Understanding Complex Diseases with Object Oriented Modelling

Miha Moškon[1], Tanja Cvitanović[2], Damjana Rozman[2], and Miha Mraz[1]

[1]*Computational Biology Group, Faculty of Computer and Information Science, University of Ljubljana*
[2]*Centre for Functional Genomics and Bio-Chips, Institute of Biochemistry, Faculty of Medicine, University of Ljubljana*

Object Oriented Modelling

Objects

Each object class corresponds to a specific object type.

Examples from electronics: resistor, current source, inductor, capacitor, battery.

Biological examples: enzyme, protein, enzyme catalysed reaction, mass transfer.

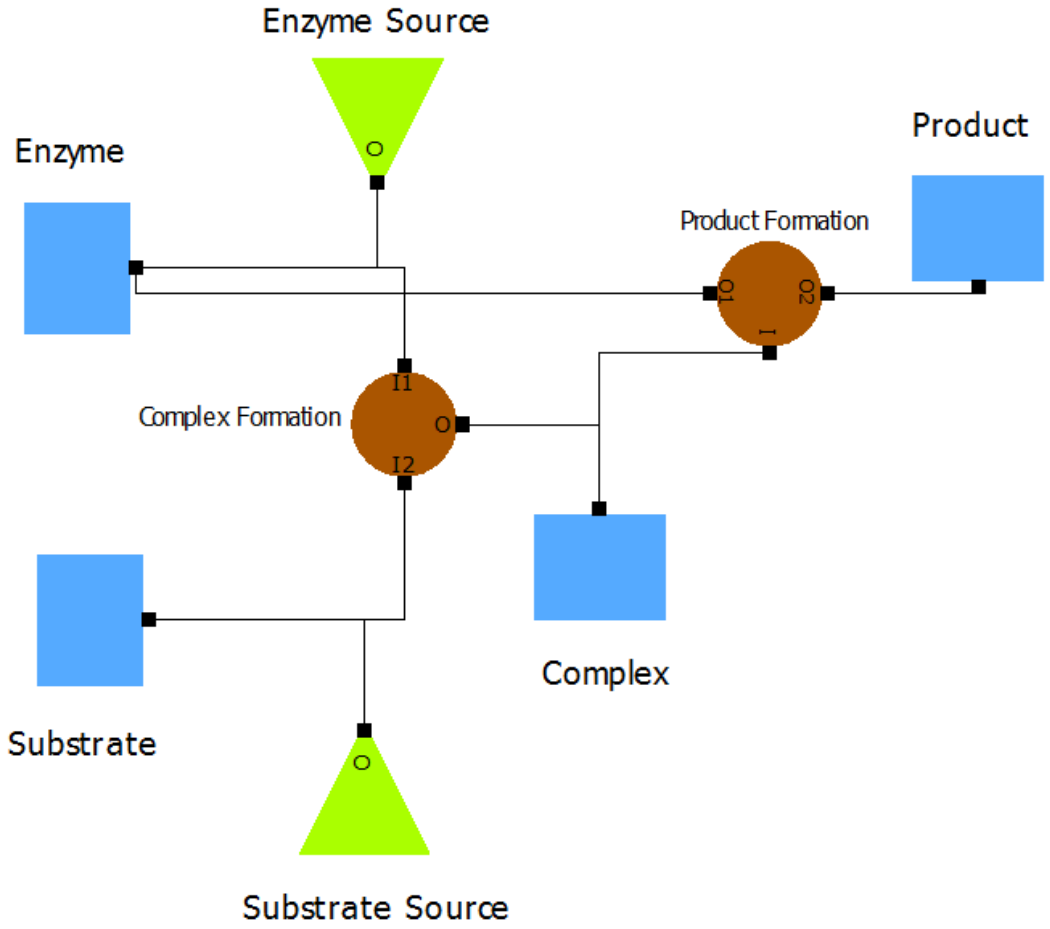
Object Oriented Modelling

Each object is described with its properties (parameters, variables, connectors and equations).

May also contain other objects (hierarchical composition).

Graphical modelling: connecting objects that correspond to actual entities into the whole model.

Example: connecting enzymes, metabolites and reactions that take place in selected metabolic pathway.



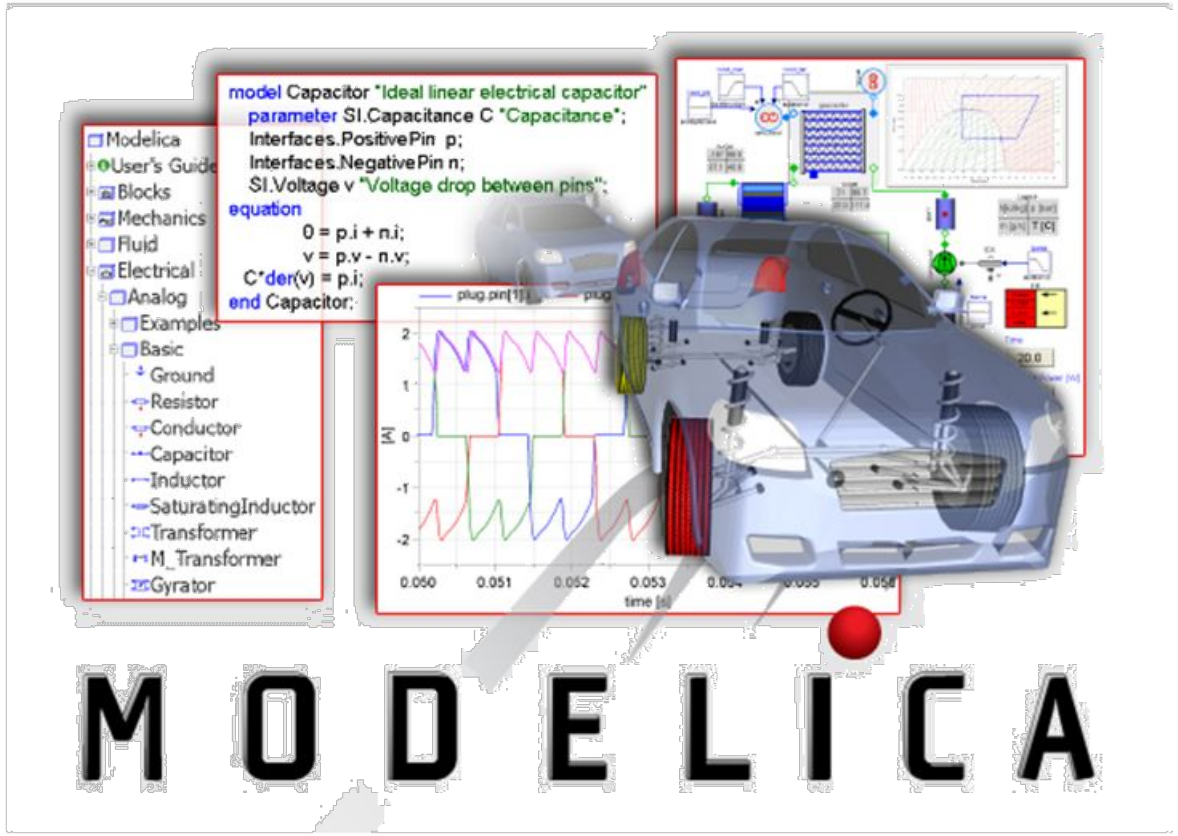
Modelica

A language for modelling of complex physical systems.

Vast scope of applications: robotics, automotive, aircrafts, satellites, power plants, **systems biology**.

Designed for simulation (dynamical modelling).

Modelica is not a tool!



Open Modelica (OMEdit)

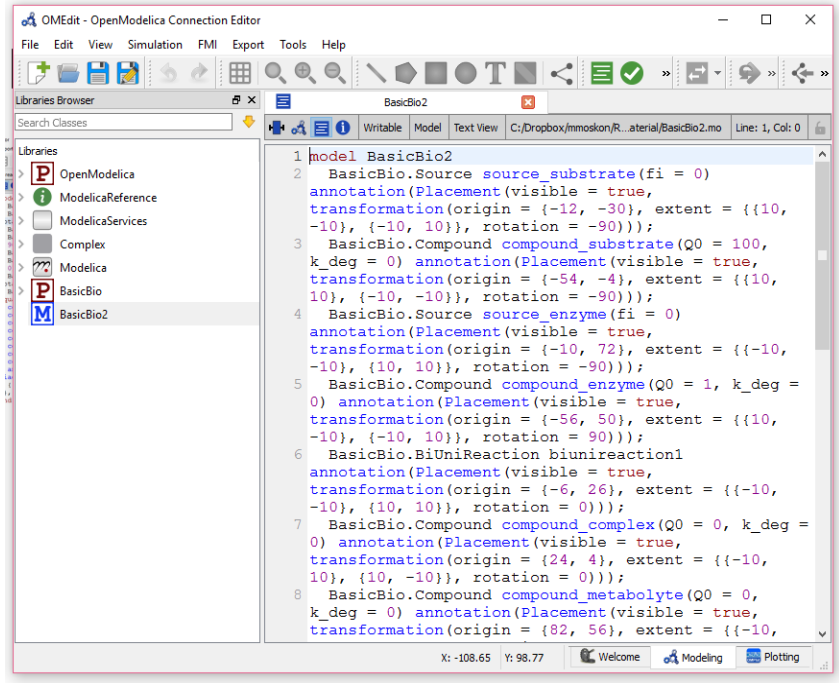
An open-source Modelica-based modelling and simulation environment intended for industrial and academic applications.

