Tutorial on IQM Tools Pro

MEX / Systems Biology/Pharmacology Projects

Integrated Solutions for Quantitative Drug Development
From Mechanistic Models to Complex Trial Simulation

Henning Schmidt, IntiQuan GmbH

henning.schmidt@intiquan.com

IntiQuan

Tutorial Outline

- **General introduction to the IQM Tools Suite**
- **High performance computation via MEX simulation functions**
- **Model Simulation**
  - Standard
  - Parametric sensitivity
- **IQMprojectSB - parameter estimation, model reduction**
  - Projects
  - Parameter estimation / manual tuning / parameter fit analysis / parameter identifiability analysis
  - Model reduction
  - Simulation of projects
  - Commenting of projects

henning.schmidt@intiquan.com

Tutorial Info

In large parts you will have the opportunity to get hands-on-experience

```
>> installIQMtoolsInitial
```

Commands shown in these boxes should be entered on the MATLAB command line, during the tutorial

```
********** MODEL NAME
Simple model
********** MODEL STATES
d/dt(A) = -R
d/dt(B) = R
A(0) = 1
B(0) = 0
********** MODEL PARAMETERS
k1 = 0.5
********** MODEL REACTIONS
R = k1*A
```

Text shown in these boxes should be entered where appropriate (will become clear later)

henning.schmidt@intiquan.com

Tutorial Goal: „You should be able to“

- Use MEX/C-code models for fast simulation
- Calculate sensitivity trajectories with MEX models
- Set up your own parameter estimation project
  - Model description
  - Measurement data
  - Experiment descriptions
- Perform parameter estimation, identifiability analysis, etc.
- Analyze the resulting model

Tutorial Outline

- **General introduction to the IQM Tools Suite**
- High performance computation via MEX simulation functions
- Model Simulation
- IQMprojectSB - parameter estimation, model reduction

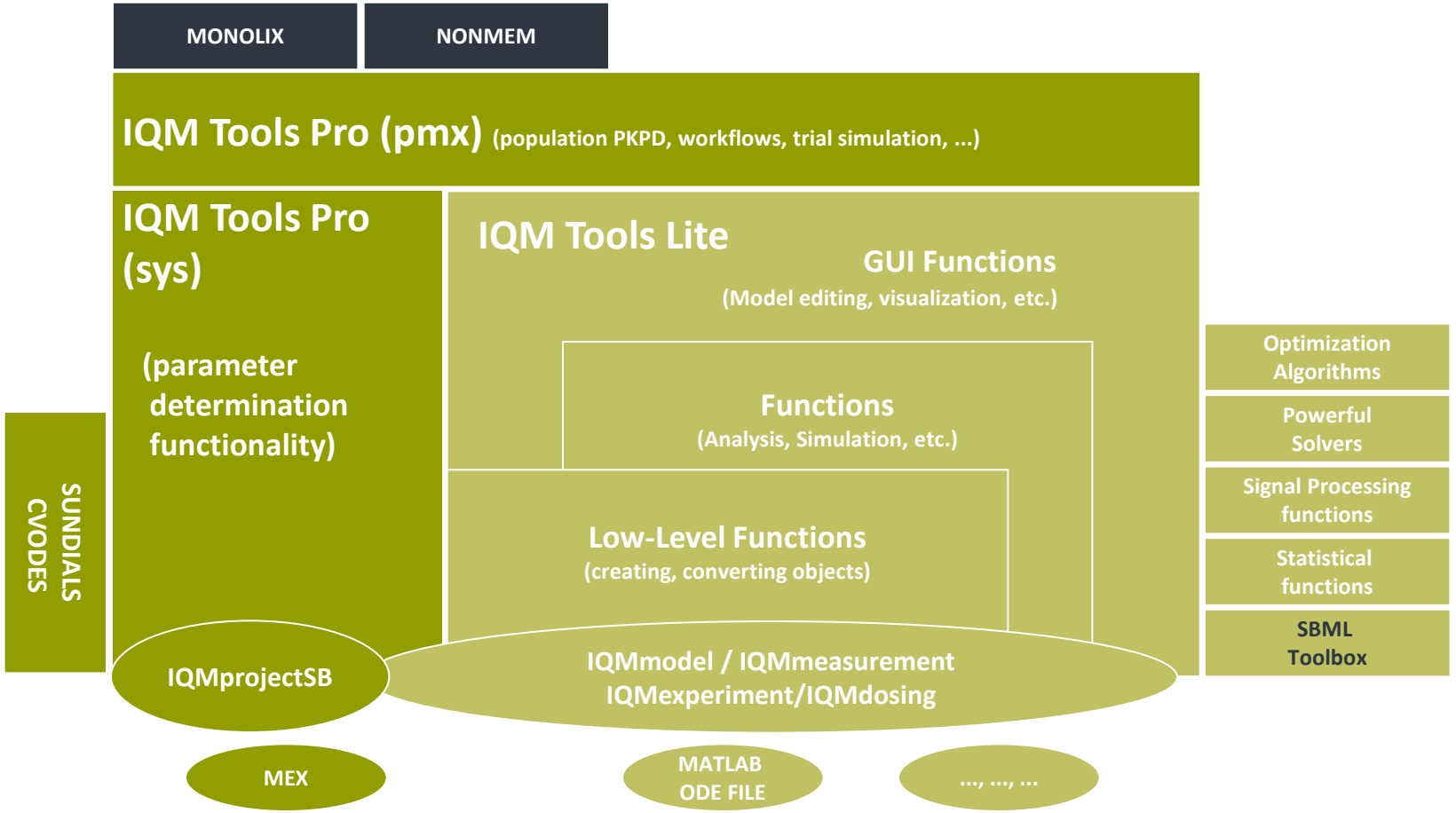
Introduction to the "IQM Tools Suite"

