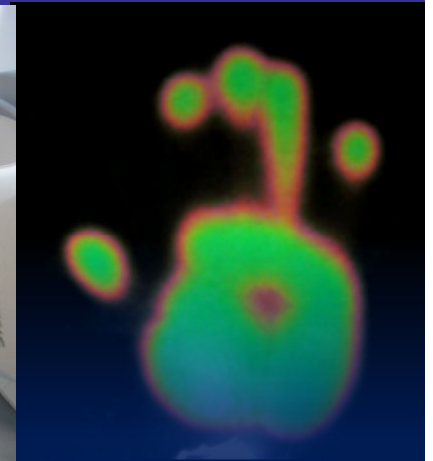
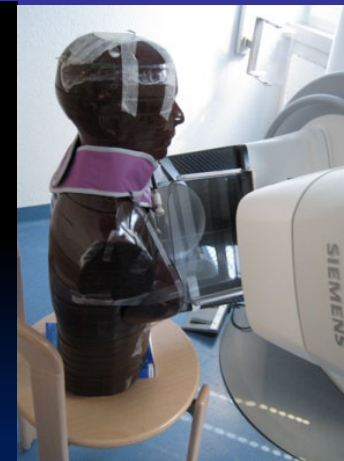
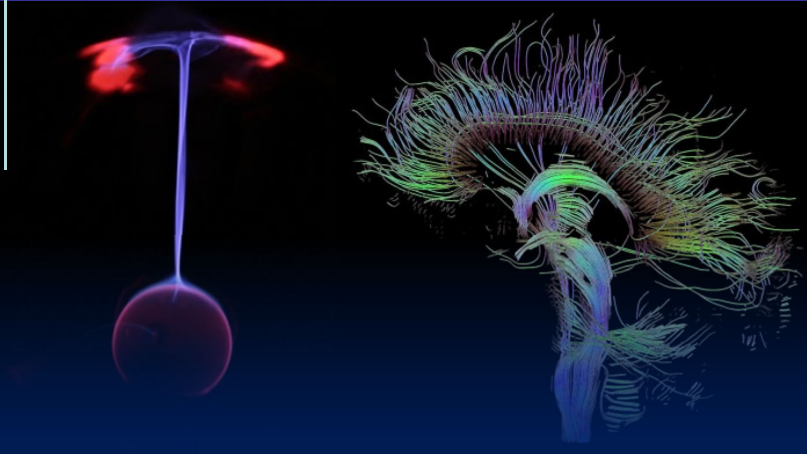
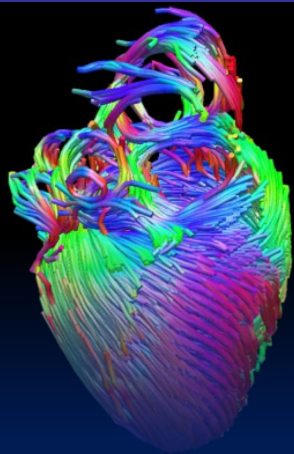


# Using Graphical Model Editors

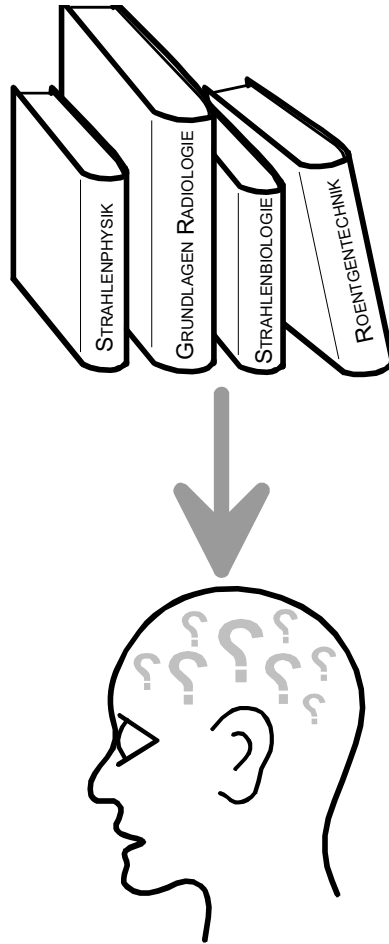
Hyperboost Training Course Model-based  
Data Analysis for Clinical Applications

Stephan Scheidegger  
Medical Biophysics Group ZHAW  
2024



# CONTENT MBDA

---



Model-based data analysis for clinical application – Modelling and Biological Systems:

Day 1

0920-1100: [Modelling and Biological Systems](#)

**1320-1400: Using Graphical Model Editors**

1400-1450: Using Python for model fitting

Day 2

1110-1200: Biokinetic / Biodynamic Modelling (→ Lab2: Model-based Data Analysis of PET Images)