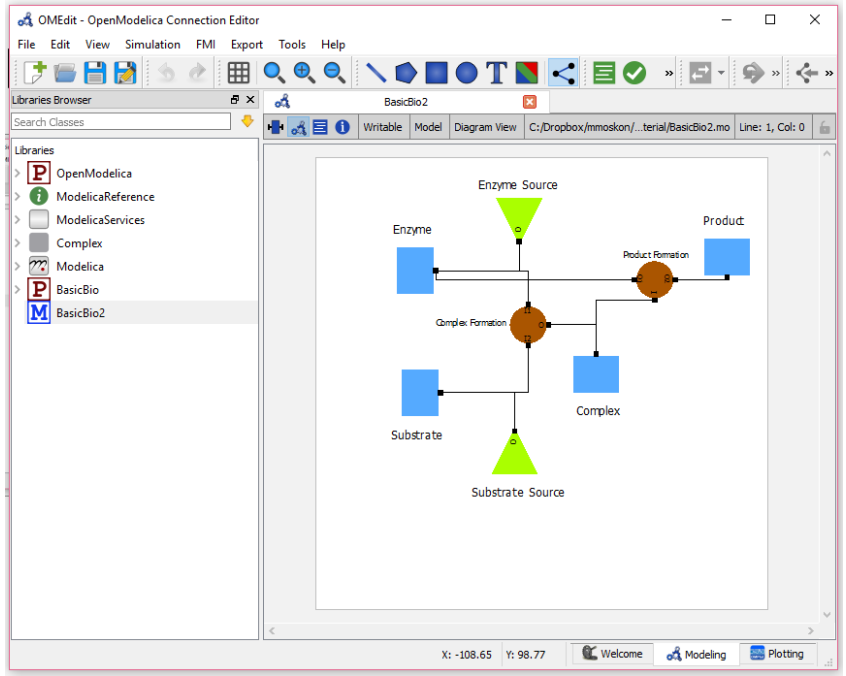
Supported by a non-profit organization – the Open Source Modelica Consortium (OSMC).

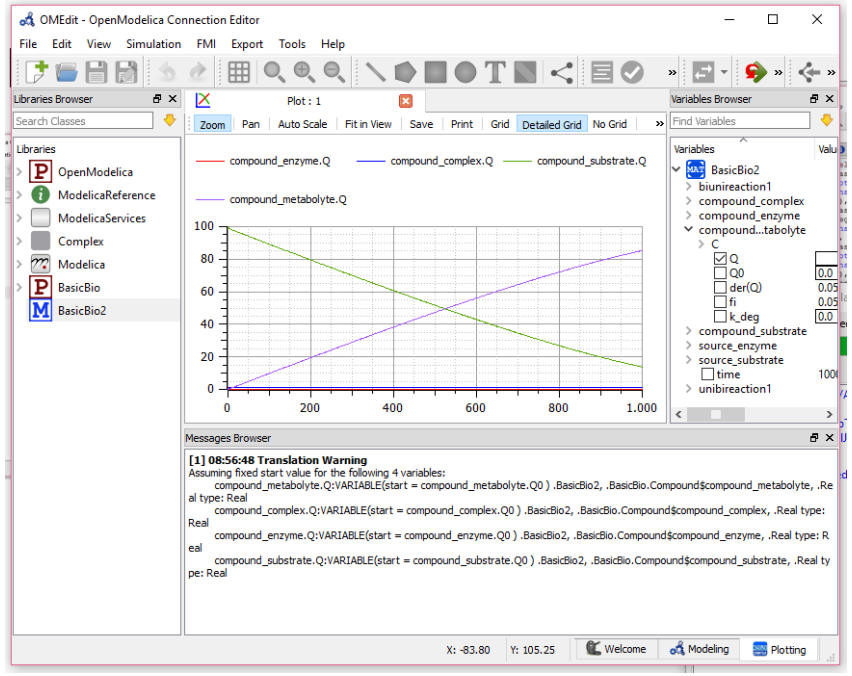
https://www.openmodelica.org/

Examples of other Modelica tools: Dymola (Dassault systems), System Modeler (Wolfram), SimulationX (ITI), MapleSim (MapleSoft).

Open Modelica (OMEdit)







Modelica and Open Modelica

Object Oriented Modelling – Object properties

Parameters:

- values do not change during a single simulation
- example: reaction constants `k_deg`

Variables:

- temporary values
- changes defined with object functionalities and its connectors
- define internal state of the object
- example: compound concentration `Q`

Equations:

- define the object functionality
- how do variables change in dependence on other variables and parameters
- example: degradation described with `der(Q) = - k_deg * Q;`

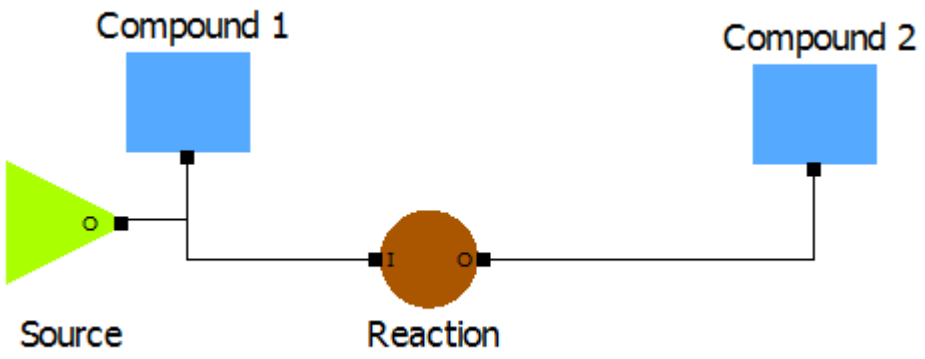
Connectors:

- connect objects with other objects
- belong to *variables* group
- example: substrate and enzyme objects are connected to a reaction object

Object Oriented Modelling – graphical representation

Visualization of each object (not mandatory).

Allows to build and analyse the model in user friendly graphical interface within the simulation environment (e.g. Open Modelica).



Object Oriented Modelling – parameters and variables

Data types: Real, Boolean, Integer, String,… + other object classes.

Parameters prefixed with `parameter`, variables not.

Explicit assignment of parameter values together with their declaration.

Definition of initial values of variables.

```
model FirstObject "My first object"
    parameter Real Q0 = 1.0;
    parameter Real k_deg = 0.01;
    Real Q(start = Q0);
    Real fi;
    FirstConnector C;
```

object class `FirstConnector`

It is also possible to define the initial conditions of the system and calculate parameter values from these (implicit assignmnet).

Hands-on Examples: `Basics/FirstObject.mo` and `Basics/FirstConnector.mo`

Object Oriented Modelling – Writing equations

In the `equation` block

```
equation
      fi = C.fi - k_deg * Q;
      der(Q) = if Q <= 0 and fi < 0 then 0 else fi;
      C.Q = Q;
end FirstObject;
```

Programming languages usually allow us to assign values.

Example: X ← Y * Z

Modelica allows us to define equations (acausal relations).

Example: X = Y * Z might be interpreted as

X ← Y * Z or

Y ← X/Z or

Z ← X/Y

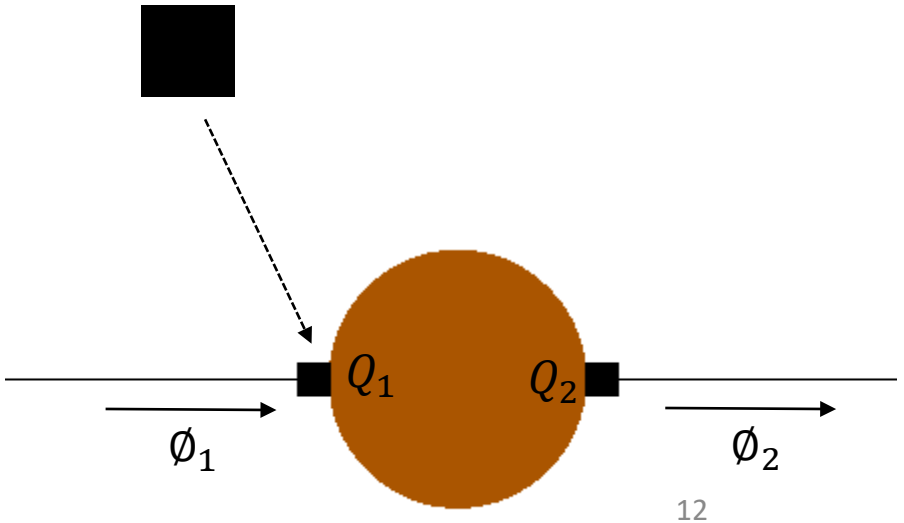
Depends on knowns and unknowns!