- The IQM Tools Suite is provided freely and as open source
- The IQM Tools Suite consists of two main packages
  - IQM Tools Lite
  - IQM Tools Pro
- The whole is based on MATLAB (www.mathworks.com)

IQM Tools Lite	Model, experiment, measurement & dosing representation, simulation, analysis functions, optimization, signal processing, statistical functions, etc.
IQM Tools Pro	Systems biology/pharmacology and PMX functionality, clinical data analysis, nonlinear mixed effect modeling, clinical trial simulations, high speed simulation through transparent C-code interface

henning.schmidt@intiquan.com

Modular Design of the IQM Tools Suite



henning.schmidt@intiquan.com

IntiQuan

Requirements

- **MATLAB R2013b (or later)**
  - Model reduction methods require the Symbolic Toolbox
  - Availability of the Parallel Toolbox useful for the Pharmacometric functions in IQM Tools Pro

- **Optional 3rd party software**
  - Monolix Version 4.2.3 (or later) (http://www.lixoft.org)
  - NONMEM Version 7.2 (or later) (http://www.iconplc.com)
  - SSm Global Optimization Toolbox (http://www.iim.csic.es/~gingproc/ssmGO.html)
  - SBML Toolbox (http://sbml.org/Software/SBMLToolbox)
    - Only needed for Unix/Linux/Mac
    - For Windows it is included in the distribution of the IQM Tools Lite

Where to get IQM Tool Suite from?

- Free download of IQM Tools Suite available from the IntiQuan webpage

- The IQM Tools Suite is distributed as a ZIP file with both IQM Tools Lite and IQM Tools Pro included

- Unzip this ZIP file on your computer at a location where you want to store the IQM Tools

How to Install the IQM Tool Suite?

- Start MATLAB
- Change into the “IQM Tools Suite” folder
- Read and update custom information in setup files:
  - IQMlite/SETUP_PATHS_TOOLS_IQMLITE.m
  - IQMpro/SETUP_PATHS_TOOLS_IQMPRO.m
- Execute the “installIQMtoolsInitial” script

You need to execute the “**installIQMtoolsInitial**” script once after obtaining a copy of IQM Tools. This will compile required libraries.

After this first installation, you can use the function “**installIQMtools**” to install IQM tools. This needs to be done each time you exit and start MATLAB again. This is on purpose for compliance and reproducibility reasons.

If you do not care about compliance, you might want to consider the use of a startup.m script (see http://www.mathworks.com/help/matlab/ref/startup.html).

Installation of optional 3rd party software

- Please follow the providers instructions when installing optional 3rd party software.

IQM Tools' Documentation

- **MATLAB style help**

```
>> help IQMlite
>> help IQMpro

>> help IQMsimulate
>> help IQMexportCSVdataset

>> doc IQMlite
>> doc IQMsimulate
```

- IQM Tools Tutorials with examples
  - Part 1: IQM Tools Lite (General Model Specification, Simulation, etc.)
  - **Part 2: IQM Tools Pro (MEX / Systems Biology/Pharmacology Projects)**
  - Part 3: IQM Tools Pro (Basic Pharmacometrics)
  - Part 4: IQM Tools Pro (General Dataset Specification and PMX Workflows)
  - Part 5: IQM Tools Pro (Advanced Clinical Trial Simulations)
  - Part 6: IQM Tools Pro (Linking Systems Pharmacology Models to Clinical Data)

- IQM Tools – Workshops
  - Given on demand and on some conferences during a year

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- **High performance computation via MEX simulation functions**
- Model Simulation
- IQMprojectSB - parameter estimation, model reduction

MEX Simulation Functions – Why?