Day3

0900-1100: Radiobiological Models

# Learning Objectives

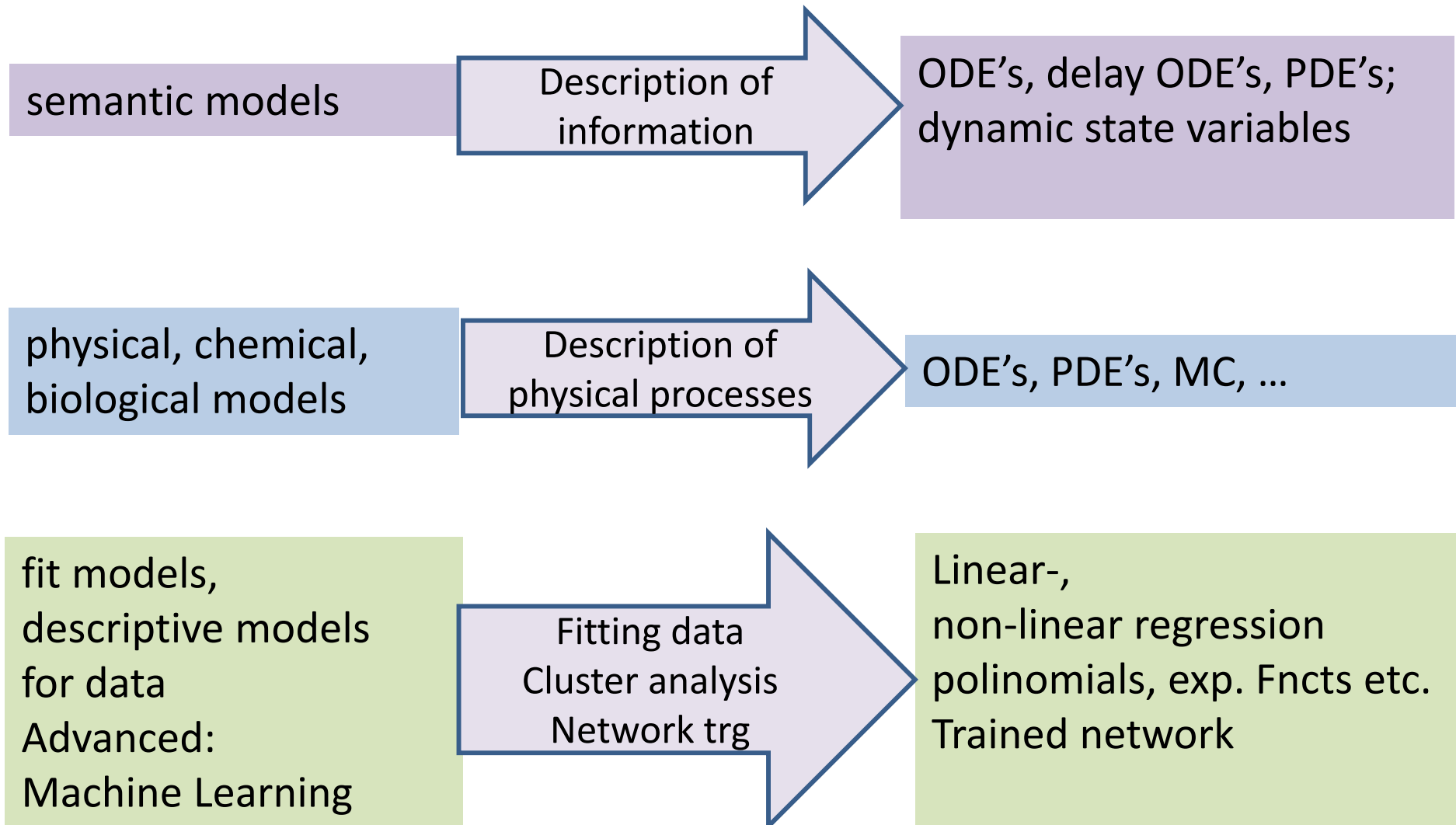


Students are able

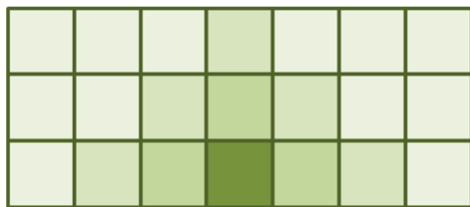
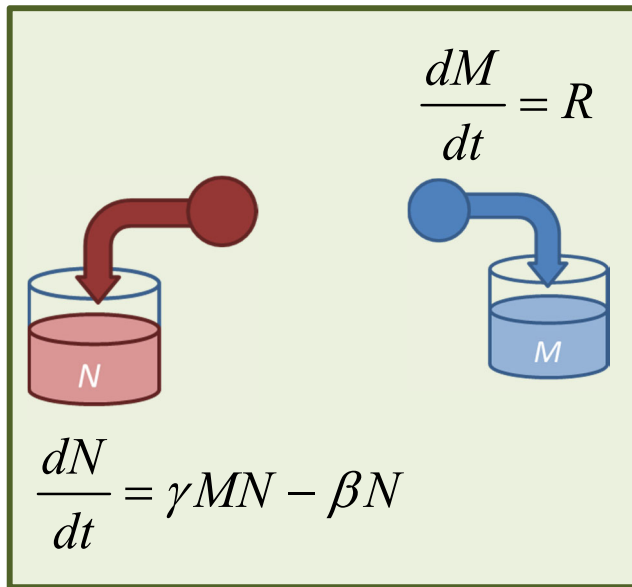
- To be aware of the different purposes of modelling
- to explain the assumption for compartmental models
- to model compartmental biological systems and explore them by using computer simulations
- to use models for biological data analysis
- To use modelling and computer simulation as in silico lab tools

# Catching the Real World in Models

---



# Modelling Approaches and Techniques



$$\frac{dn}{dt} = \gamma mn - \beta n + \kappa \cdot \left( \frac{\partial^2 n}{\partial x^2} + \frac{\partial^2 n}{\partial y^2} \right) \pm \dots$$

Depending on the purpose, different modelling approaches can be useful:

- Compartmental models:
  - Mathematical description by ordinary differential equations (ODE) or delay differential equations (DDE);
  - simulation using finite difference methods
- Spatio-temporal models:
  - Mathematical description by partial differential equations (PDE),
  - simulation using time-domain finite difference (TD-FD) methods or finite elements methods (FEM)

# Modelling and Computer Simulation

---

$$\frac{dN}{dt} = f(N, t)$$

$$\frac{\Delta N}{\Delta t} = f(N, t)$$

$$\Delta N = f(N, t) \cdot \Delta t$$

$$N(t) = N(t - \Delta t) + \Delta N(t - \Delta t) = \\ N(t - \Delta t) + f(N(t - \Delta t), t - \Delta t) \cdot \Delta t$$

To perform “biological experiment” in silico, numerical solving of the model equations is needed (in case of DE-based models):

- numerical integration using Euler’s or Runge-Kutta Method for ODE
- Time-Domain Finite Difference (TD-FD) Method for PDE
- Finite Element Method (FEM) for PDE
- Deep Learning

# Modelling and Computer Simulation

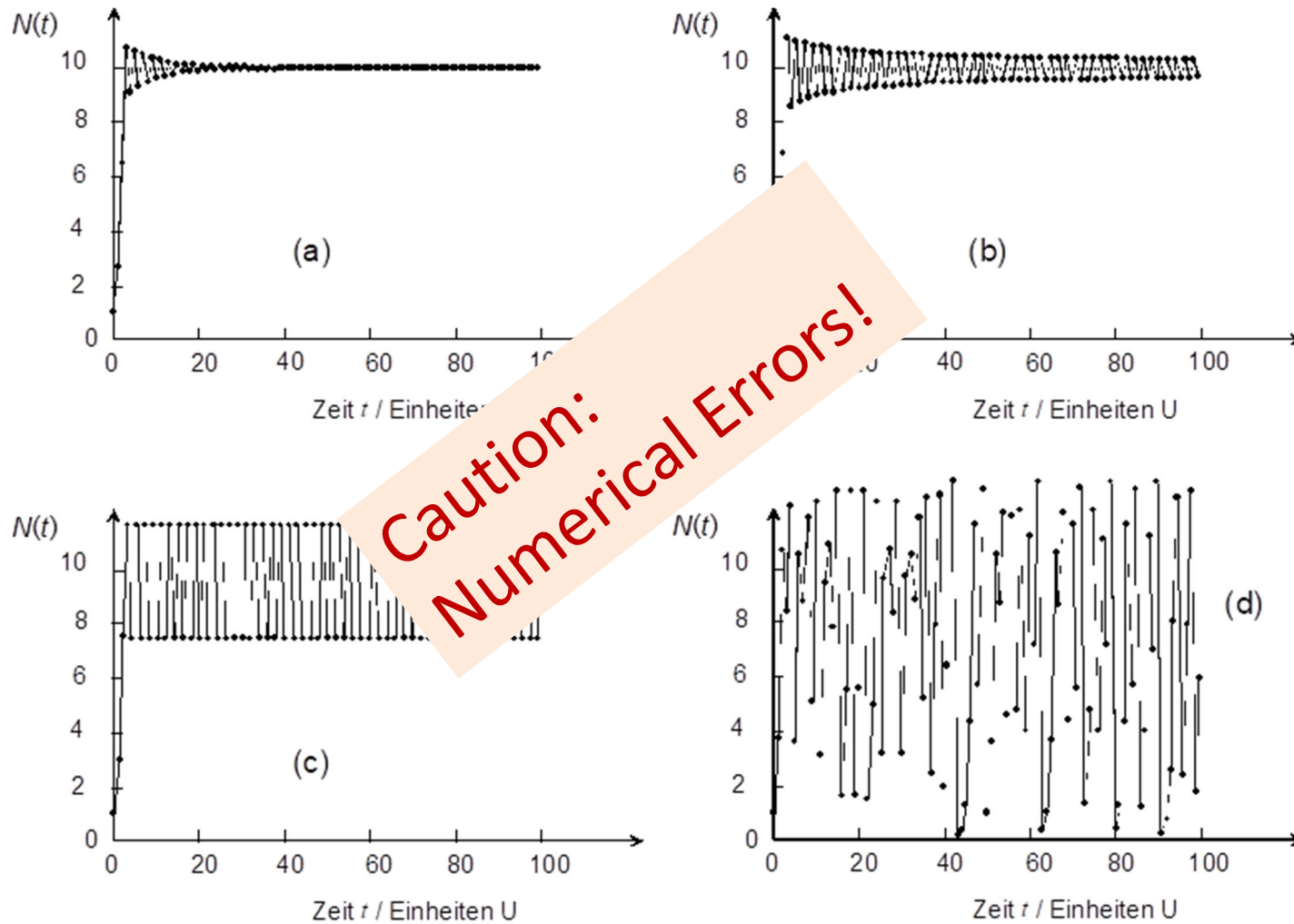
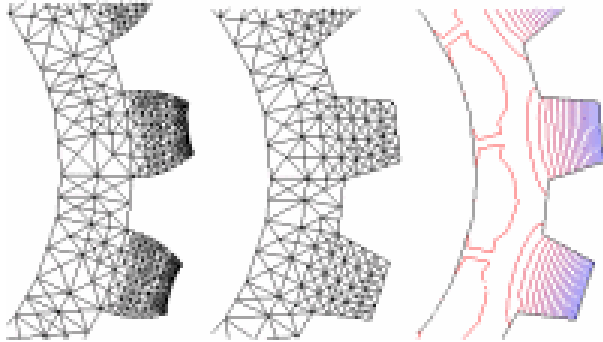


Fig.1. Numerical oscillations in the system  $\frac{dN}{dt} = \alpha N - \beta N^2$