Object Oriented Modelling – Object connectors

Interaction with other objects via connectors.

Usually have two properties:

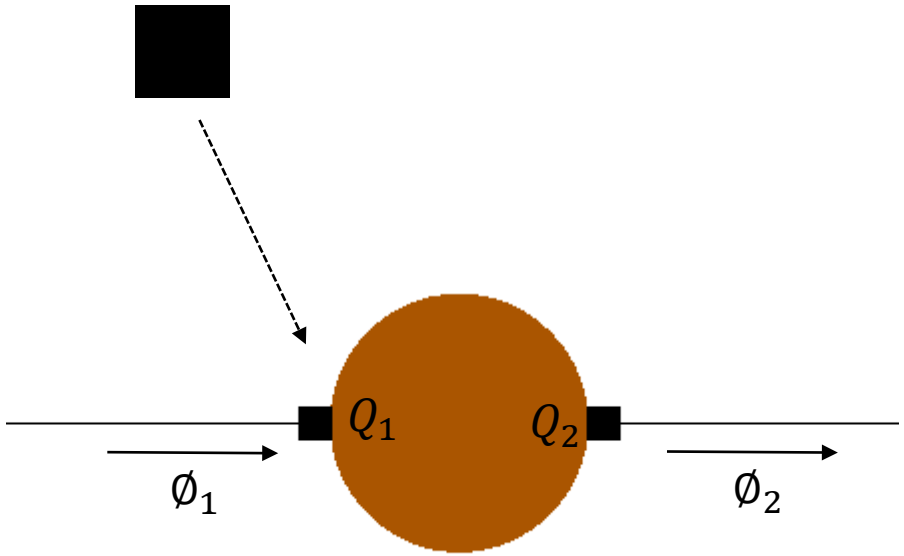
- Flux ($\emptyset$):
  - defines massflow into (influx) or from (outflux) the object
  - variable prefixed with `flow`
- Concentration (Q):
  - defines the compound concentration on the connector
  - ordinary variable



Object Oriented Modelling – Object connectors

Defining a connector and its properties (another object class):

```
connector FirstConnector "My first connector"
        Real Q;
        flow Real fi;
end FirstConnector;
```



Object Oriented Modelling – Object connectors

Placing the connector into the object:

```
model FirstObject "My first object"
      parameter Real Q0 = 1.0;
      parameter Real k_deg = 0.01;
      Real Q(start = Q0);
      Real fi;
      FirstConnector C;

...
```

Object Oriented Modelling – Equations behind the model

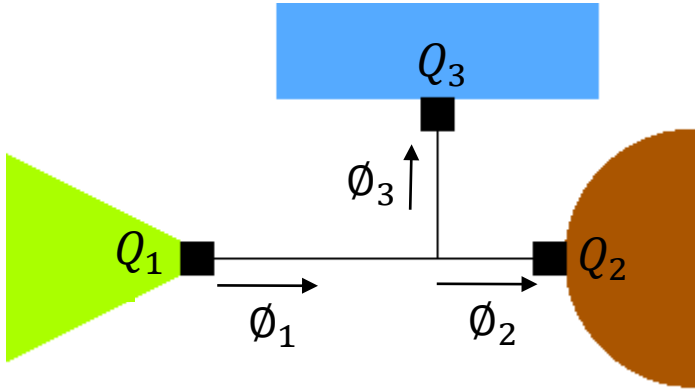
Modelica establishes a system of ODEs to run the simulations.

Equations from within the objects.

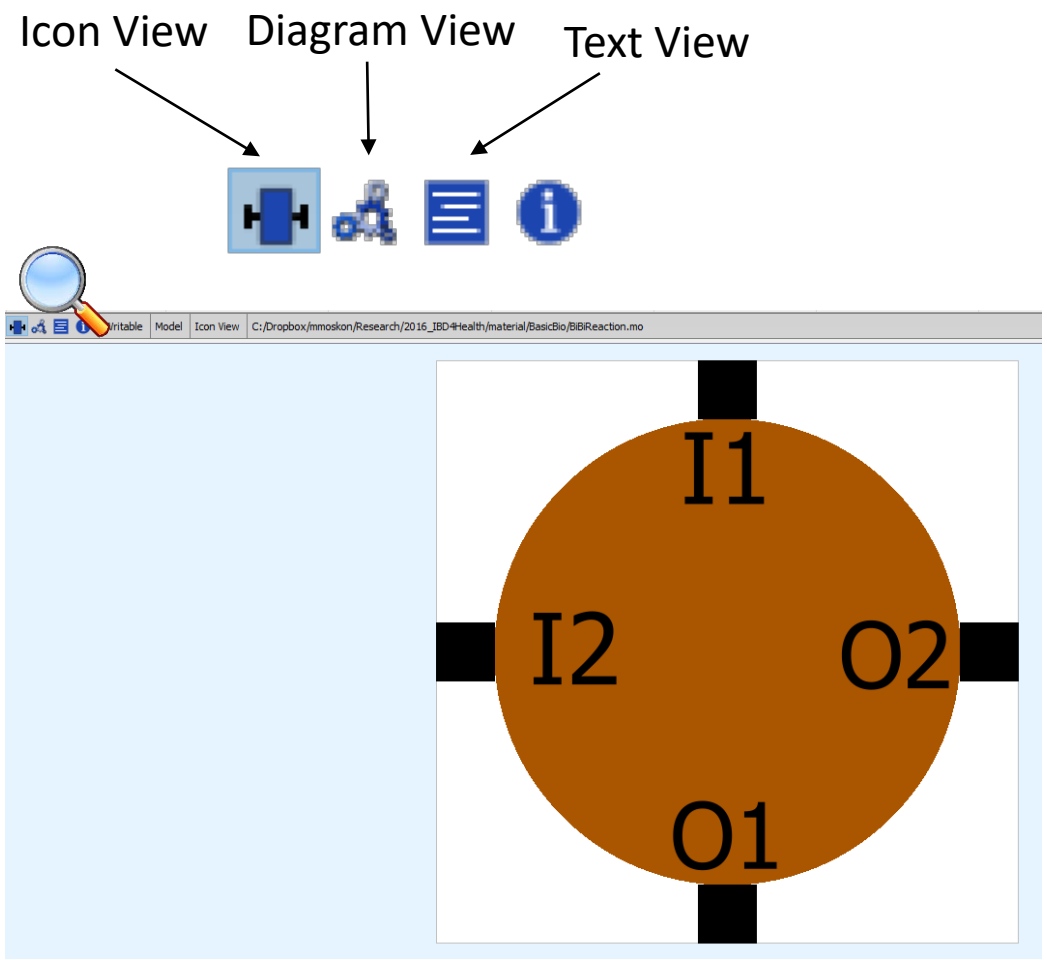
Equations derived from connected connectors:

- $Q_1 = Q_2 = \cdots = Q_m$
- $\sum_{i=1}^{m} \emptyset_i = 0$

Output fluxes should be negative.
Input fluxes should be positive.



Object Oriented Modelling - Defining graphical representation



Icon View – editing object representation

Model view – editing model representation

Text view – editing the code

```
annotation(Diagram, Icon(coordinateSystem(extent = {{-100, -100},
{100, 100}}, preserveAspectRatio = true, initialScale = 0.1, grid
= {2, 2}), graphics = {Ellipse(lineColor = {170, 85, 0}, fillColor
= {170, 85, 0}, fillPattern = FillPattern.Solid, extent = {{-80,
80}, {80, -80}}, endAngle = 360), Text(origin = {60, 64}, extent =
{{-80, -20}, {-40, 20}}, textString = "I1"), Text(extent = {{40, -
20}, {80, 20}}, textString = "O"), Text(origin = {62, -62}, extent
= {{-80, -20}, {-40, 20}}, textString = "I2")}));
```

Inserting classes into libraries

All object classes that are used within the model need to be imported into Open Modelica.

More convenient to use libraries.