- Parameter estimation and clinical trial simulations require many repeated simulations

  => Simulation speed matters a lot

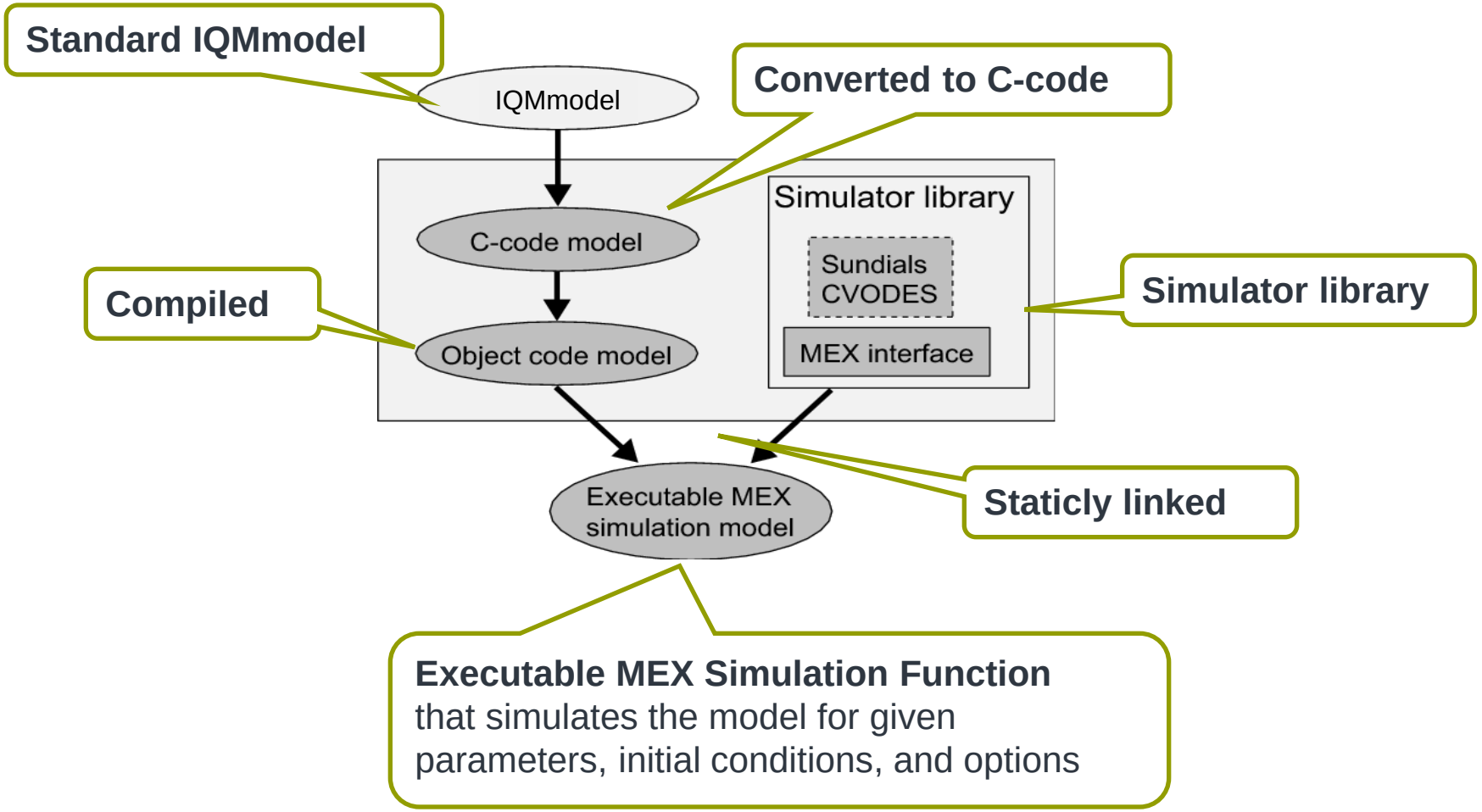
- Benchmark (**MEX simulation functions** vs. **ODE15s**)

	Model 1	Model 2	Model 3
Model description	Novak Tyson cell-cycle model	Full-scale model of glycolysis in yeast	Model 14 from the Biomodels.net database
End time (TEND)	1000	50	300
Number simulation points (NRPOINTS)	1000	200	300
Average time ODE15S [ms]	3442	3448	743
Average time SBPD [ms]	24	68	24
Speedup by SBPD	**144x**	**51x**	**30x**

henning.schmidt@intiquan.com

MEX Simulation Functions

- What are MEX Simulation Functions?



