# The RAVEN Tutorials

[Exercise 0 – Import a GEM, Set Parameters and Run FBA 2](#_Toc22573933)

[Exercise 1 – Construct a Functional Small Model 3](#_Toc22573934)

[Exercise 2 – Apply KOs, Run MOMA and Integrate External Data 7](#_Toc22573935)

[Exercise 3 – Fix an Erroneous Model 11](#_Toc22573936)

[Exercise 4 – Reconstruct and Refine a GEM from KEGG 15](#_Toc22573937)

[Exercise 5 – Reconstruct and Refine a GEM from MetaCyc+KEGG 16](#_Toc22573938)

[Answers to Questions 17](#_Toc22573939)

# Exercise 0 – Import a GEM, Set Parameters and Run FBA

This is a short introduction that shows how to load a genome-scale metabolic model (GEM), set reaction constraints, objective function and perform an optimization through flux balance analysis (FBA). The resulting fluxes are visualized and exported to a PDF file.

A GEM for the filamentous fungus *Penicillium chrysogenum* is used in this tutorial. The model can be found in a Microsoft Excel file under the name iAL1006 v1.00.xlsx and in SBML file iAL1006 v1.00.xml.

Open tutorial0.m file in MATLAB to begin this exercise. To run a section of code in MATLAB, highlight it, press right mouse button on it and choose an option “Evaluate selection”.

**NOTE:** the user must be able to successfully import the GEMs in Excel format with the RAVEN function importExcelModel. Although, this functionality is not necessary for this exercise, the users without such ability would not be able to do Exercises 1-3, which involve working with GEMs in RAVEN compatible Excel format.

# Exercise 1 – Construct a Functional Small Model

This exercise deals with the 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 tutorial1.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 tutorial1\_solutions.m file. This script is nearly identical to tutorial1.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 tutorial1.m and tutorial1\_solutions.m scripts.

-----

Figure 1 shows a somewhat simplified version of glycolysis.

|  |
| --- |
| glycolysis.jpg  **Figure 1.** Glycolysis (some reactions are lumped together). |
|  |

The task here is to make a model for this pathway in order 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 tutorial1.m (i.e. the one with importExcelModel). This line converts the GEM in Excel format to the MATLAB structure and performs a number of 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 similar to:

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 very 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”.

1. 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

1. All the “real” reactions involved in glycolysis have been added to the GEM. However, in order to be able to answer the question about the ATP production one should address the two remaining problems:
   1. How does one get the required NAD+ and how does one get rid of the produced NADH?
   2. How does one formulate the ATP production as a variable one can solve for?
2. 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 reality (the cell does not take up extracellular NAD+ and so on…). The use of fake reactions is very 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.
3. 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 has to 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.
4. The GEM is now complete and can be 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.
5. 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.
6. 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.
7. 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.
8. Print all the fluxes.

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

# Exercise 2 – Apply KOs, Run MOMA and Integrate External Data

This exercise shows how to run FBA and minimization of metabolic adjustment (MOMA) simulations and how one can use GEMs as a scaffold for interpreting microarray data. A simplified model of yeast metabolism is used in this approach as an example.

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.

-----

Figure 2 shows the system, this time it is a central carbon metabolism. Note that there are slight differences in naming of metabolites; particularly all metabolites have their compartment in their name.

|  |
| --- |
| **Figure 2.** Central carbon metabolism in yeast (adapted from Förster, 2003). |

1. Open the file smallYeast.xlsx. One will hopefully recognize the model structure from Exercise 1, even though some more stuff is added this time. The most important difference is the composition for metabolites and gene associations for reactions. Metabolite composition information is useful for finding errors in the GEMs and knowing gene associations allows to link the GEMs to other types of data, such as microarrays.

By default the production of acetate, biomass, glycerol, CO2, and ethanol is allowed and the consumption of O2 and glucose is not allowed. Try to validate the model by confirming the results from Exercise 1 (the maximal ATP production in glycolysis). One can set the simulation parameters in the Excel file like in Exercise 1, or try to use the setParam function instead (see tutorial2.m). Just remember that sucrose was used in the first example while glucose is used here.

