Practical Bioinformatics
Wagner Section

Exercise Block 1. Introduction of metabolic networks and metabolic network analysis

Stoichiometric matrix

Stoichiometric coefficients represent the relative molar amounts of reactants (educts, substrates) and products participating in a chemical reaction. For example, in the reaction A + 2 B -> C, the stoichiometric coefficient of A is 1, that of B is 2 and that of C is 1. To denote whether a molecule is a reactant or a product of a reaction, we add a sign to its stoichiometric coefficient. Thus, the stoichiometry of A is -1, the stoichiometry of B is -2 and that of C is 1.

Metabolic reaction networks can be very large and comprise hundreds to thousands of reactions. For such large networks, it is useful to represent the network's structure in the form of a stoichiometric matrix. Flux balance analysis (FBA) uses this representation, which is compact and simple. Consider the simple metabolic reaction 'network' consisting of the following three reactions R1, R2 and R3.

R1: A + 2 B -> C
R2: B + C -> D
R3: A + D -> 2 E

This network can be represented in a stoichiometric matrix of m rows and n columns, where each row corresponds to one kind of molecule, and where each column corresponds to one reaction. The structure of this matrix is shown in the Table below.

	R1	R2	R3
A	-1	0	-1
B	-2	-1	0
C	1	-1	0
D	0	1	-1
E	0	0	2

Reactions R1-R3 are irreversible reactions, as indicated by the unidirectional arrows in their stoichiometric equations. A reversible reaction can be split into two unidirectional reactions. For example, if R1 is a reversible reaction, it can be split into two reactions R1f (forward) and R1b (backward). That is, the stoichiometric equation