- Put all object class files (`*.mo`) in the same folder
- Create `package.mo` file in the folder

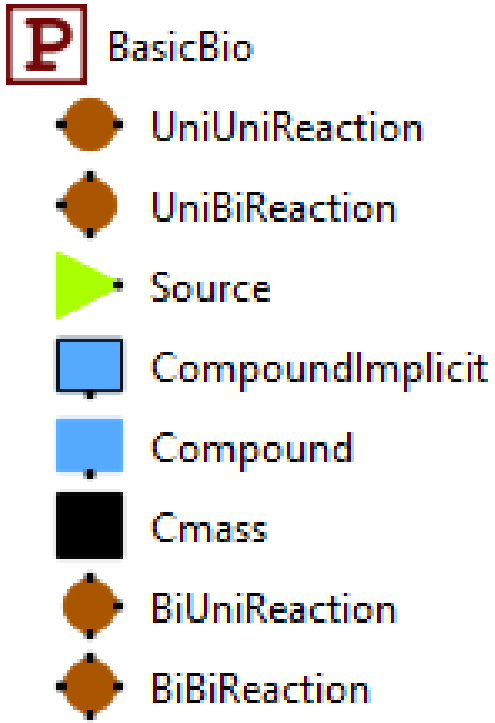
```
package exampleLibrary "Example of a modelica library"
end exampleLibrary;
```

- All object classes should include `within exampleLibrary` in the first line

```
within exampleLibrary;
model ExampleClass "Example of an object class"
...
```

Example: BasicBio library

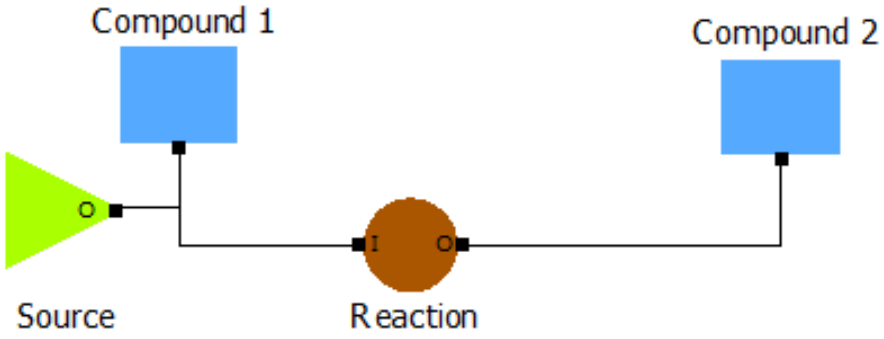
- Connector: `Cmass`
- Compound source: `Source`
- Reactions: `UniUniReaction`, `UniBiReaction`, `BiUniReaction`, `BiBiReaction`
- Compounds: `Compound`, `CompoundImplicit`



Hands-on Example: `BasicBio`

Assignment 1

Construct a model represented with the figure.

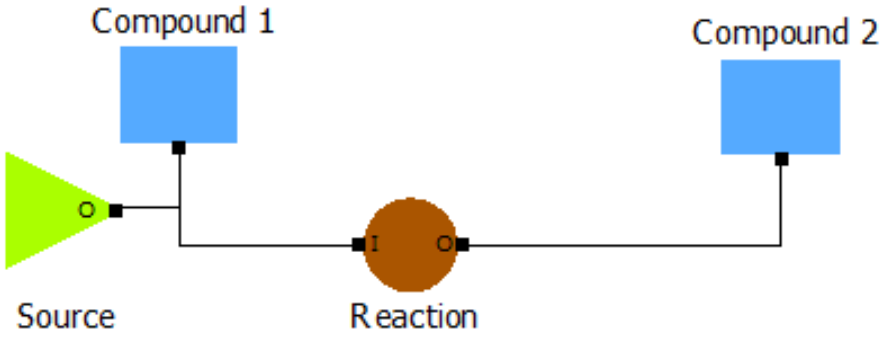


Simulate its dynamics with default parameter values.

Observe the concentrations of Compound 1 and Compound 2.

Model Building

… using graphical user interface of OpenModelica (Diagram View).



Model Building

… writing the code (Text View).

Specification of parameter values.

Name of the model.

```
model BasicBio1

BasicBio.Source source1(fi = 0.5);
BasicBio.Compound compound1;
BasicBio.Compound compound2;
BasicBio.UniUniReaction unireaction1;

equation
connect(compound1.C, source1.O);
connect(compound2.C, unireaction1.O);
connect(unireaction1.I, compound1.C);
end BasicBio1;
```