1. Calculate the maximal yields of biomass, ethanol, glycerol, and acetate on glucose and fully aerobic (biomass is in 1/h and the other fluxes are in mmol/gDW/h).
2. Assume that one wants to predict how the flux distribution would change if the conditions are changed. The user should know by now how to import models and set parameters. Import the model twice and generate one model which has access to 1 mmol/gDW/h of glucose and unconstrained oxygen, and one which has access to 1 mmol/gDW/h of glucose and 0.5 mmol/gDW/h of O2. Optimize for growth under both conditions. What were the outputs? Take a look in the tutorial and try to run drawMap for the two models. Open the resulting PDF file and see if one can understand (in broad terms) how metabolism changes between the two conditions.
3. Say that the goal is to increase glycerol production in the strain (e.g. to make sweeter wine). FBA is useful for suggesting gene deletions which can couple the set goal with the overall objective for the cell, which is to grow as fast and efficient as possible. Set the parameters to maximize for growth under anaerobic conditions and run the code for making a single gene deletion using FBA. How large was the difference in glycerol production? Can one come up with a hypothesis as to how the deletion leads to higher glycerol production? One can also use the function followChanged, which compares two flux distributions and let the user search between them (see tutorial2.m)
4. FBA is based on the assumption that cells have the same objective even after a perturbation; to grow as efficiently as possible. This is probably true after long evolution times, but shortly after, say, a gene deletion the cell might have other objectives. MOMA is based on that the perturbed cell would like to change its metabolism as little as possible. MOMA is normally used when one has the experimental data for the wild type strain and would like to guess how the phenotype would change after a perturbation. These are some batch data for growth on glucose:
   * acOUT (Production of acetate): 0
   * biomassOUT (Production of biomass): 0.67706
   * co2OUT (Production of CO2): 22.4122
   * ethOUT (Production of ethanol): 19.0946
   * glyOUT (Production of glycerol): 1.4717
   * glcIN (Uptake of glucose): 15
   * o2IN (Uptake of O2): 1.6
   * ethIN (Uptake of ethanol): 0

Load one model with these constraints and one without any constraints on the exchange reactions but with the ZWF reaction turned off. Run the qMOMA function and plot the flux distributions. Is the glycerol production larger?

1. GEMs can also be used to aid in interpretation of other types of data. The last lines load microarray data from batch growth on ethanol compared to glucose and perform a Reporter metabolites test. These are metabolites around which significant transcriptional changes occur (for such a small model it is not super interesting though, because there are changes almost everywhere). The very last lines plot all reactions involving any of those metabolites. This can be very useful when working with large models.

# Exercise 3 – Fix an Erroneous Model

The whole point of GEMs is that they are large. It is by incorporating the entire known metabolism of any given organism that complexity arises. However, this makes it almost certain that all models will contain errors. This is true regardless of whether one builds the model or if one uses a model from someone else. One of the issues is that if one group publishes a model for some specific purpose it is likely to function well in that specific part of metabolism, but it may not function at all for some problems. It is therefore a good idea to perform a round of error checking even if it is a published model one uses.

Model validation is an iterative process because some errors might not have an effect until some other errors have been fixed. It is not uncommon that the model “works” well in the beginning of the reconstruction process because there are errors that let it cheat on things like redox or energy balance. The model then works worse and worse as the errors are dealt with until all or most errors are fixed, after which it will start to work again. As RAVEN developers, we believe that it is much more important to try to make the model do something it should not be allowed to do rather than to test for the stuff it should do.

Open tutorial3.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 tutorial3\_solutions.m file.

-----

There is a version of the small yeast model with errors inserted (smallYeastBad.xlsx). The task for this exercise is to find and fix them. Some errors will be obvious (it is rather difficult to introduce errors in such a small model, because there is very little redundancy in it), but it is strongly recommended not to fix them until they are “found” during the following steps, because otherwise one might get unpredictable results. Most of the stuff done here can be done with the gapReport function, but it is strongly recommended to do them step by step.