R1: A + 2 B <-> C

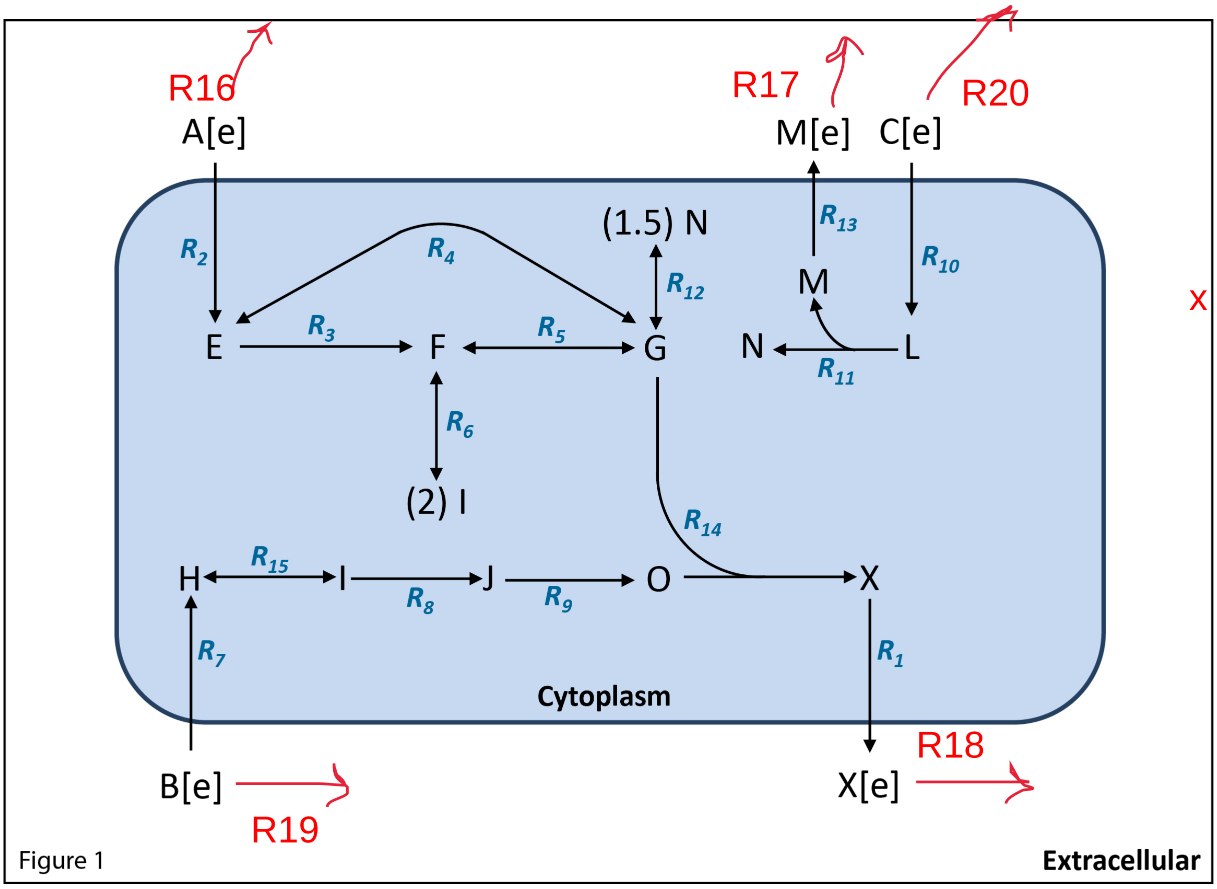
Would translate into

R1f: $A + 2 B \rightarrow C$
R1b: $C \rightarrow A + 2B$

Exercise 1.1. Draw the structure of the stoichiometric matrix above if all reactions R1-R3 are reversible.

Exercise 1.2. For the reaction network below (figure 1):
a) Draw the stoichiometric matrix manually (feel free to show non-zero entries only).
b) Think of three alternative ways to produce X when A, B and C are present in the extracellular compartment (and when the metabolic network is in a steady state).
c) Now imagine the situation where B and C are not present in the extracellular compartment but A is. If the flux through reaction R2 takes a value of 10 mmol/gDW/hr, what's the maximum rate at which X can be produced assuming steady state?

Note that metabolic network analysis often uses different symbols for the same metabolite in two different biological compartments, such as the cytoplasm and the extracellular space. Specifically, [e] is usually used to denote extracellular space.



essential:
R:2,1

Figure 1 – A simple toy metabolic network.

You have so far performed a metabolic optimization similar to that of FBA, but manually with a simple network. FBA can be solved by inspection with networks as simple as this one but as a network gets more complex it becomes impossible to solve manually. Let's now try to perform FBA with the assistance of computers.

COBRApy is a python package to model biological networks computationally (http://opencobra.github.io/). It allows building and reading models in various formats as well as simulating FBA. The basics of how to use COBRApy will be introduced as they are needed in this course. Old (up to 0.6.0) and new (0.6.0 onwards) versions of COBRApy differ in the way they retrieve the solutions. In this document we describe how to perform all analysis with an old and new version of COBRApy. We suggest that you check the COBRApy version you have installed before getting started: in python, import COBRApy with `import cobra` and check the version with `cobra.__version__`. You can then follow the instructions that suit the COBRApy version you have installed. We also encourage you to take a look at the documentation (https://cobrapy.readthedocs.io/en/stable/).

Before FBA can be performed, a computational model of a metabolic network has to be

created in the computer. Models can be easily created and written in standard formats with the aid of COBRApy. The following paragraphs explain how to build a model and save it in SBML format. SMBL(Systems Biology Markup Language) is an XML-based standard format for representing and distributing computational models of biological systems. You can also take a look at the part of the COBRApy documentation explaining how to build a metabolic model (https://cobrapy.readthedocs.io/en/stable/building_model.html).
Let's build the model for the network used in our initial example, which has 3 reactions and 5 metabolites. For the present example let's consider R3 to be a reversible reaction:

R1: A + 2 B -> C
R2: B + C -> D
R3: A + D <-> 2 E

The python script must start loading the needed packages and defining the model:

```
#Import needed packages
import cobra
from cobra import Model, Reaction, Metabolite

#Define model
cobra_model = Model('example_BuildingModel')
```