MEX Simulation Functions - Generation

- A simple example (change into the „**Example Files**“ folder)

```
>> model = IQMmodel('model1.txt')                % load a model
>> IQMmakeMEXmodel(model, 'model1MEX')           % create a MEX simulation function
```

- Simulation of a MEX model

```
>> simdataMEX = model1MEX([0:0.1:25])
>> simdata = IQMsimulate(model,[0:0.1:25])
```

```
>> plot(simdataMEX.time,simdataMEX.statevalues,'k')
>> hold on; plot(simdata.time,simdata.statevalues,'r--')
```

- The results of both simulations are identical
- C-Conversion only (no compilation) => **WYSIWYC**

```
>> IQMmakeMEXmodel(model,'MEXmodel',1)  % convert to MEXmodel.c and MEXmodel.h
>> edit MEXmodel.c                      % look at the created files
>> edit MEXmodel.h
>> IQMmakeMEXmodel(model,'MEXmodel')    % Same but with compilation!
```

henning.schmidt@intiquan.com

MEX Simulation Functions – Syntax

- How a MEX simulation function can be called

```
output = MEXmodel()

output = MEXmodel('states')

output = MEXmodel('parameters')

output = MEXmodel('parametervalues')
```

- With prior definition of **`timevector`**, **`initialconditions`**, etc.:

```
output = MEXmodel(timevector)

output = MEXmodel(timevector, initialconditions)

output = MEXmodel(timevector, initialconditions, parametervector)

output = MEXmodel(timevector, initialconditions, parametervector, options)
```

MEX Simulation Functions – Syntax

- Options for MEX model simulation (more available, see help text)

```
options.abstol:          Absolute tolerance (default: 1e-6)
options.reltol:          Relative tolerance (default: 1e-6)
options.minstep:         Minimal integrator step size (default: 0)
options.maxstep:         Maximal integrator step size (default: inf)
options.maxnumsteps:     Maximal number of steps between two output points (default: 500)
options.xdotcalc:        =0: do integration (default), =1: return RHS of ODEs for
                         given state and parameter values. Time information
                         is neglected and it is assumed that time=0.
```

- More information and options available in documentation

```
>> help IQMmakeMEXmodel
```

MEX Simulation

- A simple example (change into the „**Example Files**“ folder)

```
>> model = IQMmodel('model1.txt')                % load a model
>> IQMmakeMEXmodel(model, 'model1MEX')           % create a MEX simulation function
```

- Simulate again – in three different ways

```
>> simMEX1 = model1MEX([0:0.1:25])               % Calling directly the MEX function
>> simMEX2 = IQMPsimulate('model1MEX',[0:0.1:25]) % Simulate MEX function
>> simMEX3 = IQMPsimulate(model,[0:0.1:25])      % create MEX function on the fly
```

- Advantage of the IQMPsimulate function is that parameters and intial conditions can be changed more easily than when calling the MEX function directly

- Please have a look at the documentation of IQMPsimulate

```
>> help IQMPsimulate
```

henning.schmidt@intiquan.com

Limitations of MEX Simulation Functions

- The MEX models can not handle
  - SBML Algebraic rules
  - The fast reaction flag included in SBML
- Delays and delayed events can be handled, but the simulation performance becomes VERY poor ... In this case the standard MATLAB simulation is faster
- C compatibility necessary (no MATLAB functions can be used that do not exist in C)
  - Several standard functions provided by IQM Tools (andIQM, delayIQM, indexmaxIQM, maxIQM, minIQM, multiplyIQM, orIQM, piecewiseIQM, piecewiseSmoothIQM, piecewiseT0IQM, xorIQM)

Tutorial Outline

- General introduction to the IQM Tools Suite
- High performance computation via MEX simulation functions
- **Model Simulation**
  - Standard
  - Parametric sensitivity
- IQMprojectSB - parameter estimation, model reduction

Deterministic Simulation

- Serves also as example for using the Cell-Mode

- Change into the „**Example Files**“ folder

```
>> edit simpleSimulation
```

1. Read the documentation in the opened file
2. Execute the cells sequentially by pressing „**Ctrl+Enter**“

Forward Sensitivity Analysis

- Determine the sensitivy trajectories wrt to perturbations in IC and parameters

$$\begin{aligned} \frac{d}{dt}x &= f(x,p) \\ x(0) &= x_0 \end{aligned}$$

$$\begin{aligned} S(t) &= \frac{dx(t)}{dp} \\ \frac{d}{dt}S &= \frac{\partial f(x,p)}{\partial x}S + \frac{\partial f(x,p)}{\partial p} \\ S(0) &= 0 \end{aligned}$$

```
>> model = IQMmodel('novaktyson1.txt')
>> output = IQMsensitivity(model, [0:1:200], {'k1', 'Ka', 'khs'}, {'Cyclin','YT'})
output =

                time: [1x201 double]
              states: {'Cyclin'  'YT'  'PYT'  'PYTP'  'MPF'  'Cdc25P'  'Wee1P'  'IEP'  'APCstar'}
         statevalues: [201x9 double]
           variables: {'k2'  'kwee'  'k25'}
      variablevalues: [201x3 double]
           reactions: {'R1'  'R2'  'R3'  'R4'  'R5'  'R6'  'R7'  'R8'  'R9'  'R10'  'R11'  'R12'  'R13'  'R14'  ... 'R19'}
      reactionvalues: [201x19 double]
      sensparameters: {'k1'  'Ka'  'khs'}
    paramtrajectories: [1x1 struct]
        sensicstates: {'Cyclin'  'YT'}
      ictrajectories: [1x1 struct]


>> output.paramtrajectories
ans =

      states: {[201x9 double]  [201x9 double]  [201x9 double]}
   variables: {[201x3 double]  [201x3 double]  [201x3 double]}
   reactions: {[201x19 double]  [201x19 double]  [201x19 double]}
```