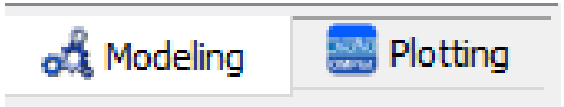
Objects within the model

Connections among objects

Running simulations

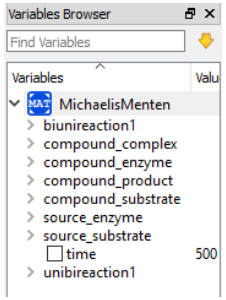
Simulate, Simulation Setup

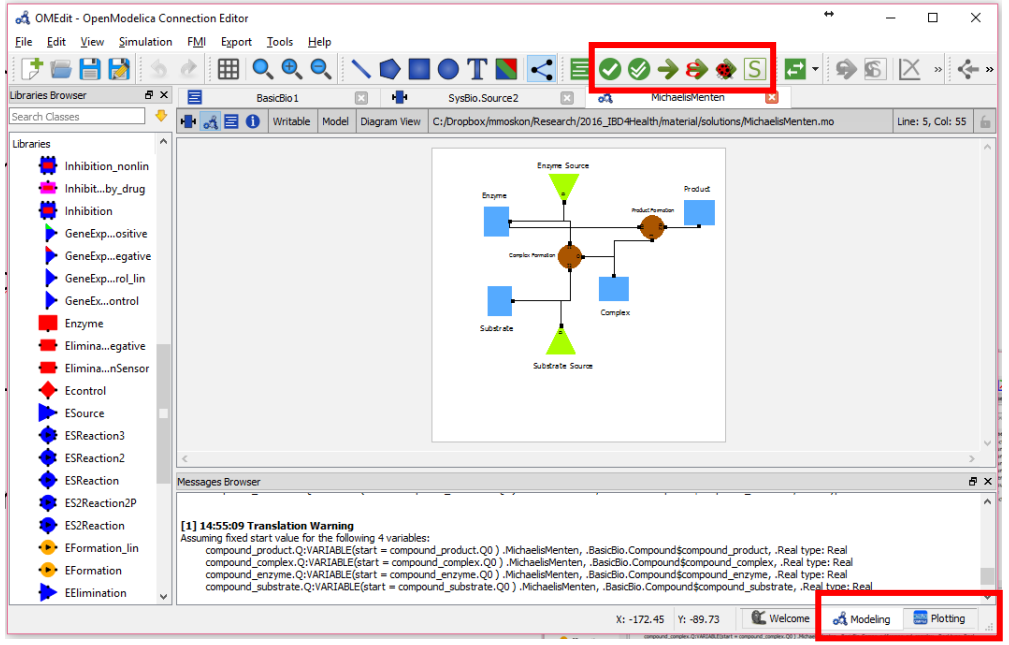
Modeling, Plotting

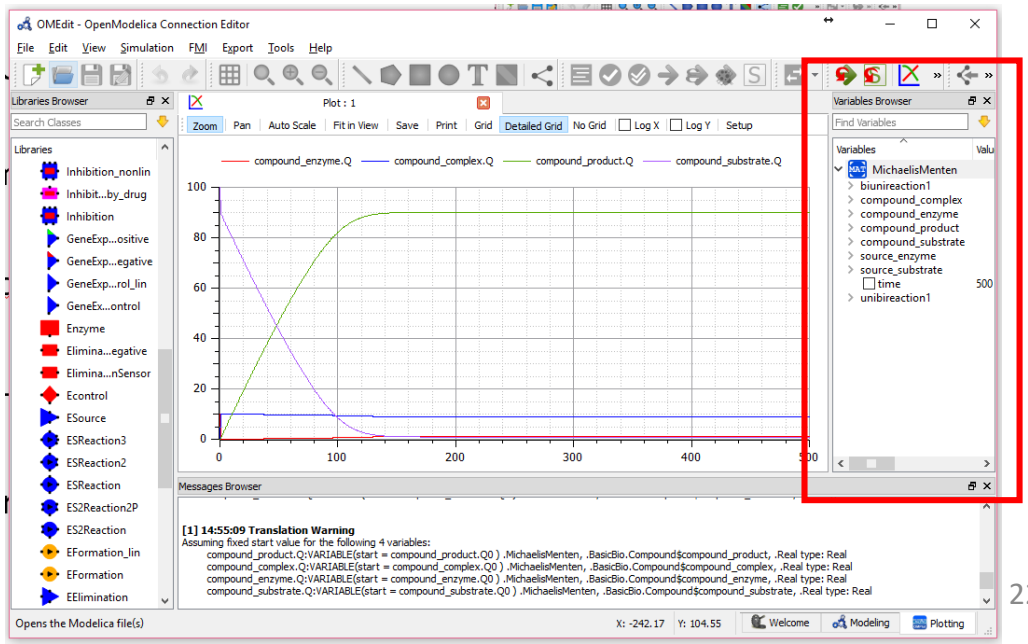


Re-simulate, Re-simulate Setup

Variable Browser, Clear Plot Window

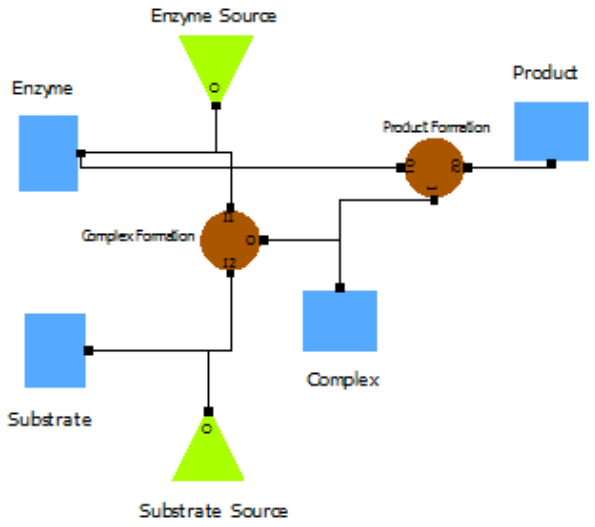






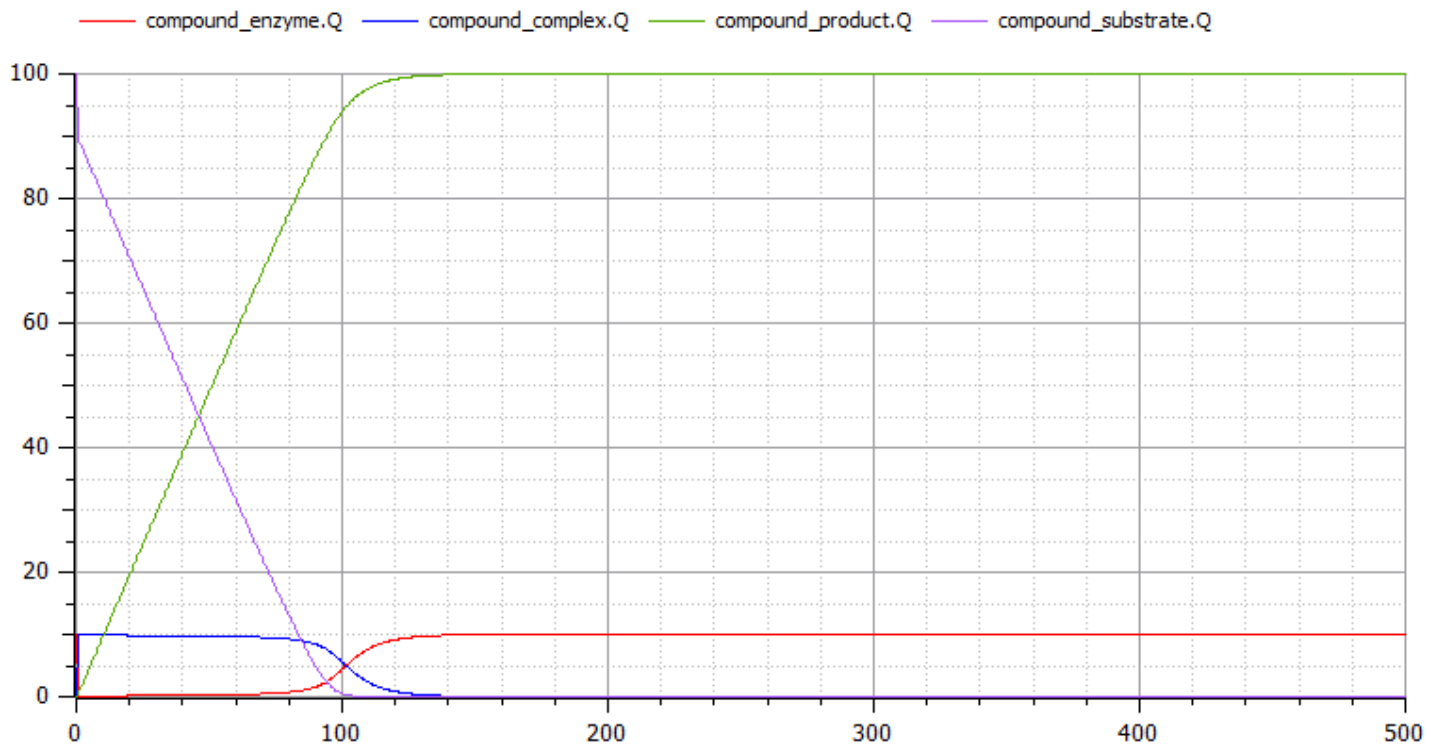
Assignment 2

$$E + S \underset{k_{CR}}{\overset{k_C}{\rightleftharpoons}} C \underset{k_{PR}}{\overset{k_P}{\rightleftharpoons}} P + E$$



Open model file `MichaelisMenten.mo`

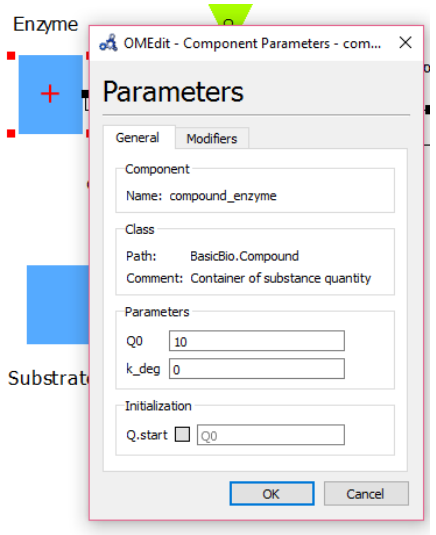
Set the parameters and initial conditions to obtain the following plot



Assignment 2

Changing default parameter values – three options:

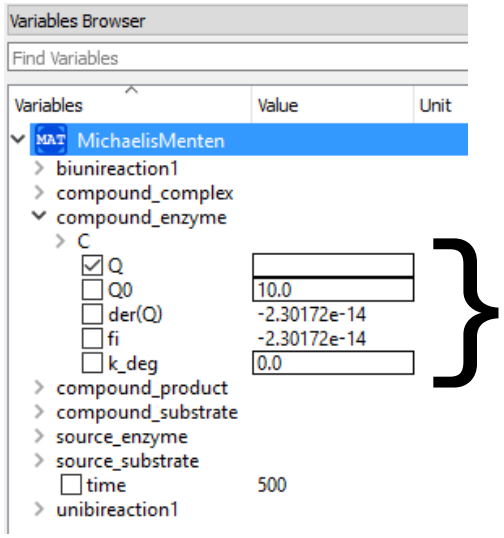
- From Diagram view: right click on the object → parameters



- From Text view: after object declaration

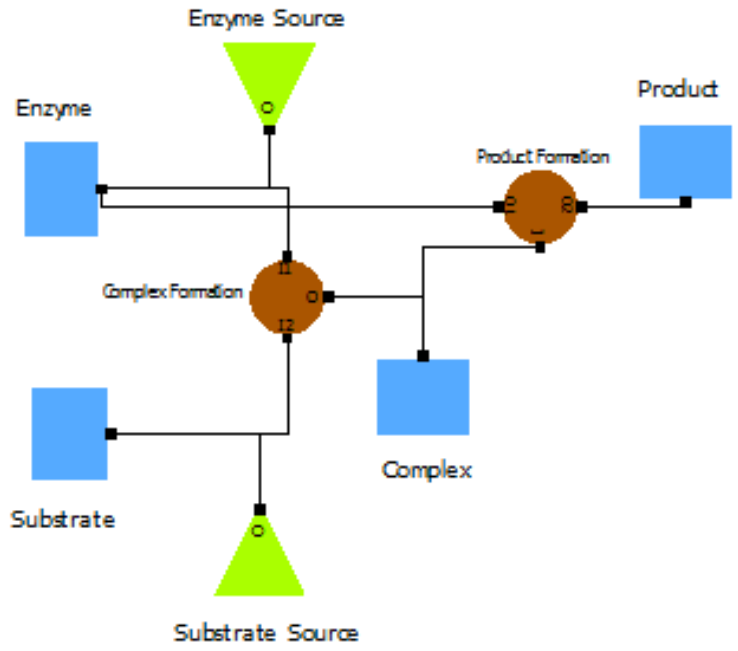
```
BasicBio.Compound compound_enzyme(Q0 = 10, k_deg = 0)
```

- From the simulator's (Plotting) Variable Browser



Set the values and resimulate

Assignment 2 – cheat sheet



Initial substrate concentration = 100

Substrate degradation rate = 0

Initial enzyme concentration = 10

Enzyme degradation rate = 0

Initial complex concentration = 0

Complex degradation rate = 0

Initial product concentration = 0

Product degradation rate = 0

Output flux of substrate source = 0

Output flux of enzyme source = 0

Reversibility of Product Formation reaction = 0 (kB=0)

Systems Biology (SysBio) Library

Background

- Metabolic networks

- Steady State Analysis

- Normalised concentrations

Can be used to perform the simulations for metabolic networks even in the case of very sparse experimental data.