1. The first thing to check for is that the model cannot make something from nothing, i.e. no metabolites should be produced if one does not give the model access to any carbon sources (this should be done for all elements, but carbon is the most important). A simple way to do this would be to optimize for the sum of all the producing exchange reactions, while keeping the consuming reactions closed. Any solution other than 0 would then be bad. Try that.
2. The previous step did not provide any non-zero solutions, right? That is good, but there could be other factors that prevent the error from showing its ugly face. Maybe it costs energy or redox power for example. Or maybe the necessary reactions are in different compartments. It is generally a good idea to relax as many constraints as possible when searching for errors. For instance, one can include a temporary reaction like “ATP + H2O ⬄ ADP + Pi” and similar reactions with NADH and NADPH. Remember that the aim here is to try to “provoke” the model to show the errors. Add these reactions and try again.
3. Did one get the production of ethanol? If so, print the resulting fluxes and see if it is possible to find the error. GEMs are normally very underdetermined, which means that there are infinite numbers of solutions to any given problem. When one solves using solveLP(model) one just gets a random solution which meets the objective and satisfies the constraints. These solutions often contain loops and are therefore difficult to interpret. One can read more about the solveLP function by typing “help solveLP” in MATLAB, but here it is chosen to solve using solveLP(model,1). This minimizes the sum of fluxes in order to have more easily interpreted results. Find and fix the error and rerun.

**Question 2:** what modification is needed to prevent the of ethanol from nothing?

One should not have any production of anything now.

1. In GEMs it is normal to have excretion of only a few metabolites while having very many internal metabolites. A common case is that one has an error that would like to produce something from nothing, but in order to do so it also has to produce some other metabolite for which there is no exchange reaction. A convenient way to test this is to allow all metabolites to be excreted. One can do this by changing the model.b structure. Normally it is always a vector of zeros, but if one adds a second column RAVEN will interpret it as lower and upper bound on the equality constraints. So if one puts model.b=[model.b inf(numel(model.b),1)]; one can now excrete anything. Do this and see if the model can produce anything. For instance, one should get ethanol, glycerol, and CO2. Look at the fluxes and find the error. One can get a clue by looking at the warnings from SBMLFromExcel. Since this is a problem that comes from reactions being unbalanced, the problematic ones have to be in one of the warnings. Which was the metabolite that had to be excreted in order for the error to appear? Do this step two times to find both errors.

**Question 3:** what two modifications are needed to fix the warnings?

1. The same thing done in step 4 can be done with the function canProduce. There is a sister function called canConsume. It checks which metabolites can be consumed by the model. Change so that no production is allowed and run canConsume. One should see that 12 metabolites could be consumed even though the model is not allowed to produce anything. Pick one of them, force uptake of it by setting the lower bound to non-zero. If one does this one may not be able to get a feasible solution. That is because the problem solved by canConsume allows input of all metabolites, but the current model allows input for only O2 and glucose. Modify the model.b variable to allow for uptake of all metabolites.

**Question 4:** study the fluxes and try to find the wrong one. What fix should be applied to the corresponding reaction?

Even if one fixes the problem one will see that the model can still get rid of O2. This is because of the reactions that were included for testing (NAD ⬄ NADH is not elementally or redox balanced). This part of the exercise is done, so those reactions can now be deleted.

1. Unbalanced reactions are a relative small problem, since they are so easy to find. A much bigger problem is when metabolites are named differently even though they are meant to be the same. Use smallYeastBad2.xlsx from here on. A first check is to see which reactions can carry flux when one allows for all uptakes and outputs of exchange metabolites. There are several ways to check this, but use the function simplifyModel here. The primary purpose of this function is to remove unnecessary stuff from a model to make it smaller, but since it removes “bad” reactions one can use it for error identification as well. If one runs it like it is in tutorial3.m one will see that there are about 20 metabolites and reactions that are dead ends. That is quite a lot, so take a look at the warnings from importExcelModel and see if it is possible to catch the obvious spelling error.