Then we need to define the reactions. As stated earlier, if a reaction is reversible it can be split into two unidirectional reactions. Alternatively, the reaction can be defined in one direction only and the directionality is then indicated by setting limits on the flux value. Genome scale metabolic models (metabolic models including all reactions that can take place in an organism) indicate the directionality of reactions by giving a lower and an upper bound for the flux of each reaction. We will follow the same convention for building our example:

```
#Define reactions
reaction1 = Reaction('R1')
reaction1.lower_bound = 0.
reaction1.upper_bound = 1000.

reaction2 = Reaction('R2')
reaction2.lower_bound = 0.
reaction2.upper_bound = 1000.

reaction3 = Reaction('R3')
reaction3.lower_bound = -1000.
reaction3.upper_bound = 1000.
```

The above statements created three reactions with reaction ids R1, R2 and R3. The reversibility or irreversibility of a reaction is given by the flux boundaries. Reaction 3 can take a positive or negative flux value, which indicates that the reaction is reversible.

Now it is time to introduce the metabolites and linking the metabolites to the reactions:

```
#Define metabolites
A = Metabolite('A')
B = Metabolite('B')
C = Metabolite('C')
D = Metabolite('D')
E = Metabolite('E')

#Linking (‘adding’) metabolites to reactions with their respective stoichiometric
coefficients
reaction1.add_metabolites({A: -1.0,
                           B:-2,
                           C: 1.0})
reaction2.add_metabolites({B: -1.0,
                           C: -1.0,
                           D: 1.0})
reaction3.add_metabolites({A: -1.0,
                           D: -1.0,
                           E: 2.0})
```

The model is still empty; we now must include the reactions in the model:

```
#Add the reactions to the model, which will automatically add all associated
metabolites
cobra_model.add_reaction(reaction1)
cobra_model.add_reaction(reaction2)
cobra_model.add_reaction(reaction3)
```

The model is now built. You can check the number of reaction and metabolites included in the model with `len(cobra_model.reactions)` and `len(cobra_model.metabolites)` respectively. You can also inspect your model further by typing:

```
for x in cobra_model.reactions:
    print("%s : %s : %s : %s" % (x.id, x.reaction, x.lower_bound, x.upper_bound))
```

Once you have checked that the model contains everything you wanted to it is time to save it. This will allow you to work with it later without the need to build it again. With the following line you will save your model in SBML format. The file will be saved in your current working directory. If you want to change the location just add the specific path before the file name.

```
#Save model in SBML format
cobra.io.write_sbml_model(cobra_model,'example_BuildingModel')
```

In the file example_BuildingModel.py you have a script with all these commands that creates and saves the example model.

Exercise 1.3. Building model including external reactions and objective function.

Now you know the basics of creating a model. It is time to introduce some terminology and conventions that will facilitate working with more complex models in the following exercises. Genome scale metabolic models also include external reactions and external metabolites that are not part of the model itself but facilitate analysis of the model. An external metabolite represents a metabolite that occurs in the environment outside a cell. For an external metabolite to exist there must be a transport reaction, also referred to as an external reaction, which allows the metabolite to be transported between the interior and the exterior of the cell. External reactions are added to simplify the analysis of a network. The external reaction associated with each external metabolite, by convention, consumes the metabolite (the external metabolite is the reactant of the external reaction). As a consequence, if a metabolite is imported into a cell, the associated external reaction will have a negative flux value. If the metabolite is excreted, the reaction will have a positive value.

Modify the script example_BuildingModel.py to include the transport of metabolites A, B and E to the extracellular space and the external reactions. The whole model is now:
R1: A + 2 B -> C
R2: B + C -> D
R3: A + D <-> 2 E
R4: A <-> A[e]
R5: B <->B[e]
R6: E <-> E[e]
R7: A[e] <->
R8: B[e] <->
R9: E[e] <->

FBA can be used to maximize the rate at which any given reaction proceeds. Before performing FBA we must indicate which reaction will be the objective function of the FBA problem, that is, the rate at which this reaction proceeds is to be maximized. The objective function can be indicated with the command:

```
cobra_model.reactions.get_by_id( ' REACTION ID ' ).objective_coefficient = 1.
```

For this exercise, we may be interested in maximizing the production of E[e]. Write the python/cobra statement that indicates that this is the objective function?