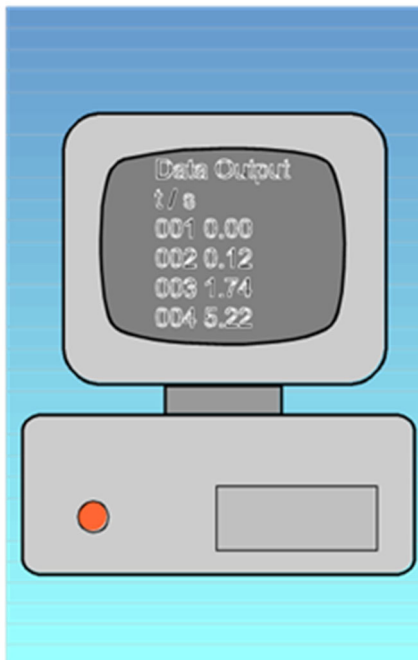
# Modelling and Computer Simulation

---



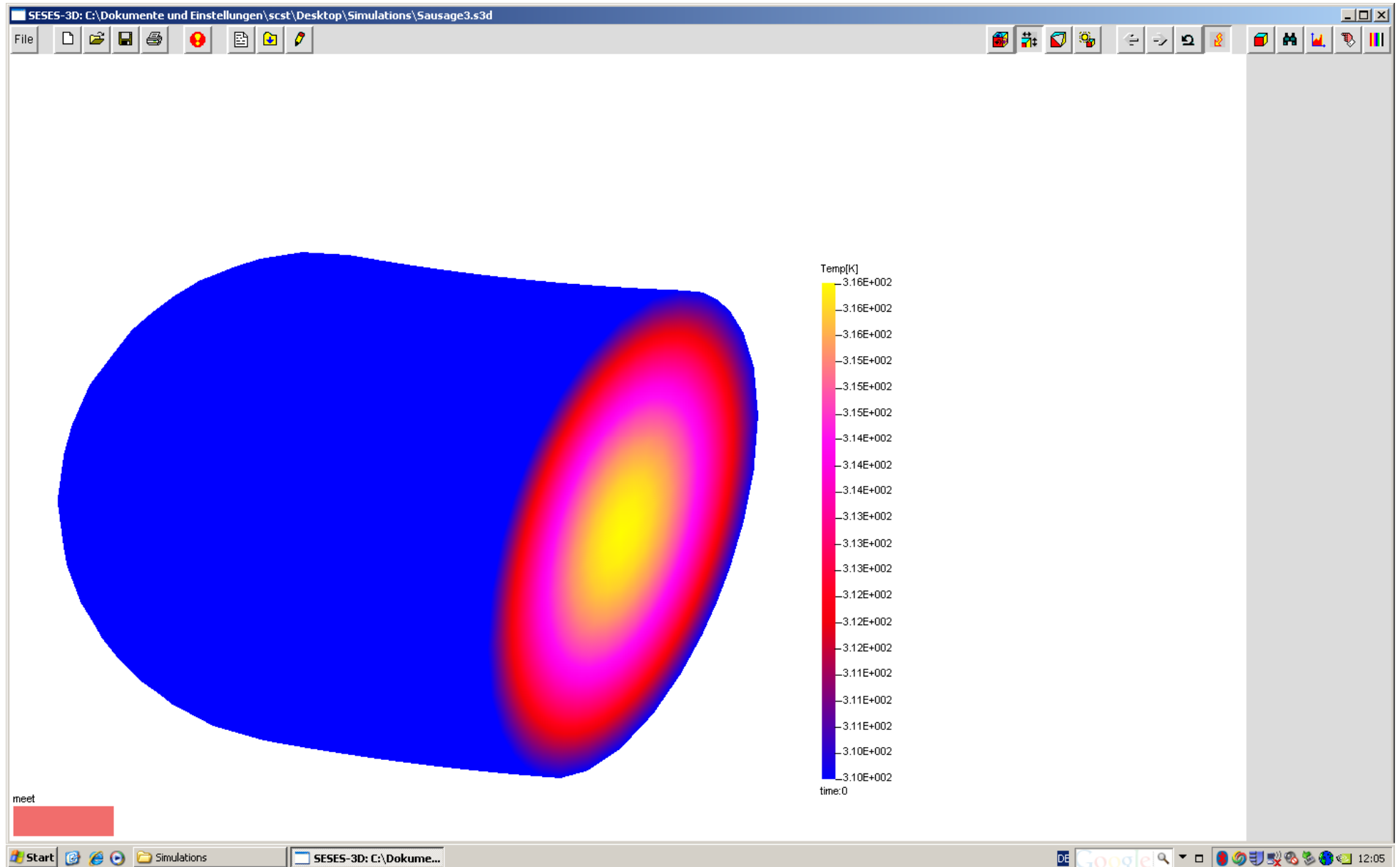
Different tools for implementation are available:

- (high level) programming languages without or with libraries (e.g. Python, Matlab, Octave, ...)
- Graphical modelling tools; generation of a mark-up language code which can be feeded into a solver (e.g. Systems Biology Mark-up Language SBML; M. Hucka et al.: The Systems Biology Markup Language (SBML: A medium for representation and exchange of biochemical network models. In: *Bioinformatics*. Vol. 19, no. 4, 2003, S. 524–531)





# Modelling and Computer Simulation: FEM Multi-Physics Tools



# Modelling and Computer Simulation: Spyder (Python Editor)

The screenshot displays the Spyder Python IDE interface. The main window is titled "Spyder (Python 3.4)" and contains several panes:

- Editor:** Shows a Python script named `ball_batch.py` with the following code:

```
26 dt=0.0005
27 z=2000000
28 n=3
29
30 for j in range(0,n):
31
32     v=v0
33     h=h0
34     tim=0
35     time=0
36     y=y+j*20
37     tim = 0
38     time = 0
39
40     for i in range(0,z):
41         s=h
42         tim=time
43         if h>rd:
44             u=v+(-g-(cw*rh*A/(2*m))*v*sqrt(v*v))*dt
45             velocity.append(u)
46             h=s+v*dt
47             position.append(h)
48             time=tim+dt
49             t.append(time)
50             v=u
51             s=h
52         else:
53             dz=dt/10000
54             u=v+(-g-y*v+D*(rd-s))*dz
55             velocity.append(u)
56             h=s+v*dz
57             position.append(h)
58             time=tim+dz
59             t.append(time)
60             v=u
61             s=h
62
63     plot(t, position)
64     xlabel('t/s')
```
- Object inspector:** Displays a "Usage" message:

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Object Inspector*.

[New to Spyder? Read our tutorial](#)
- IPython console:** Shows the IPython prompt and output:

```
Python 3.4.3 [Anaconda 2.3.0 (64-bit)] (default, Mar  6 2015, 12:06:10) [MSC v.1600 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 3.2.0 -- An enhanced Interactive Python.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
?          -> Introduction and overview of IPython's features.
%quickref  -> Quick reference.
help       -> Python's own help system.
object?    -> Details about 'object', use 'object??' for extra details.
%gui?ref   -> A brief reference about the graphical user interface.

In [1]:
```

The bottom status bar shows: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 30, Column: 1, Memory: 17 %.

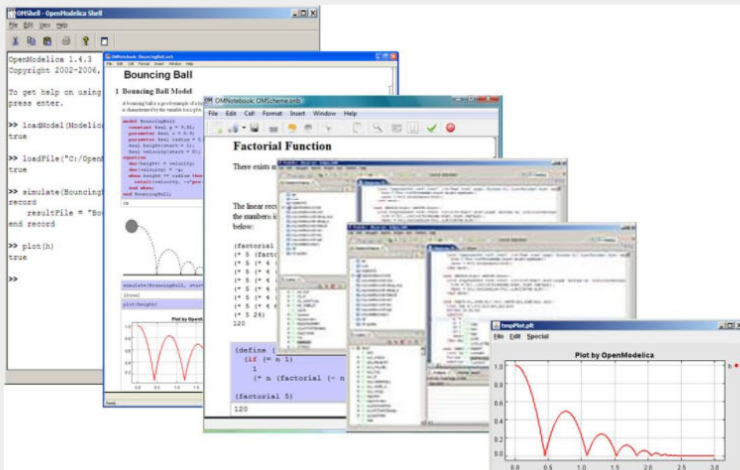
# OpenModelica

Home Download Users Developers Events Research

## Introduction

OPENMODELICA is an open-source Modelica-based<sup>1</sup> modeling and simulation environment intended for industrial and academic usage. Its long-term development is supported by a non-profit organization – the [Open Source Modelica Consortium \(OSMC\)](#). An overview journal [paper](#) is available and [slides](#) about Modelica and OpenModelica.