Basic SysBio object classes

Enzyme catalysed reaction: `ESReaction`

Substrate source: `Source`

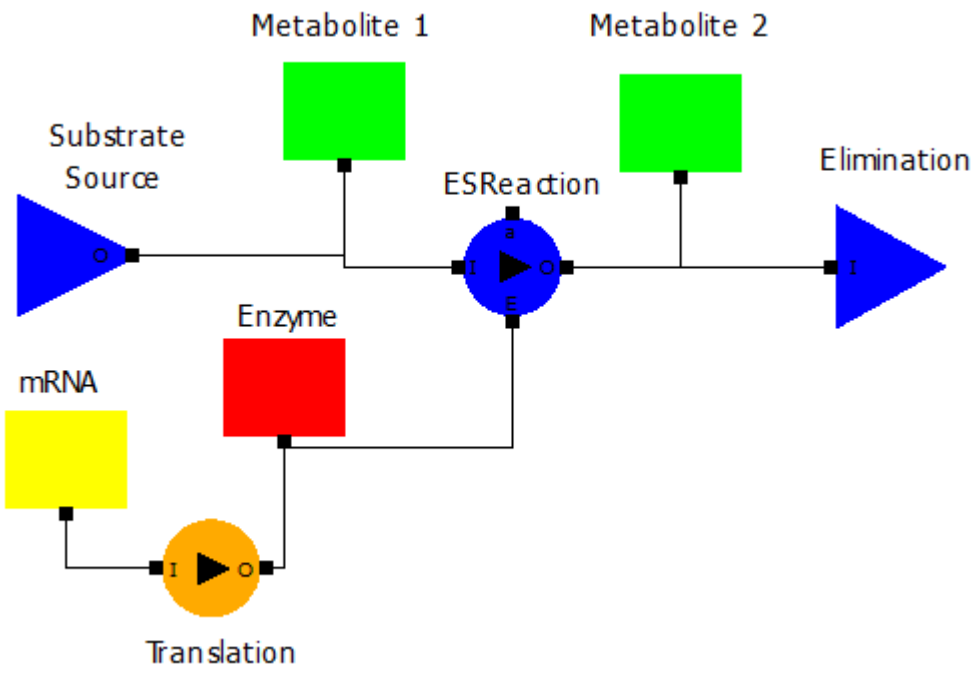
Compounds: `Metabolite, Enzyme, mRNA, Protein`

Transcription regulation:
`GeneExpressionControl(_positive,_negative)`

Translation: `EFormation_lin`

Post-translation regulation: `Activation, Inhibition`

Elimination: `NElimination`



SysBio documentation and examples

Documentation, library derivations and examples available:

- http://lrss.fri.uni-lj.si/bio/sysbio/documentation.html
- http://lrss.fri.uni-lj.si/bio/sysbio/downloads.html

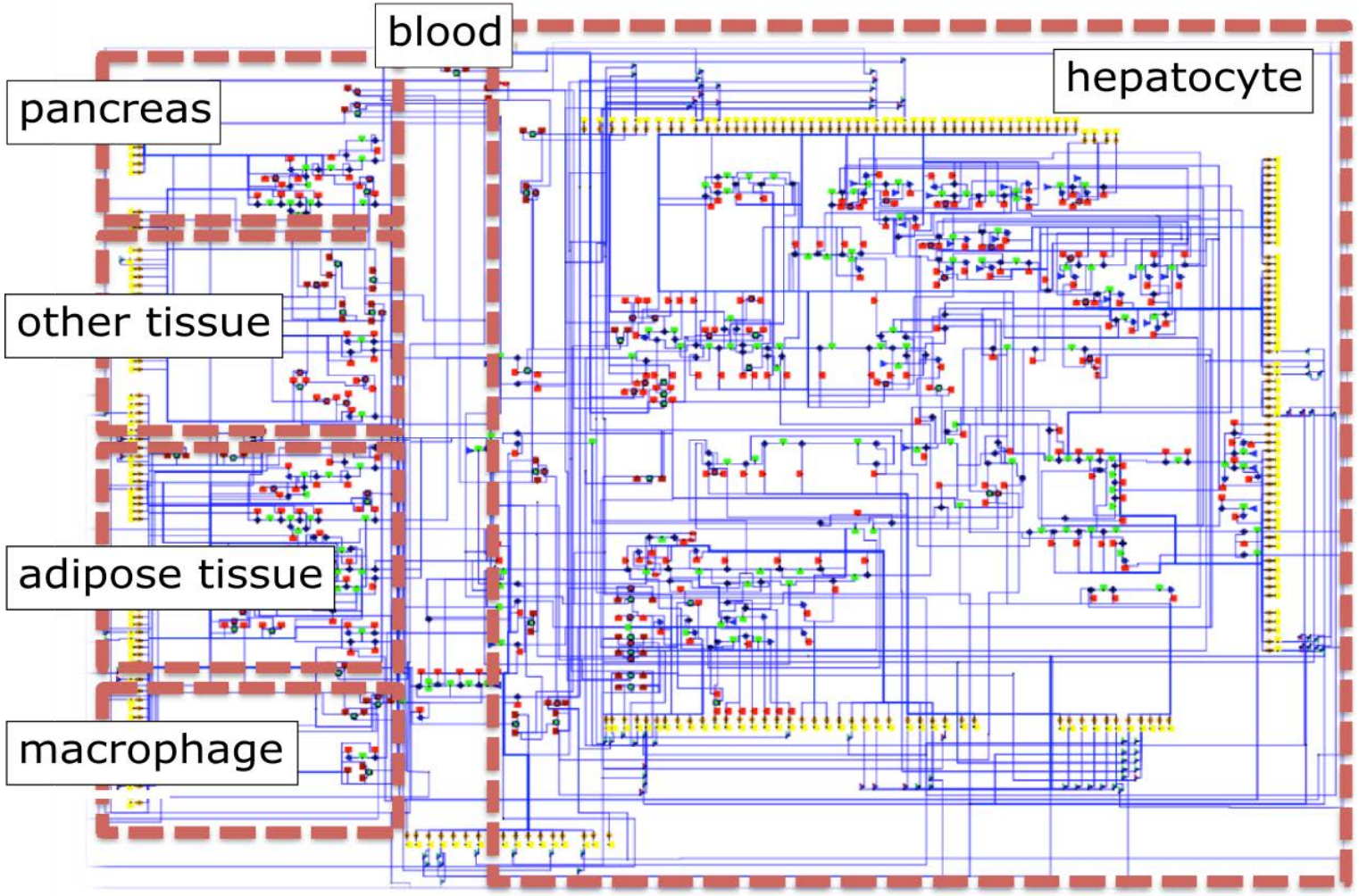
- IBD4Health examples
- Basic SysBio usage example
- SteatoNet
- Cholesterol synthesis pathway

Case study: effects of different diets on hepatic lipogenesis

SteatoNet

Human metabolic model with multi-layered regulation.

Developed to investigate liver-associated pathologies, such as non-alcoholic fatty liver disease.



A. Naik, D. Rozman, and A. Belič, ***SteatoNet*: The first integrated human metabolic model with multi-layered regulation to investigate liver-associated pathologies**, *PLOS Computational Biology*, vol. 10, no. 12, e1003993, 2014