henning.schmidt@intiquan.com

Forward Sensitivity Analysis

- The function IQMsensitivity can be used on IQMmodels or on MEX simulation functions
- Underlying the CVODES integrator is used to determine the sensitivity trajectories
- This is considerably faster than determining the sensitivities by finite differences and repeated simulations

```
>> IQMmakeMEXmodel(model,'modelMEX')
>> output = IQMsensitivity('modelMEX',[0:1:200], {'k1', 'Ka', 'khs'}, {'Cyclin','YT'})
output =

                time: [1x201 double]
              states: {'Cyclin'  'YT'  'PYT'  'PYTP'  'MPF'  'Cdc25P'  'Wee1P'  'IEP'  'APCstar'}
         statevalues: [201x9 double]
           variables: {'k2'  'kwee'  'k25'}
      variablevalues: [201x3 double]
           reactions: {'R1'  'R2'  'R3'  'R4'  'R5'  'R6'  'R7'  'R8'  'R9'  'R10'  'R11'  'R12'  'R13'  'R14'  ... 'R19'}
      reactionvalues: [201x19 double]
      sensparameters: {'k1'  'Ka'  'khs'}
    paramtrajectories: [1x1 struct]
        sensicstates: {'Cyclin'  'YT'}
      ictrajectories: [1x1 struct]


>> output.paramtrajectories
ans =

      states: {[201x9 double]  [201x9 double]  [201x9 double]}
   variables: {[201x3 double]  [201x3 double]  [201x3 double]}
   reactions: {[201x19 double]  [201x19 double]  [201x19 double]}
```

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- High performance computation via MEX simulation functions
- Model Simulation
- **IQMprojectSB - parameter estimation, model reduction**
  - Projects
  - Parameter estimation / manual tuning / parameter fit analysis / parameter identifiability analysis
  - Model reduction
  - Simulation of projects
  - Commenting of projects

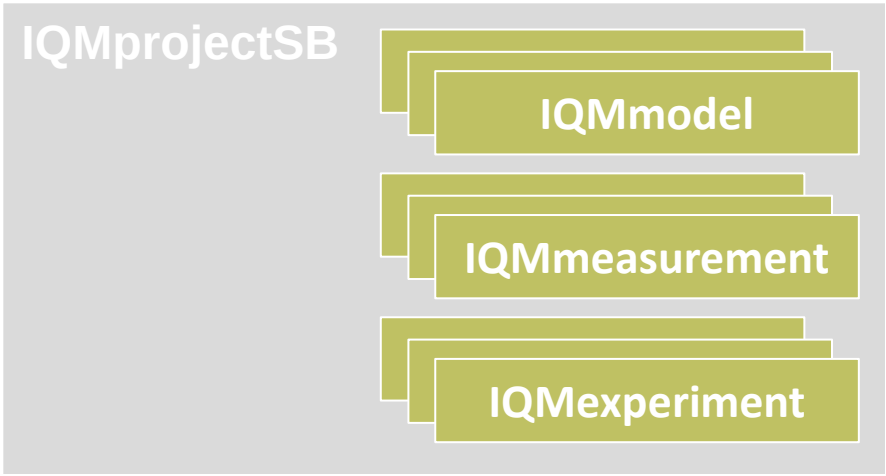
- Projects
  - Modeling example

IQMprojectSB

henning.schmidt@intiquan.com

Modeling Projects: IQMprojectSB

- An **`IQMprojectSB`** is a container for
  - Models
  - Measurement data
  - Experiment descriptions
  - Modeling information



- Parameter estimation, etc., can directly be run on projects
- **When modeling a system we build a project**