The goal with the OpenModelica effort is to create a comprehensive Open Source Modelica modeling, compilation and simulation environment based on free software distributed in binary and source code form for research, teaching, and industrial usage. We invite researchers and students, or any interested developer to participate in the project and cooperate around OpenModelica, tools, and applications.



Join the OpenModelicaInterest [mailing list](#) to get information about new releases.

Help us: get the latest [source code](#) or [nightly-build](#) and report [bugs](#).

To learn about Modelica, read a [book](#) or a [tutorial](#) about [Modelica](#).

Interactive step-by-step beginners Modelica [on-line spoken tutorials](#) Interactive [OMWebbook](#) with examples of Modelica textual modeling and [textbook companions](#) with application OpenModelica exercises. A [Jupyter notebook](#) Modelica mode, available in OpenModelica.

To get advice how to make existing Modelica libraries work in OpenModelica, see [Porting](#).

For systems engineering with requirement traceability and verification, see [ModelicaML](#).

OpenModelica provides [library coverage reports](#) of open-source Modelica libraries showing which libraries work well with OpenModelica and how the support improved over time.

# SBML Systems Biology Markup Language

Session: /home/mkoenig/git/cy3sbml/src/main/resources/sessions/Koenig\_demo\_10.cys

File Edit View Select Layout Apps Tools Help

Control Panel

Network Style Select

Network	Nodes	Edges
▼ Koenig_demo_10		
Koenig_demo_10	36(0)	69(0)
Main: Koenig_demo_10	13(0)	14(0)
▼ Koenig_demo_10		
Koenig_demo_10	36(0)	69(0)
Main: Koenig_demo_10	13(0)	14(0)

Main: Koenig\_demo\_10

Results Panel

Model : Koenig\_demo\_10 (Koenig\_demo\_10)

L3v1

### Koenig Demo Metabolism

Description

This is a demonstration model in SBML format. The content of this model has been carefully created in a manual research effort. This file has been produced by [Matthias Koenig](#).

Terms of use

Copyright © 2016 Matthias Koenig.

Redistribution and use of any part of this model, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of this SBML file must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in a different form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

This model is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Table Panel

shared name	name	id	sbml-type	sbo	metald	biomodels.sbo	go	fma	label	value	units	derivedUnits	constant
external compartment	external com...	e	compartment	SBO:0000...	meta_22d897...	SBO:0000290	GO:0005...	FMA:70022	external com...	1.0E-6	m3	m^3	<input type="checkbox"/>
cell compartment	cell comp...	c	compartment	SBO:0000...	meta_78b0e7...	SBO:0000290	GO:0005...	FMA:68646	cell compar...	1.0E-6	m3	m^3	<input type="checkbox"/>
plasma membrane	plasma m...	m	compartment	SBO:0000...	meta_bcd47...	SBO:0000290	GO:0005...	FMA:63841	plasma me...	1.0	m2	m^2	<input type="checkbox"/>
metabolic scaling fa...	metabolic ...	Km_C	parameter	SBO:0000...	meta_c63c69...	SBO:0000027			Km_C	3.0	mM	mol*m^(-3)	<input checked="" type="checkbox"/>
		scale_f	parameter						metabolic s...	1.0E-6	dimensionless	dimensionless	<input checked="" type="checkbox"/>
		Vmax_BB	parameter	SBO:0000...	meta_871a28...	SBO:0000186			Vmax_BB	2.0	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>
		Vmax_BC	parameter	SBO:0000...	meta_ad898f...	SBO:0000186			Vmax_BC	2.0	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>
		Vmax_BA	parameter	SBO:0000...	meta_351d07...	SBO:0000186			Vmax_BA	5.0	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>
		Vmax_V2	parameter	SBO:0000...	meta_074616...	SBO:0000186			Vmax_V2	0.5	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>
		Vmax_V3	parameter	SBO:0000...	meta_1e29b...	SBO:0000186			Vmax_V3	0.5	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>
		Vmax_V1	parameter	SBO:0000...	meta_78fe37...	SBO:0000186			Vmax_V1	1.0	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>
		Km_A	parameter	SBO:0000...	meta_98f0e1...	SBO:0000027			Km_A	1.0	mM	mol*m^(-3)	<input checked="" type="checkbox"/>
		Vmax_V4	parameter	SBO:0000...	meta_20f045...	SBO:0000186			Vmax_V4	0.5	mole_per_s	mol*s^(-1)	<input checked="" type="checkbox"/>

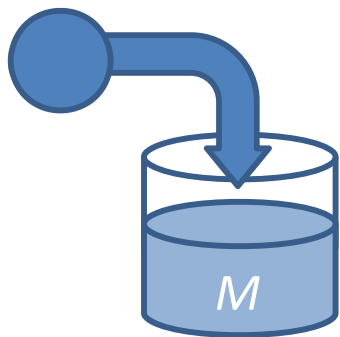
Node Table Edge Table Network Table

Memory

# Modelling and Computer Simulation: Graphical Model-Builder for Compartmental Systems

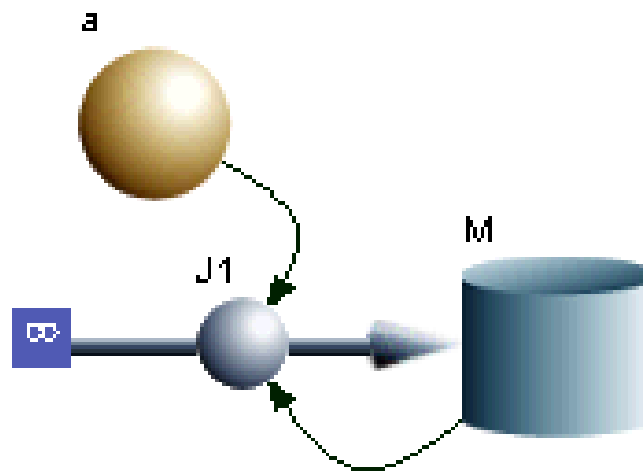
## Semantics

$$\frac{dM}{dt} = R$$



$$M(t) = \int R dt$$

## Graphical Representation (Flow Chart)



## Model Equations (Mark-up language)

{Top model}

{Reservoirs}

d/dt (M) = + J1

INIT M = 0

{Flows}

J1 = a\*M

{Functions}

a = 1

{Globals}

{End Globals}

Graphical modelling editors for compartmental system simulation:

*Berkeley Madonna*

<https://berkeley-madonna.myshopify.com/>

*Vensim*

<https://vensim.com/vensim-personal-learning-edition/>

*Stella*

<https://www.iseesystems.com/store/products/stella-architect.aspx>

Markup code:  
Model equations

Flow chart:  
Graphical model  
representation

Simulation "cockpit":  
Numerical integration procedure  
Time increment  $\Delta t$   
Parameter values

Berkeley Madonna - SIR\_Mod1.mmd (modified)

File Edit Flowchart Model Compute Graph Parameters Window Help Run

SIR\_Mod1.mmd (modified)

{Top model}

```
{Reservoirs}
d/dt (S) = - J1
INIT S = 8700000
d/dt (I) = + J1 - J2
INIT I = 1
d/dt (R) = + J2
INIT R = 0
d/dt (S1) = + J3
INIT S1 = 8700000
d/dt (I1) = + J4
INIT I1 = 1
d/dt (R1) = + J5
INIT R1 = 0
d/dt (cI) = + J6
INIT cI = 0

{Flows}
J1 = a*I*S
J2 = b*I
J3 = -a*I1*S1
J4 = a*I1*S1-b*I1
J5 = b*I1
J6 = J1

{Functions}
a = 0.0000001
b = 0.1
```

Name	Value
STARTTIME	0
STOPTIME *	100
DT *	0.001
DTOUT	0
INIT S	8700000
INIT I	1
INIT R	0
INIT S1	8700000
INIT I1	1
INIT R1	0
INIT cI	0
a	1.0E-7
b	0.1