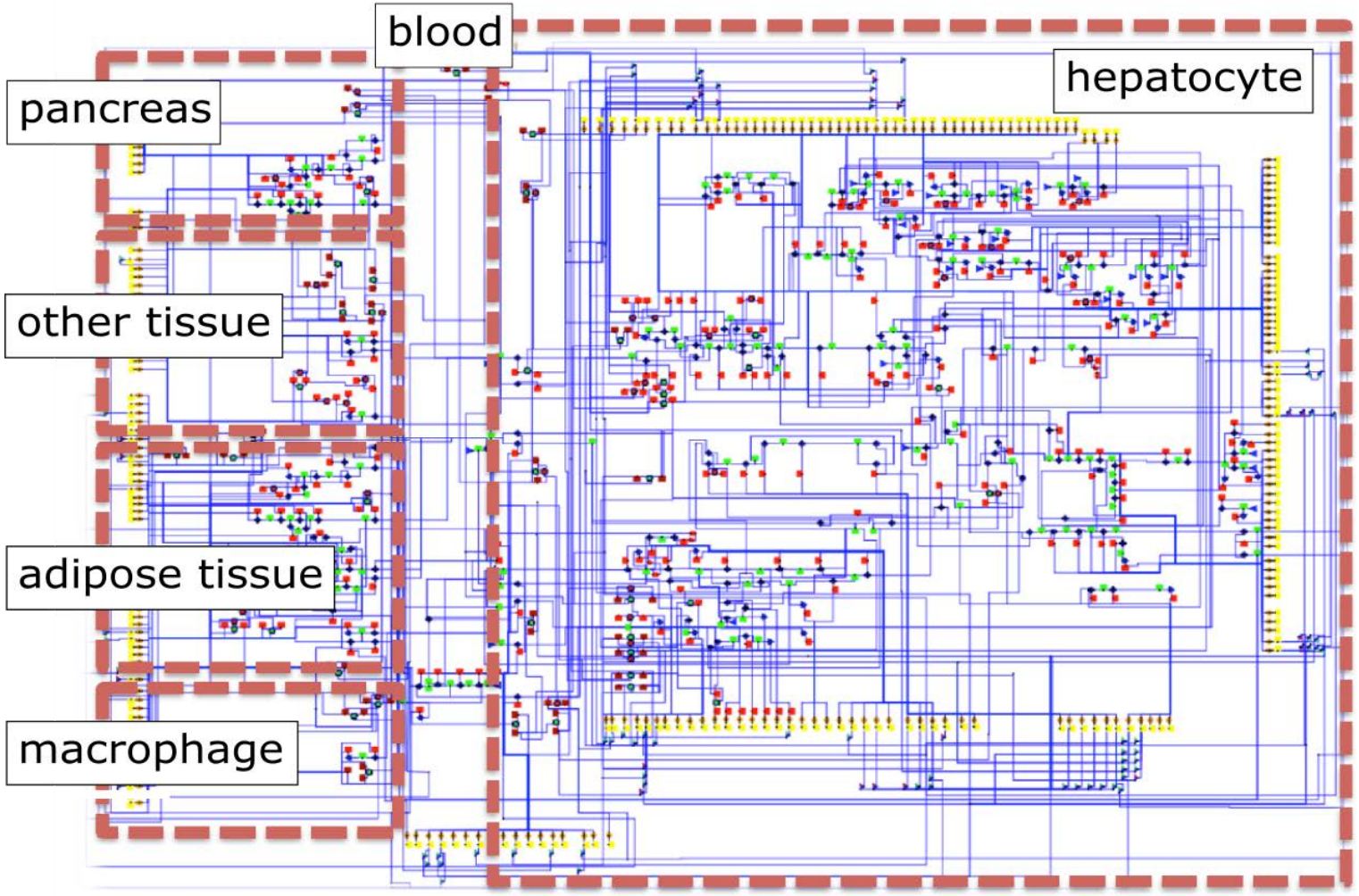
SteatoNet

193 reactions

- 159 metabolites
- 224 enzymes
- 31 regulatory proteins

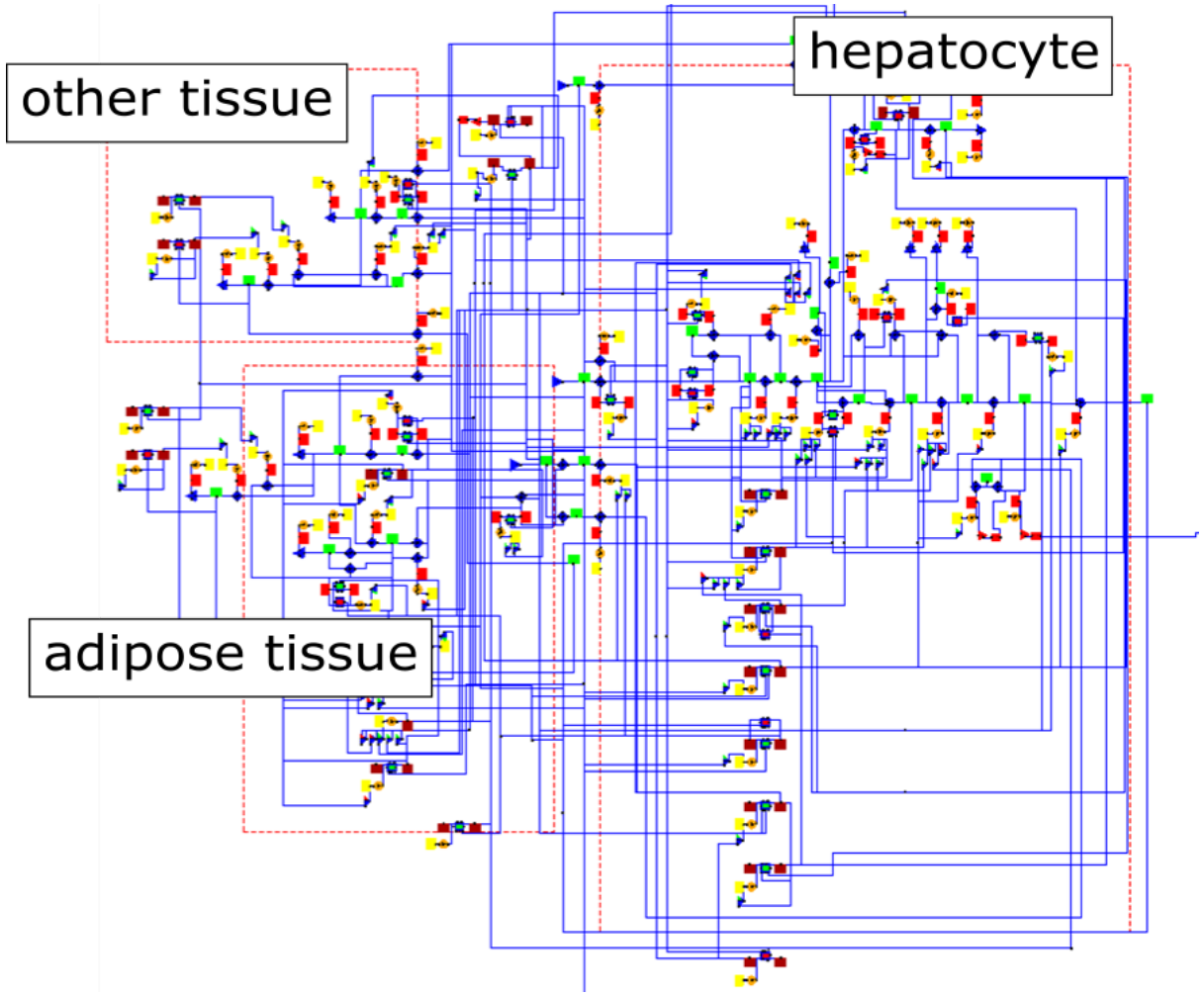
inter-tissue metabolite transport

regulation at transcriptional and post-translation level



A. Naik, D. Rozman, and A. Belič, ***SteatoNet*: The first integrated human metabolic model with multi-layered regulation to investigate liver-associated pathologies**, *PLOS Comp Biol*, vol. 10, no. 12, e1003993, 2014

SteatoNet Reduced



Case study - background

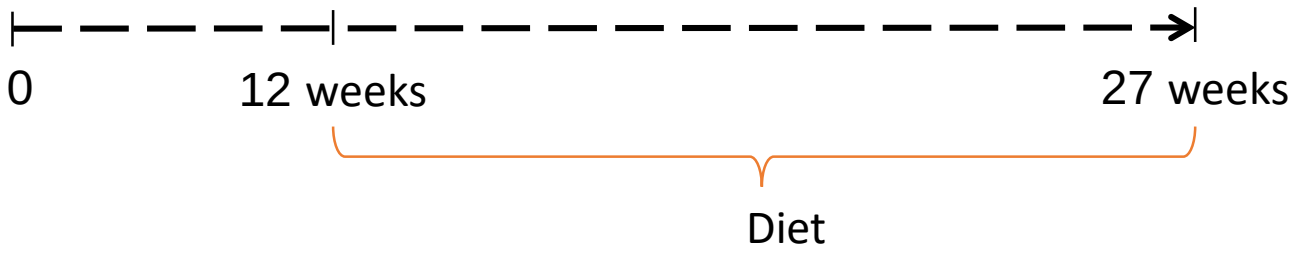
The consumption of simple carbohydrates such as fructose and sucrose is correlated with the development of various diseases, including obesity and insulin resistance/type II diabetes.

Insulin resistance contributes to the presence of the intra-hepatic fat by signalling for de novo lipogenesis, resulting in nonalcoholic fatty liver disease – NAFLD

NAFLD may lead to nonalcoholic steatohepatitis (NASH), cirrhosis and hepatocarcinoma.

A. Schultz, S. Barbosa-da-Silva, M. B. Aguila, C. A. Mandarim-de-Lacerda, **Differences and similarities in hepatic lipogenesis, gluconeogenesis and oxidative imbalance in mice fed diets rich in fructose or sucrose**, *Food Funct*, vol. 6, no. 5, 1684-91, 2015

Case study - background



(a) **Standard chow,** SC (76% carbohydrates- source: corn starch).

(b) **High-fructose diet**, HFru (50% fructose).

(c) **High-sucrose diet**, HSu (50% sucrose).

High sucrose and *high fructose* groups showed significant increase than *standard chow* groups in the synthesis of fatty acids in liver.