Exercise 1.4. Building the model from figure 1.
In the file ToyModel_building.py part of the model has already been introduced. Complete the missing information and save it in sbml format.

Exercise 1.5. Explore how the synthesis rate of X[e] changes, as you change the uptake rate of A[e].
The synthesis rate of X[e] depends on the rate of substrate uptake, in this case A[e]. Increasing the uptake rate of A[e] would result in a proportional increase in the synthesis of X[e]. Perform FBA to get the maximal production of X[e] when A[e] is imported into the cell ("taken up") at a maximal rate of 10 mmol/gDW/hr. Make sure to set the right boundaries on reactions and to select an objective function. For this exercise, the objective function is the reaction R1 that synthesizes X[e].

To read a SBML model (.xml) use:

```
import cobra
from cobra.io import read_sbml_model
model = read_sbml_model('model_name.xml')
```

Once a model has been created you can always read it and temporarily modify its information without changing the saved model. For the specific case of the boundaries, you can change them with:

```
model.reactions.get_by_id("REACTION ID").lower_bound = ANY VALUE.
```

Use the following command to perform FBA:

```
solution = model.optimize()
```

With `cobra_model.summary()` you can get a further description of the solution.

Compare the result to what you got in Exercise1.2 (c).
Now change the uptake rate of A[e] and monitor the rate of synthesis of X[e]. Is the increase in X[e] linear with the increase in A[e]? Why or why not?

Exercise 1.6. List the reactions that are essential for the production of X from input substrate A[e].
Consider, as an example, the reactions involved in the synthesis of metabolite O from the substrate B[e] in figure 1. Synthesis of O from B[e] requires reactions R7, R15, R8 and R9. If we were to delete any one of these reactions, it would be impossible to synthesize O from B[e]. We can therefore say that reactions R7, R15, R8 and R9 are essential for the production of O from substrate B[e]. A reaction is essential for the synthesis of a molecule from a specific substrate if its removal abolishes the ability to synthesize that molecule from the substrate.

Go over Figure 1 and list all reactions that you think are essential for the synthesis of X[e] from substrate A[e]. (You are implicitly assuming that the synthesis rate of X[e] is to be

maximized.)
Here are a few reactions that are worth paying special attention to: Is reaction R3 essential? Why or why not? And what about reaction R6?

The following lines allow you to calculate the effect of deleting a reaction (follow the instructions that suit the COBRApy version you have installed):

<u>COBRApy versions up to 0.6.0:</u>

```
from cobra.flux_analysis import single_reaction_deletion
gr, st = single_reaction_deletion(model, model.reactions[:])
```

The output of the function `single_reaction_deletion` is stored in two python data structures called dictionaries (`gr` and `st` above) whose keys are the reactions ids. The first has the form {reaction_id:flux_value} which contains the growth rate values after deleting each reaction. The second (st) has the form {reaction_id:optimization_status} contains the status of the optimization after deleting each reaction. If there were no problems with the optimization its entries should say “optimal”.

<u>COBRApy versions 0.6.0 and onwards:</u>

```
from cobra.flux_analysis import single_reaction_deletion
solution_deletion = single_reaction_deletion(model, model.reactions[:])
```

The output of the function `single_reaction_deletion` is stored as a pandas dataframe. When printing `solution_deletion` you will see a table with three columns: ids, growth and status. The ‘ids’ column shows the id of every reaction subject to deletion; the ‘growth’ column shows the growth rate values after deleting each reaction; the ‘status’ column contains the status of the optimization after deleting each reaction. If there were no problems with the optimization its entries should say “optimal”. There are multiple ways to excess the information stored in this dataframe. One possibility is to work with it as if it were a dictionary. You can access, for example, the information in the growth column with `solution_deletion[‘growth’]` or get the ids with `solution_deletion[‘growth’].keys()`.

How can you use this to get the essential reactions from the network? Try it. Now compare your answer with the prediction from flux balance analysis. What do you see?