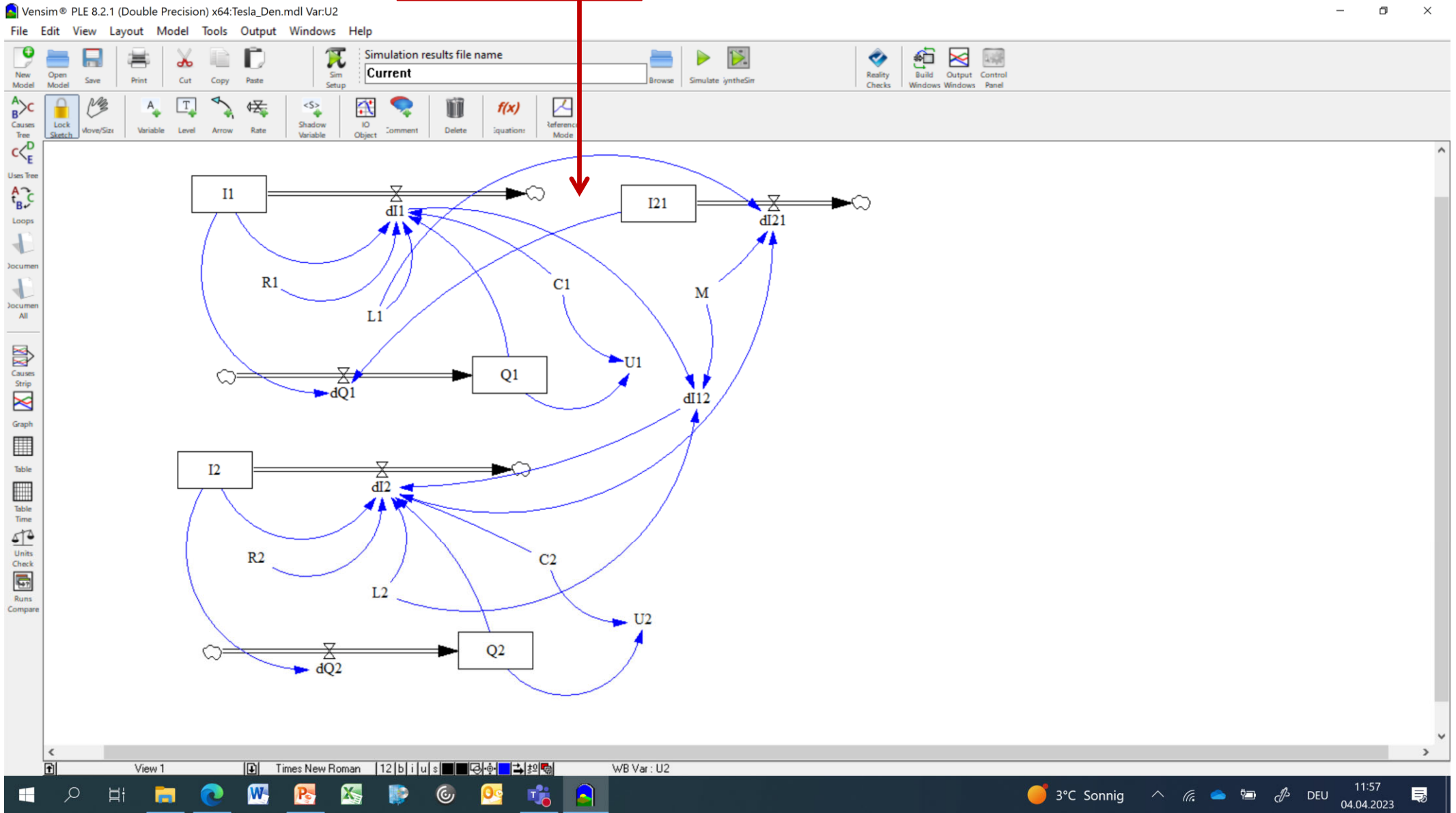
Default parameter set

Parameter set for original equations.