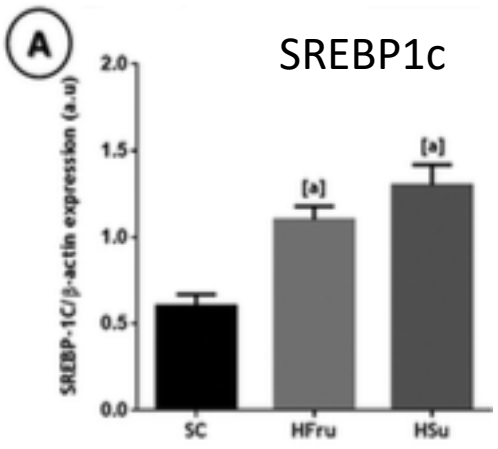
A. Schultz, S. Barbosa-da-Silva, M. B. Aguila, C. A. Mandarim-de-Lacerda, **Differences and similarities in hepatic lipogenesis, gluconeogenesis and oxidative imbalance in mice fed diets rich in fructose or sucrose**, *Food Funct*, vol. 6, no. 5, 1684-91, 2015

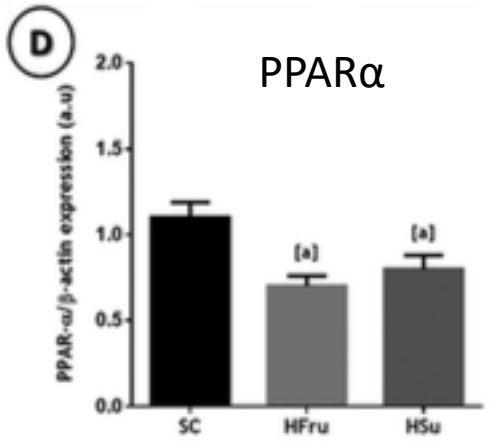
Case study

Observe the difference between normal diet and sucrose (using glucose source) rich diet on the steady state concentrations of

- SREBP-1c: regulates genes required for glucose metabolism and fatty acid and lipid production; its expression is regulated by insulin.

- PPAR-α: a transcription factor and a major regulator of lipid metabolism in the liver.





Hands-on Example: `CaseStudy/SteatoNet.mo`

Case study: model perturbation

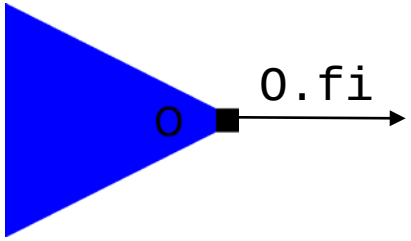
Non-perturbed model corresponds to a normal diet.

Model perturbation:

- Source `Glucose_source`
- Increased source output corresponds to glucose rich diet.

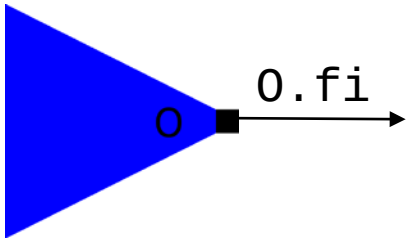
`Glucose_source` belongs to an object class `Source2`:

- can be used to simulate perturbations
- `switchtime1` and `massflow2`: `O.fi =` $\begin{cases} \texttt{massflow1; time < switchtime1} \\ \texttt{massflow2; time >= switchtime1} \end{cases}$



Case study: model perturbation

Initial state corresponds to a normal diet.



Glucose rich diet:

- source `Glucose_source`
- set `massflow2` to `10.0` and `switchtime1` to `5000`

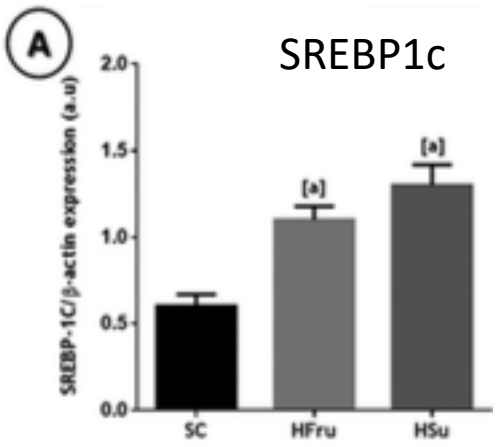
Simulation duration: `60000`

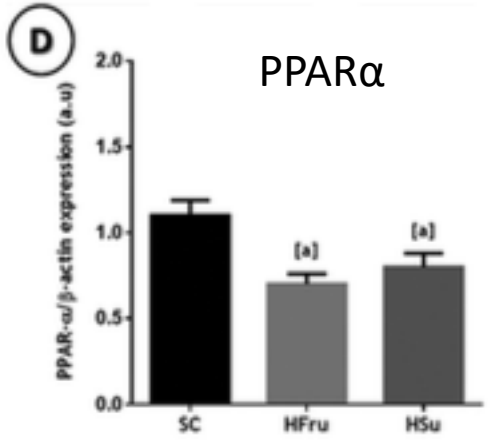
$$\texttt{O.fi} = \begin{cases} 1; & \texttt{time} < 5000 \\ 10; & \texttt{time} >= 5000 \end{cases}$$

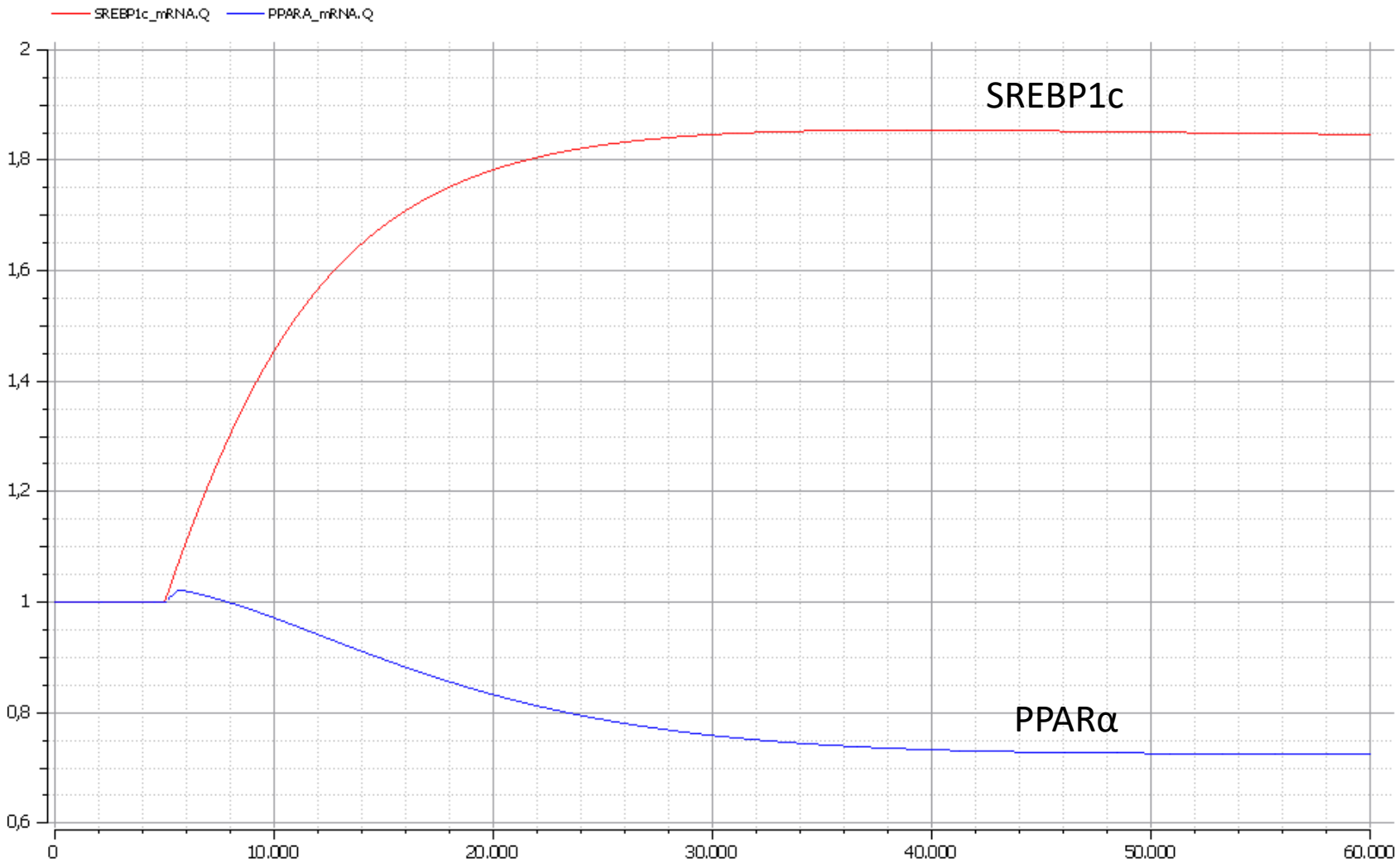
Observe the concentrations (Q) of:

- `SREBP1c_mRNA`
- `PPARA_mRNA`

Case study: results







Case study: conclusion

Results do not correspond to actual concentrations (normalisation) and actual time response (steady state presumption).

In-silico results have the same trends as experimental results.

Predicting the consequences of perturbations, forming novel hypotheses.

More accurate results would require model extension (*SteatoNet* currently only includes glucose source).