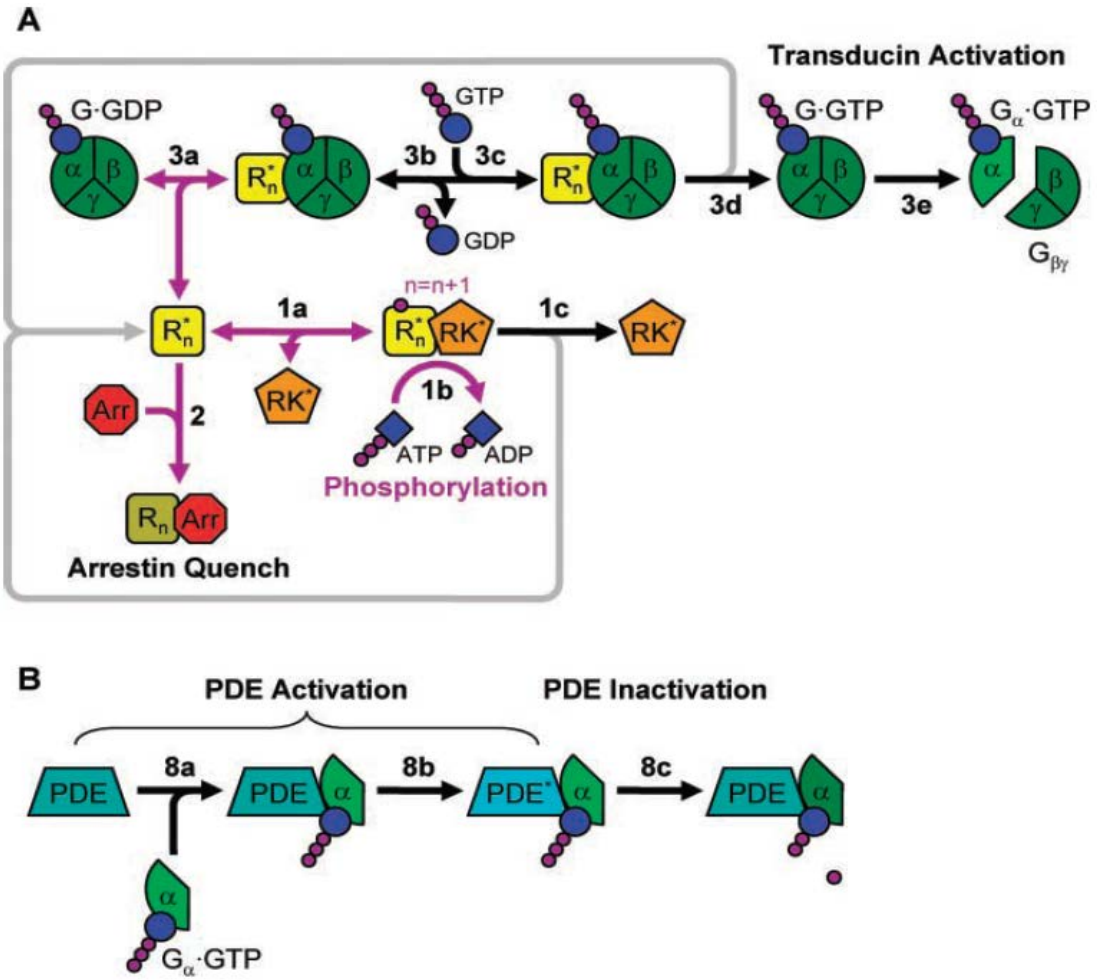
Let‘s do it!

henning.schmidt@intiquan.com

IntiQuan

The System to be Modeled:

Photo Transduction in Rod Cells

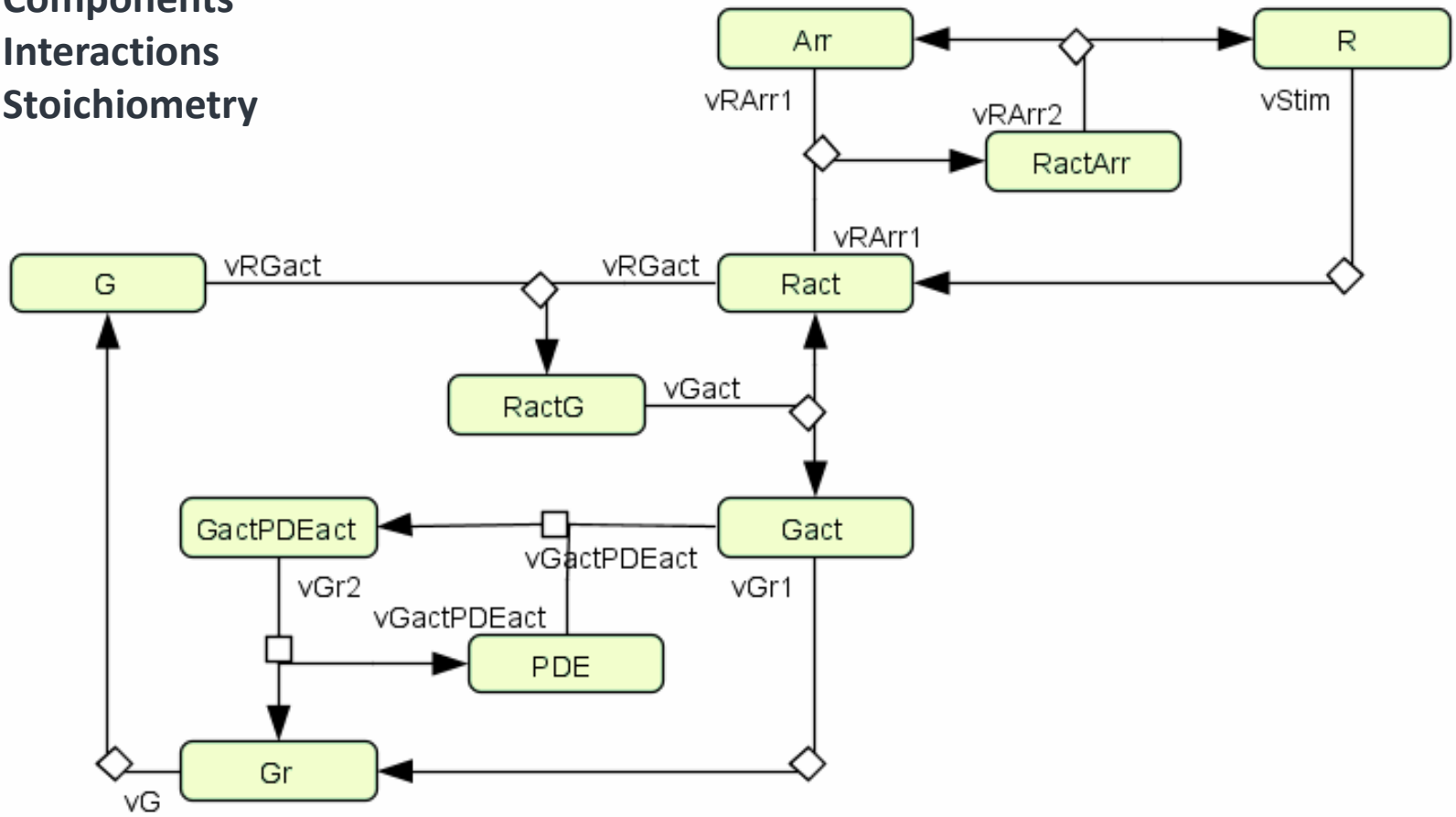


The example shown in the following is adapted for this tutorial

henning.schmidt@intiquan.com

1) Model Network Structure

Components
Interactions
Stoichiometry



Adding mathematics in graphical tools is often a pain

henning.schmidt@intiquan.com

IntiQuan

2) Kinetic Rate Laws

- Export graphical model to SBML => Import to IQM Tools
- Add kinetic rate expressions in the IQMmodel
  - We start simple, by assuming **mass action kinetics**
  - Only in a single reaction we assume Michaelis Menten kinetics
- Initialize parameters with guessed values
  - Nothing known about the parameters => we set all to „1“

3) Initial Conditions

- In this example we assume that:
  Carefully undertaken experiments showed that the number of molecules of the involved species (WT) in the inactive state (thus their total amount) is as follows:

  R = 500

  G = 3000

  Arr = 5

The number of PDE molecules is unknown but believed to be somewhere between 10 and 1000

All remaining species have an initial amount of 0

The Initial Model

- Based on the previous information we can construct an IQMmodel representation

- Change into the folder:
  „**Example Files\projectexample\phototransduction project\models**”

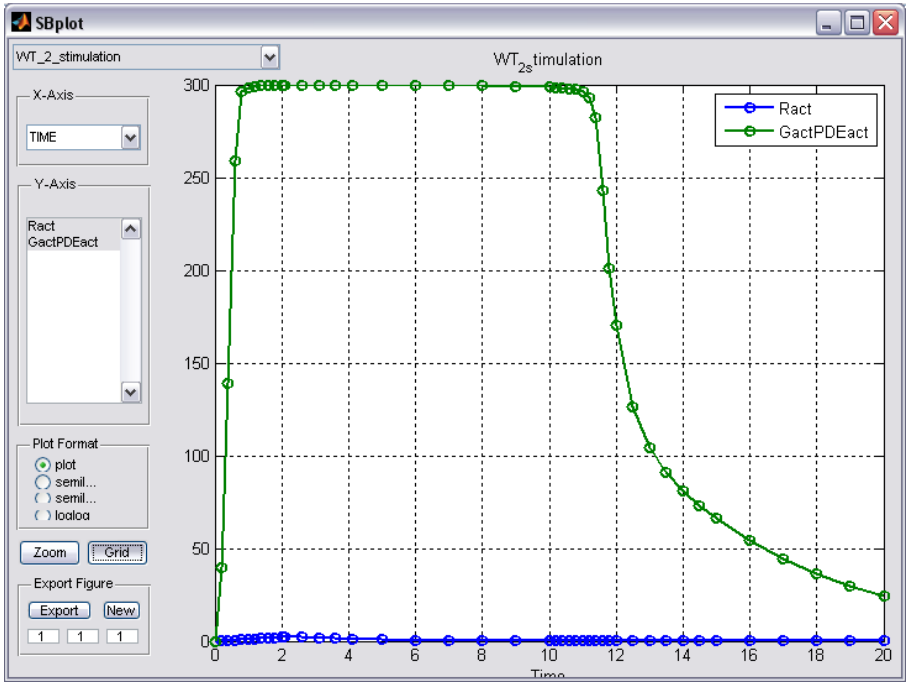
```
>> edit model.txtbc          % or just doubleclick on the model file
```