New Edit Del

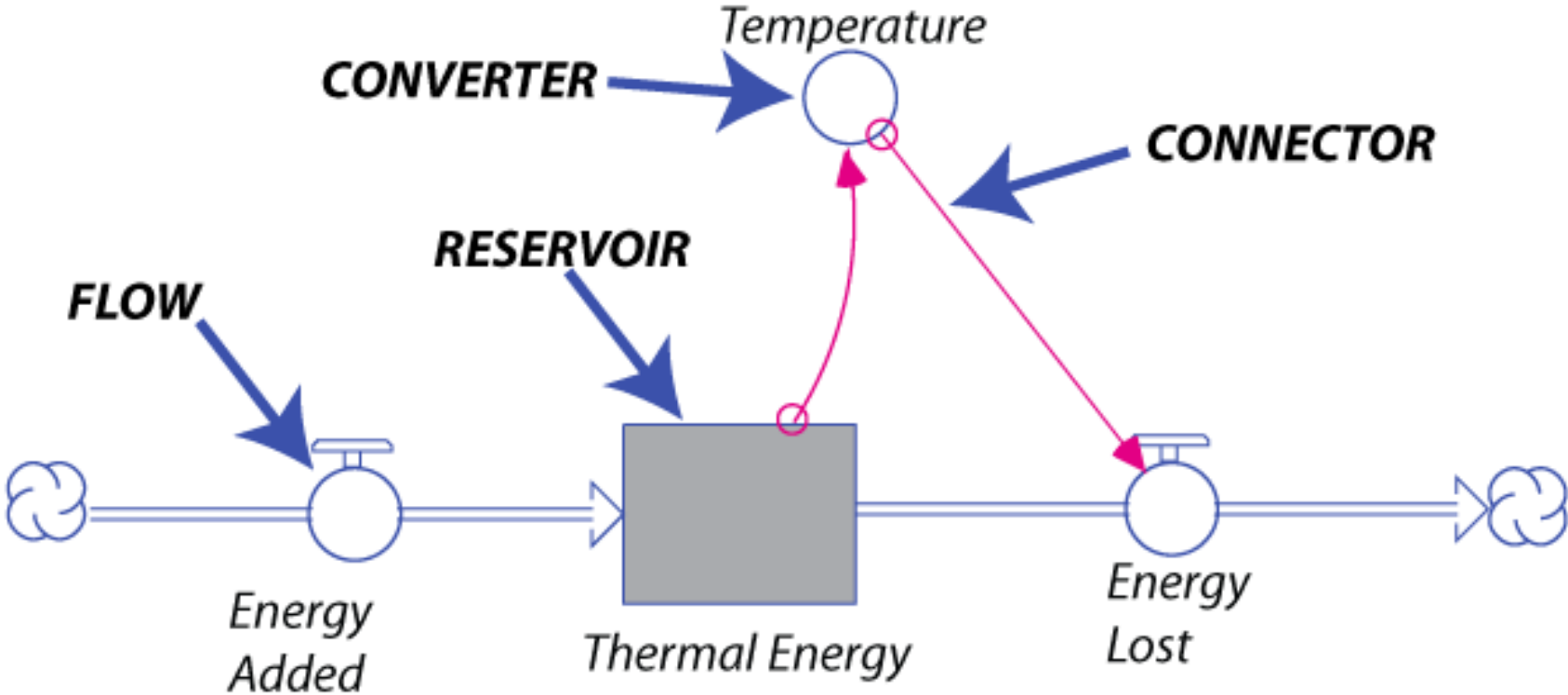
3°C Sonnig 11:46 04.04.2023

Flow chart:  
Graphical model  
representation





# STELLA Model Diagram





moving\_car - Simulink

SIMULATION    DEBUG    MODELING    FORMAT    APPS

FILE    LIBRARY    PREPARE    SIMULATE    REVIEW RESULTS

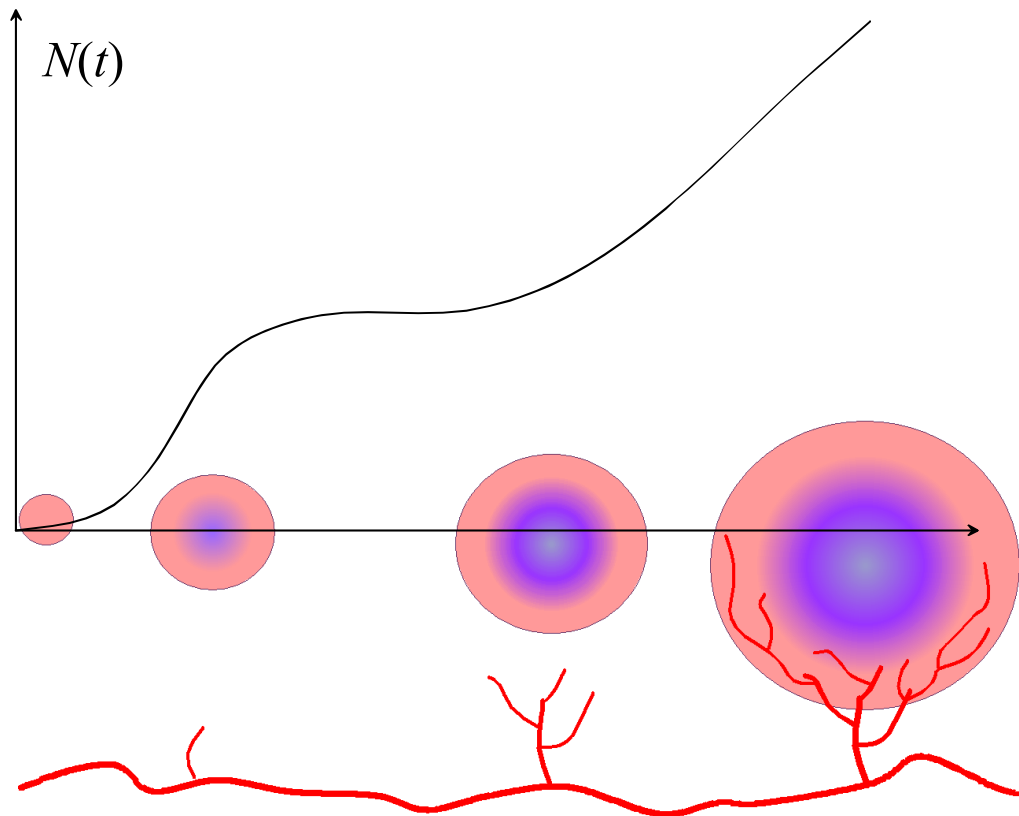
moving\_car

Copyright 2017-2018 The MathWorks, Inc.

Ready    100%    VariableStepAuto

# Control of Gene Expression: Oxygenation and VEGF

---



Vascular Endothelial Growth Factor (VEGF) is expressed in hypoxic tissues.

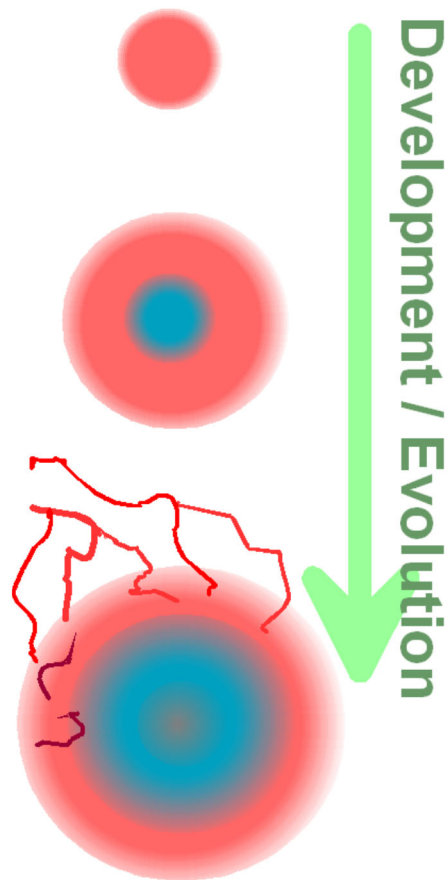
- VEGF expression and secretion lead to activation of vascular endothelial cells
- Result: vessels sprout in direction of the VEGF gradient.

---

Fig.1. Chromosomes during mitosis

# Control of Gene Expression: Oxygenation and VEGF

---

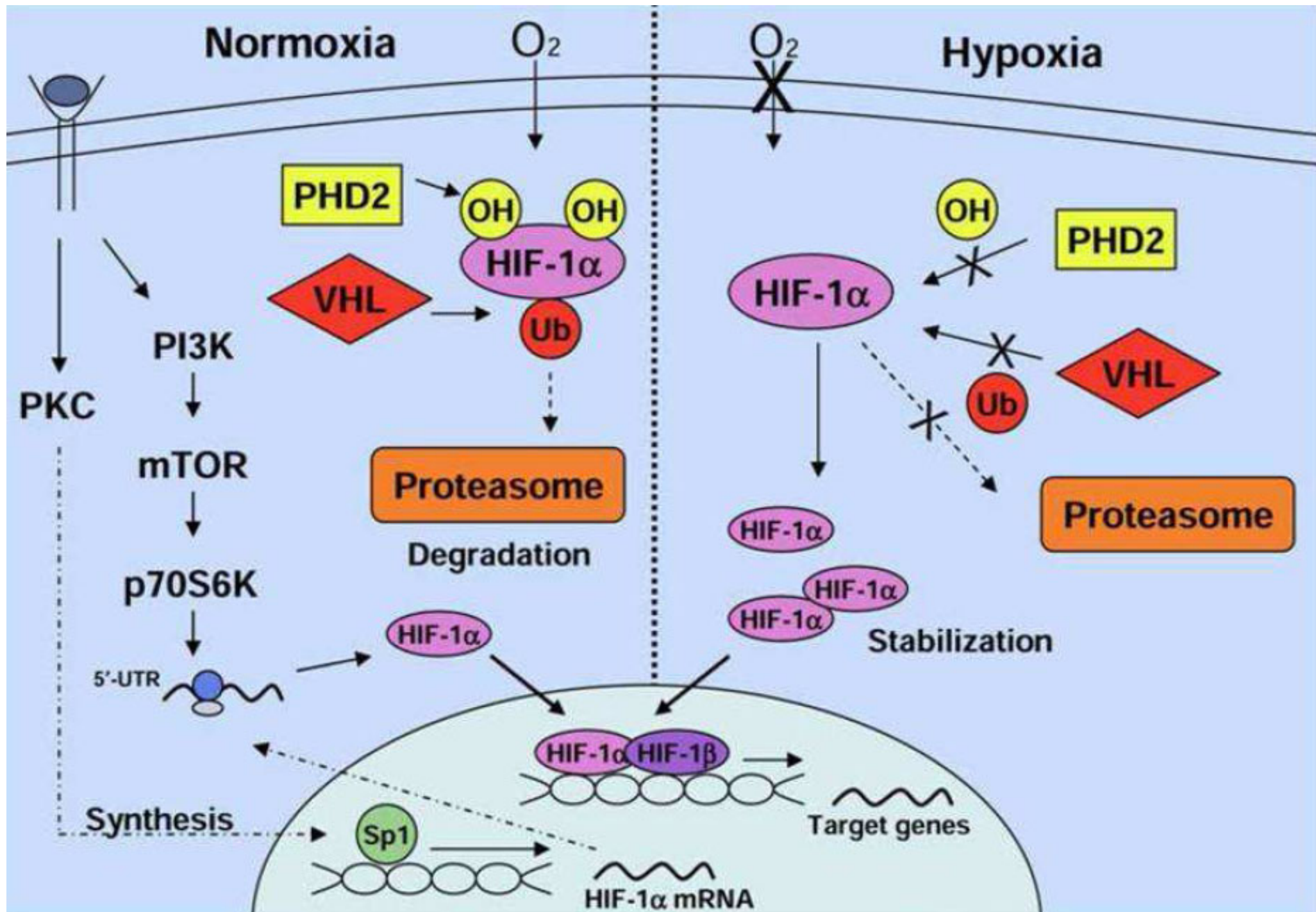


During growth, tumours develop a hypoxic (and subsequently a necrotic) core:

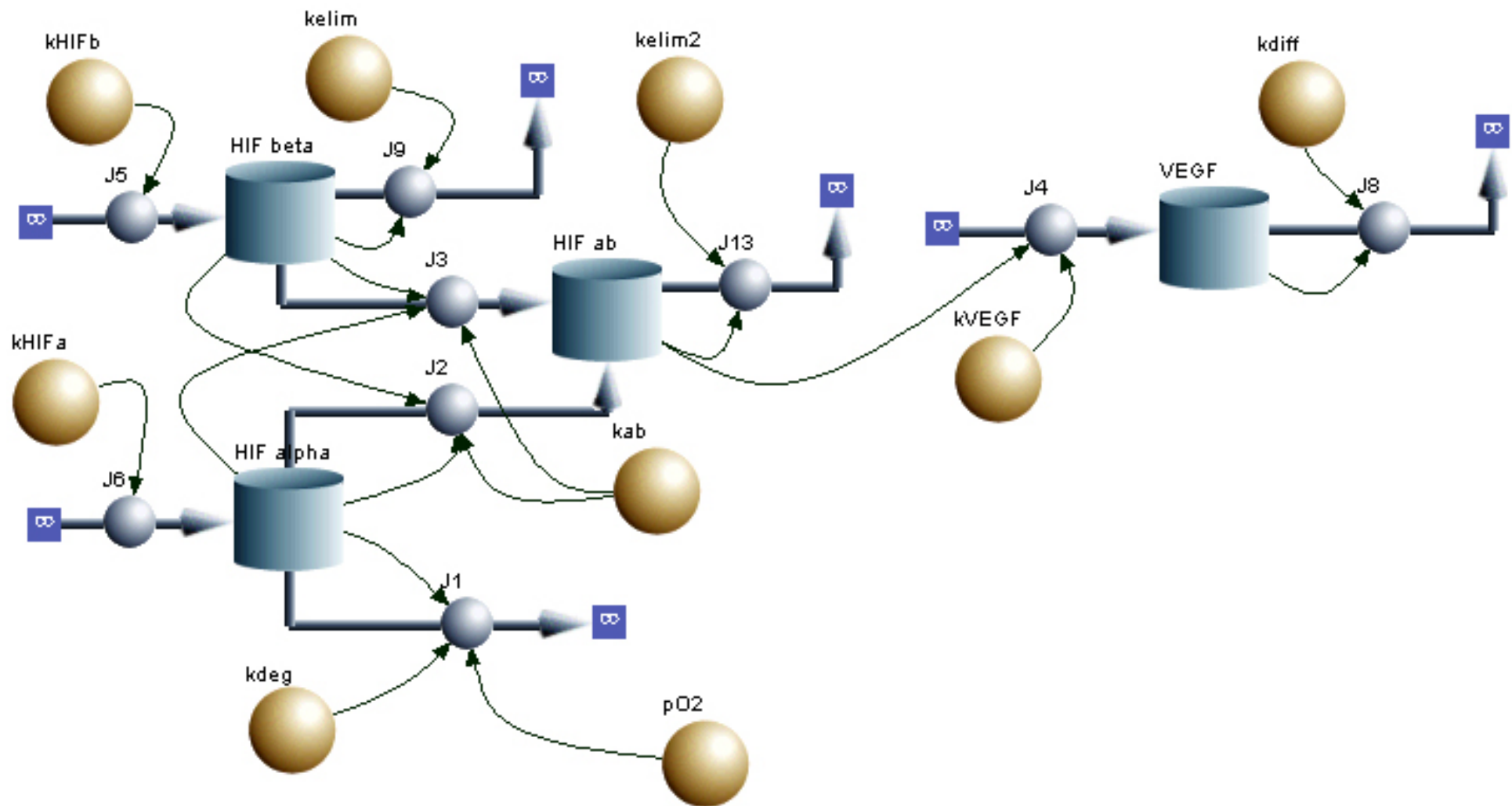
- VEGF release leads to ingrowth of external vessels (angiogenesis)
- Some tumours acquire a vascularized rim
- As a result of the fast growing tumour cell population and lack of tissue organization, tumour vessels often are non-function or leaky.

---

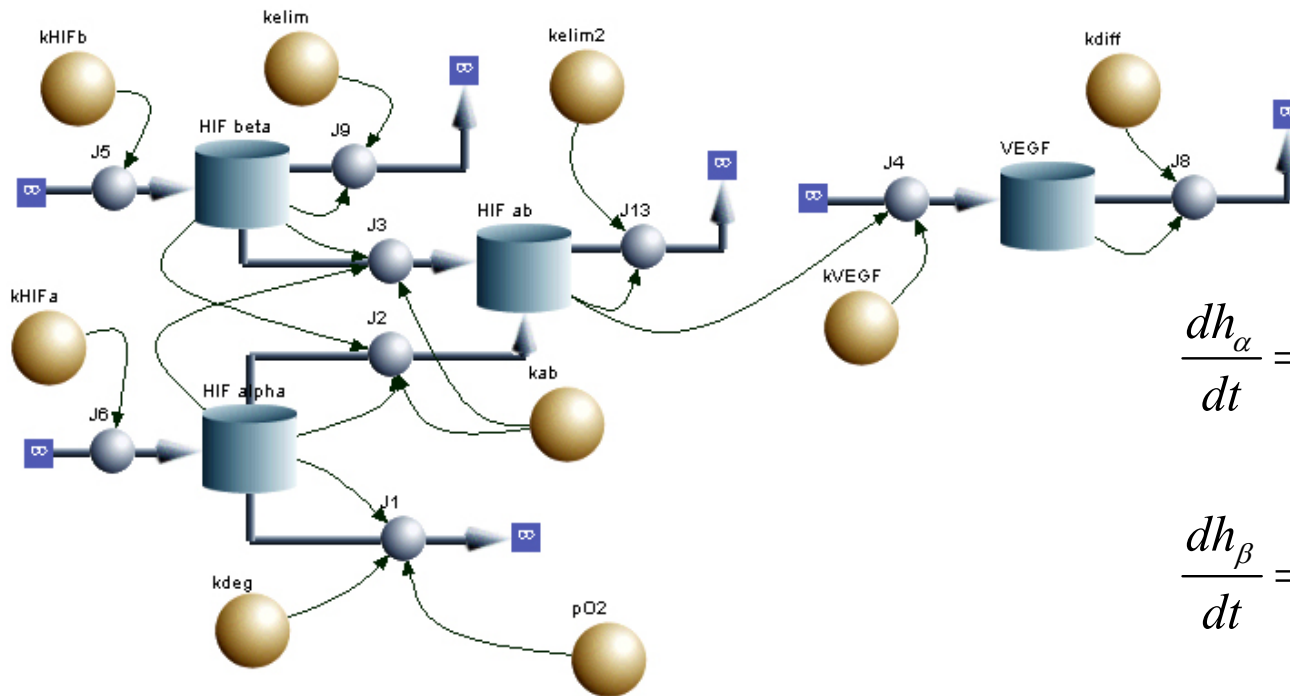
Fig.1. Development of hypoxic and necrotic core during tumour growth



# Control of Gene Expression: VEGF



# Control of Gene Expression: VEGF



$$\frac{dh_{\alpha}}{dt} = k_{\alpha} - k_{O_2} p_{O_2}^2 h_{\alpha} - k_{\alpha\beta} h_{\alpha} h_{\beta}$$

