internal metabolite(s) are only used in one reaction (zero flux is the only solution):

(m16 [c]) NAD+
(m17 [c]) NADH
(m19 [c]) phosphate

6. All the “real” reactions involved in glycolysis have been added to the GEM. However, to be able to answer the question about the ATP production one should address the two remaining problems:

    a. How does one get the required NAD+ and how does one get rid of the produced NADH?

    b. How does one formulate the ATP production as a variable one can solve for?

7. The first problem could be solved either by expanding the model to contain a larger proportion of metabolism so that the model has a way to regenerate NAD+ from NADH or by including a “fake” uptake reaction for NAD+ and a “fake” excretion reaction for NADH. These reactions are called “fake” because they are not how the system works in vivo (the cell does not take up extracellular NAD+ and so on…). The use of fake reactions is common in this type of modelling and cleverly designing fake reactions can help a lot when doing simulations. Choose either to deal with the problem by including ethanol production from pyruvate (via pyruvate decarboxylase + alcohol dehydrogenase) or by including fake exchange reactions.

8. The second problem also requires fake reactions. Remember that the variables that are solved for are fluxes through reactions. One way to know how much ATP the system can generate is to maximize for the degradation of ATP (since the production and consumption must match). Add a fake reaction for hydrolysis of ATP. Be careful about directionality so that the free ATP synthesis is not added instead. Upon completion, one should see no warnings when running importExcelModel.

9. The GEM is now complete and can finally be used to answer the question about the ATP production. In the GEM modelling the problem is defined by (a) setting constraints on the fluxes and (b) defining an objective for the simulation.

10. An usual modelling practice is to set constraints only for the exchange reactions; the things that the GEM can consume and produce. In this case it is enough to constrain the uptake of sucrose. Set the “UPPER BOUND” for that reaction to 1.0 unit. Once talking about the fluxes, the units are mmol/gDW/h, but when one looks at yields, one might as well think of them as mol or “molecules”. Set the objective to maximize ATP degradation by putting 1 in the “OBJECTIVE” column for that reaction.

11. Import the model to MATLAB by running importExcelModel. One should see that the model structure is printed. Select the name “smallModel” from the MATLAB “Workspace” section and double click it with left mouse button (or just write “open smallModel”). Click around a little bit and try to figure out what the different fields stand for.

12. Solve the optimization problem by running solveLP. Print the resulting exchange fluxes to be sure that everything worked like it should. Make sure that the carbon balance is correct.

13. Print all the fluxes.

In [ ]:
```
smallModel=importExcelModel('./output/solved_empty_b.xlsx');
printModelStats(smallModel);
```

```
Network statistics for hans_tuto2: Me doing tutorial 2
Genes*                          13
        extracellular   4
        cytosol 12
Genes*                          13
        extracellular   4
        cytosol 12

Reactions*                      24
        extracellular   12
        cytosol 18
Unique reactions**      24

Metabolites                     26
        extracellular   7
        cytosol 19
Unique metabolites      20

* Genes and reactions are counted for each compartment if any of the corresponding m
etabolites are in that compartment. The sum may therefore not add up to the total nu
mber.
** Unique reactions are defined as being biochemically unique (no compartmentalizati
on)
```

In [ ]:
```
sol = solveLP(smallModel);
disp(sol);
printFluxes(smallModel, sol.x, false, 10^-7);
```

```
        x: [24x1 double]
        f: -4
     stat: 1
      msg: 'Optimal solution found'
   sPrice: [26x1 double]
    rCost: [24x1 double]

FLUXES:
r1      (Breakdown of sucrose (invertase)):     1
r2      (Glucose transport):    1
FLUXES:
r1      (Breakdown of sucrose (invertase)):     1
r2      (Glucose transport):    1
r3      (Fructose transport):   1
r4      (Glucose 6 kinase):     1
r5      (Fructose 6 kinase):    1
r6      (Glucose to fructose):  1
r7      (Fructose phosphate 1 kinase):  2
r8      (Fructose 1,6 BP aldolase):     2
r9      (DHAP GA3P isomerase):  2
r10     (1,3-Bisphospho-D-glycerate synthase):  4
r11     (rx11): 4
r12     (rx12): 4
EX_sucrose      (exchange sucrose):     1
EX_H2O  (exchange water):       1
synth_2 (ATP recycling):        4
r13     (Pyruvate decarboxylate):       4
r14     (Ethanol dehydrogenase):        4
trans_CO2       (trans carbon dyoxide): 4
EX_CO2  (exchange carbon dyoxide):      4
trans_etOH      (transport ethanol):    4
EX_etOH (exchange ethanol):     4
```

Question 1: how much ATP could be generated from one unit of sucrose?

In [ ]:
```
printFluxes(smallModel, sol.x, false, 10^-7, [], '%rxnID (%rxnName):\n\t%eqn\n\t%fl
```

```
FLUXES:
r4 (Glucose 6 kinase):
        D-glucose[c] + ATP[c] => ADP[c] + D-glucose 6-phosphate[c]
        1
r5 (Fructose 6 kinase):
        D-fructose[c] + ATP[c] => ADP[c] + D-Fructose 6-phosphate[c]
        1
r7 (Fructose phosphate 1 kinase):
        ATP[c] + D-Fructose 6-phosphate[c] => ADP[c] + D-Fructose 1,6-bisphosphate
[c]
        2
r4 (Glucose 6 kinase):
        D-glucose[c] + ATP[c] => ADP[c] + D-glucose 6-phosphate[c]
        1
r5 (Fructose 6 kinase):
        D-fructose[c] + ATP[c] => ADP[c] + D-Fructose 6-phosphate[c]
        1
r7 (Fructose phosphate 1 kinase):
        ATP[c] + D-Fructose 6-phosphate[c] => ADP[c] + D-Fructose 1,6-bisphosphate
[c]
        2
r11 (rx11):
        ADP[c] + 1,3-Bisphospho-D-glycerate[c] => H2O[c] + ATP[c] + Phosphoenolpyruv
ate[c]
        4
r12 (rx12):
        ADP[c] + Phosphoenolpyruvate[c] => ATP[c] + Pyruvate[c]
        4
synth_2 (ATP recycling):
        H2O[c] + ATP[c] => ADP[c] + Orthophosphate[c]
        4
```