**Question 5:** what correction should be applied to fix the spelling error?

1. That did not help very much. Sometimes it is very difficult to find out where the root of the problem is. This is particularly true if it is in a region with many interconversions between metabolites and no clear input/output (Figure 3).

|  |
| --- |
| **Figure 3.** An example of pathway featuring many interconversions between metabolites and unclear input/output. If one reaction is wrong here it will be difficult to find since everything looks so connected because it is produced and consumed in many reactions. |

A powerful but somewhat tricky function is checkProduction. It helps to identify metabolites needed to synthesize in order to to have the net synthesis of everything. Look at the suggestions from checkProduction if when running it like in the tutorial3.m. The function minToConnect tells that is needed to synthesize 12 metabolites in order to have the net synthesis of everything. However, 8 of them are co-factors or contain co-factors. Since there is no net synthesis of co-factors in this small model those are not very interesting (coenzyme A or ATP are not synthesized from glucose). One should take a look at the top one that is not a co-factor. This one is a bit tricky and one might want to look it up in KEGG.

**Question 6:** what is the suspicious similarity between some metabolites?

1. Still quite a bit of dead ends, and nothing that immediately looks like it would fix everything. It could be that some reactions are missing. One could try to include reactions from a set of other models to fill the gaps. This is a computationally expensive task for a large network, but for this small model it is easy. One could use any model structure, but here one can take the small yeast model from Exercise 2. Run the code and include the suggested reaction(s). Run the previous tests to make sure that everything works.
2. Finished! And do not forget that the gapReport function does all these things.

# Exercise 4 – Reconstruct and Refine a GEM from KEGG

This exercise is about creating a model from KEGG, based on protein sequences in a FASTA file, and doing some functionality checks on the model. The example case is for the yeast *Saccharomyces cerevisiae*. This tutorial is more of a showcase and its main purpose is to serve as a scaffold to reconstruct a GEM for any organism.

Open tutorial4.m in MATLAB to begin this exercise.

# Exercise 5 – Reconstruct and Refine a GEM from MetaCyc+KEGG

This exercise demonstrates how to reconstruct a combined draft GEM from KEGG and MetaCyc pathway databases. A combined model with the comprehensive coverage of metabolic pathways is generated from different de novo reconstruction approaches. The input is a FASTA format file with whole-proteome sequences. The combined model is subsequently used for refinement of existing high-quality model and generation of a new version of GEM, by utilizing the manual curation results. This tutorial is a showcase of the new features released in RAVEN 2.0 through demonstrating the utilization of the newly developed functions on GEM reconstruction and curation for *Streptomyces coelicolor* strain A3(2). Users may apply this tutorial as the template in their own work for other organisms.

Open tutorial2018.m in MATLAB to begin this exercise.

# Answers to Questions

**Question 1**: 4 mol/mol

**Question 2:** Change ADH1 from

“acetaldehyde[c] + NADH[c] => 2 ethanol[c] + NAD(+)[c]”

to

“acetaldehyde[c] + NADH[c] => ethanol[c] + NAD(+)[c]”.

**Question 3:** Change FBP from

“beta-D-fructofuranose 1,6-bisphosphate[c] => 2 beta-D-fructofuranose 6-phosphate[c] + phosphate[c]”

to

“beta-D-fructofuranose 1,6-bisphosphate[c] => beta-D-fructofuranose 6-phosphate[c] + phosphate[c]”

and change PFK from

“ATP[c] + beta-D-fructofuranose 6-phosphate[c] => ADP[c] + 2 beta-D-fructofuranose 1,6-bisphosphate[c]”

to

“ATP[c] + beta-D-fructofuranose 6-phosphate[c] => ADP[c] + beta-D-fructofuranose 1,6-bisphosphate[c]”.

**Question 4:** Change PDC from

“pyruvate[c] => acetaldehyde[c]”

to

“pyruvate[c] => acetaldehyde[c] + CO2[c]”.

**Question 5:** 6-O-phosphono-D-glucono-1,5-lactonec] in ZWF1 should be 6-O-phosphono-D-glucono-1,5-lactone[c].

**Question 6:** Dihydroxyacetone phosphate (DHAP) and glycerone phosphate (GLYP) are the same metabolite.