Exercise 1.7. Identify an active pathway for the synthesis of X[e] from A[e] in the reaction network of Figure 1.
Active reactions are reactions that have a non-zero metabolic flux, that is, reactions proceeding at a rate different from zero.
Study figure 1 to find reactions that have to be active for the synthesis of X[e] from substrate

A[e]. Are these reactions different from the list of essential reactions in exercise 1.5? Why or why not?

After running `solution = model.optimize()`, `solution.x_dict` yields an output dictionary (COBRApy versions up to 0.6.0) or a pandas dataframe (COBRApy versions 0.6.0 onwards) which contains the flux through each reaction. This output will also harbor information about the active reactions.

Compare the list of active reactions obtained in this way with the list you obtained by hand?

Exercise 1.8. Flux variability analysis and alternative pathways.
In exercise 1.7 you found an active pathway to produce X[e] from A[e], is the solution that FBA came up with the only possible one? Why or why not? Pay particular attention to reactions R3 and R4.
Once you have taken some time to think about this, let's use COBRApy to find out what is going on. To do so, we will use Flux Variability Analysis (FVA).
FVA calculates the maximum and minimum flux that each reaction in a model can have, at a given rate of production of X[e].
What do you expect FVA to yield for the rates of essential reactions? What about active reactions?

To perform FVA with COBRApy we can type:
```
fva_result = cobra.flux_analysis.flux_variability_analysis(model,
model.reactions[:])
```

The result is a more complex data structure, also called a dictionary of dictionaries (COBRApy versions up to 0.6.0), which will look something like this:
```
{'R1': {'maximum': 6.666666666666667, 'minimum': 6.666666666666667},
 'R10': {'maximum': 0.0, 'minimum': 0.0}}
```
or a pandas dataframe (COBRApy versions 0.6.0 onwards).

Up to now we have introduced the concept of essential and active reactions. Let's introduce some new terminology. For the specific case shown here, you can see that both reactions (R1 and R10) show the same maximum and minimum value. When this happens we say that a reaction is fixed. Reaction R10 is fixed with a zero flux value, in which case we say the reaction is blocked. It cannot be active if the production X[e] is maximized.
With the tool of FVA in hand, revisit the uniqueness of the FBA solution. Is there only one way to produce X[e] from A[e]? Look at the flux values of reactions R3, R4 and R5. Does the solution make sense biologically speaking?
Now pay attention to reaction R12. Why is this reaction blocked?

Exercise 1.9. Quantify the maximal yield of product X[e] for each substrate A[e], B[e], and C[e] in the network of Figure 1

The maximal yield of a product is defined as the ratio of the rate of synthesis of the product to the rate of utilization or uptake of the substrate. The maximal yield of X[e] with substrate A[e] can be expressed as $Y_{X[e]/A[e]}$.

To compute the synthesis rate of X[e] from each substrate, use the `model.optimize()` command. Make sure to set the boundaries for each reaction appropriately before performing FBA.

Once a model has been created you can always read it and temporarily modify its information without changing the saved model. For the specific case of the boundaries, you can change them with:

```
model.reactions.get_by_id("REACTION ID").lower_bound = ANY VALUE.
```

Based on yields $Y_{X[e]/A[e]}$, $Y_{X[e]/B[e]}$, and $Y_{X[e]/C[e]}$, which substrate is the best one for the production of X[e]. Why? (Hint: consider the roles of metabolites I and N carefully.)

Exercise 1.10. Build the model with the reactions shown below, save it in SBML format and compute the maximal synthesis rate of E[e].

We have seen earlier that FBA can be used to maximize the rate at which any one given reaction proceeds. In the reaction system below, maximize the production of E[e]. Assume an uptake flux of A of 1 mmol/gDW/hr. List the reactions that need to be active. Explain why these is the case. (Hint: which reactions will allow the complete conversion of A[e] to E[e]?)

R1: A -> D

R2: A -> E + F

R3: D -> 2 E

R4: E -> E[e]

R5: A[e] -> A