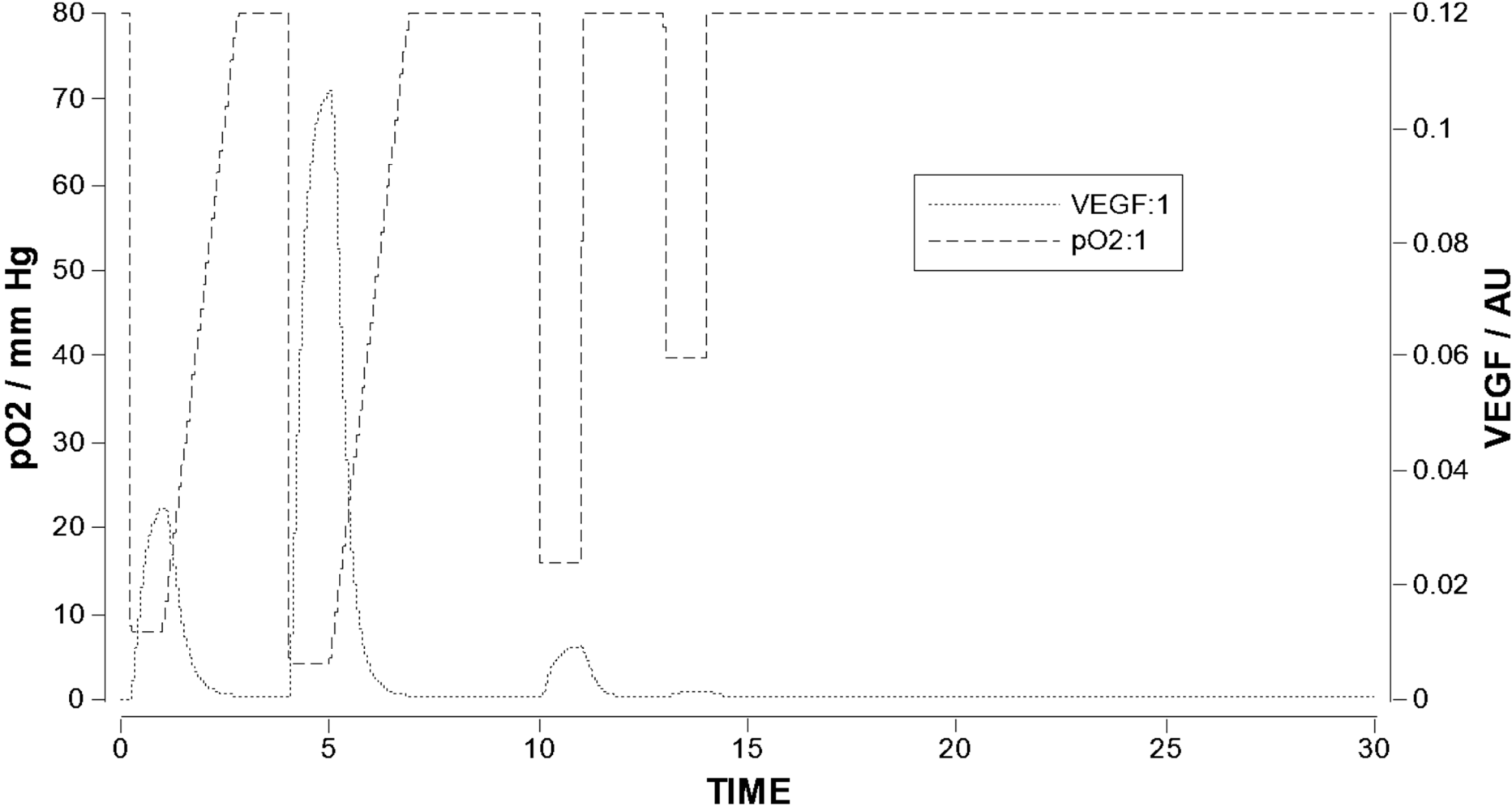
$$\frac{dh_{\beta}}{dt} = k_{\beta} - k_{\alpha\beta} h_{\alpha} h_{\beta} - k_{\beta e} h_{\beta}$$

$$\frac{dh_{\alpha\beta}}{dt} = 2k_{\alpha\beta} h_{\alpha} h_{\beta} - k_{\alpha\beta e} h_{\alpha\beta}$$

$$\frac{dv}{dt} = k_v h_{\alpha\beta} - k_{ve} v$$

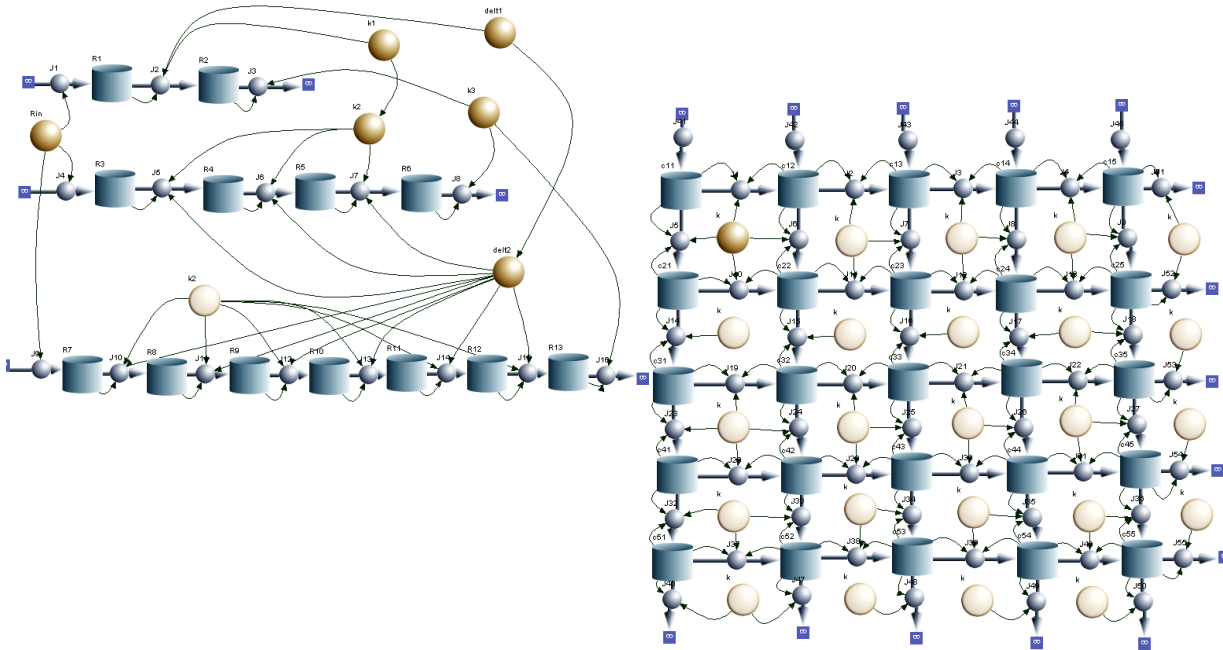
# Control of Gene Expression: VEGF

---



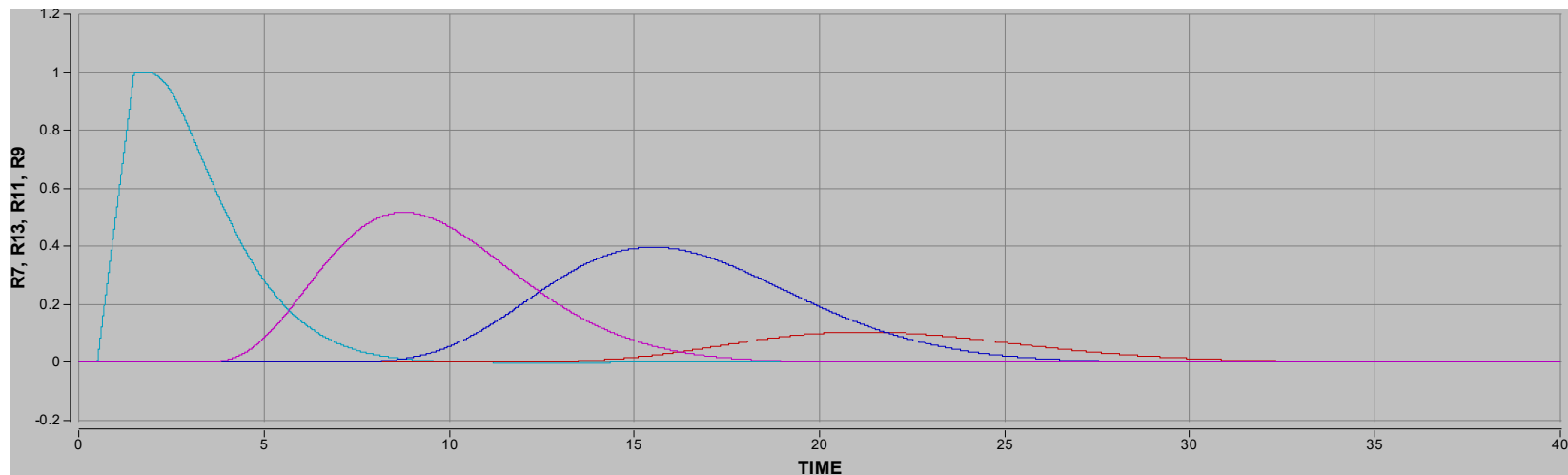


# Modelling Diffusion Networks



## Reaction chains

- Lead to signal dispersion
- Some aspects of complex biological systems may be covered by reaction-chain – diffusion networks



# Compartmental Graphical Model Editors: Pro & Con's

---

## Pro

- Fast programming
- System visualization via flow chart
- Graphically supported modelling process
- Powerful and stable solvers
- Batch run options
- Implemented fitting methods

## Desadvantages:

- Dedicated for compartmental models, therefore limited to models that can be described by ODE's or signaling chains
- Graphical representation for large systems or diffusion chains / waveguides difficult → generative algorithms for writing markup code useful