- Have a look at the model

- Realizing the pulse stimulation:
  stimulus = piecewiseIQM(magStim,le(time,durStim),0)

4) Measurement Data

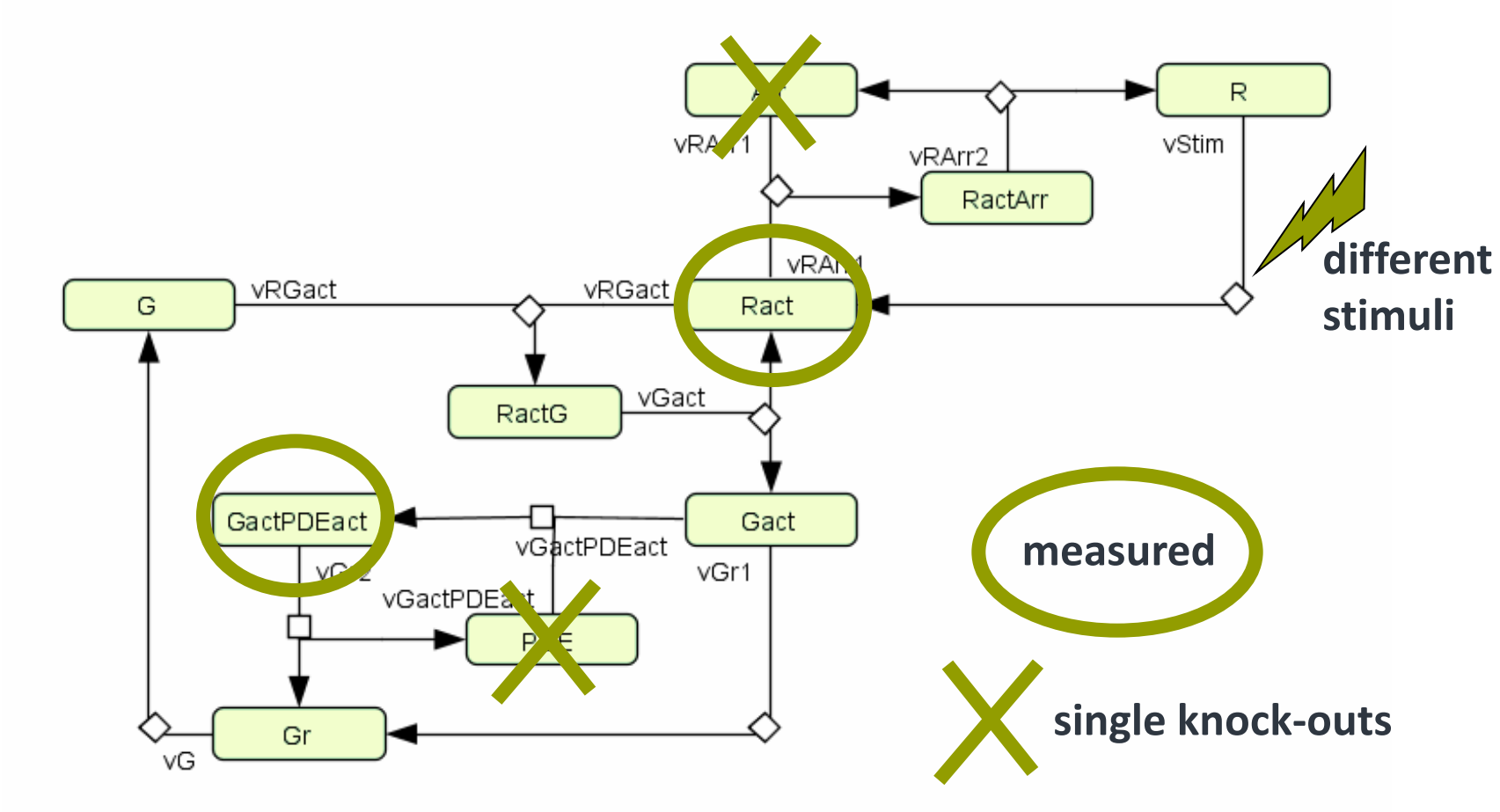
- The experimentalists gave us **4 documented CSV data files**
  - Time-series measurement data of Ract and GactPDEact
  - **Information about the biological experiments** that have been performed to obtain the data
  - The information in the documentation is the base for the specification of the experiment descriptions



henning.schmidt@intiquan.com

IntiQuan

Experiments and Measurements



Time-series data measured

henning.schmidt@intiquan.com

IntiQuan

5) Constructing Experiment Descriptions

- Change into the folder:

  „**Example Files\projectexample\phototransduction project\experiments\deltaArr_01**”

```
>> edit deltaArr_0point1_stimulation.csv  % or right-click => “open as text”
```

- Have a look at the measurement documentation

- Define experiment description:

```
********** EXPERIMENT INITIAL PARAMETER AND STATE SETTINGS
Arr(0) = 0  % knock-out

magStim = 2
durStim = 0.1
```

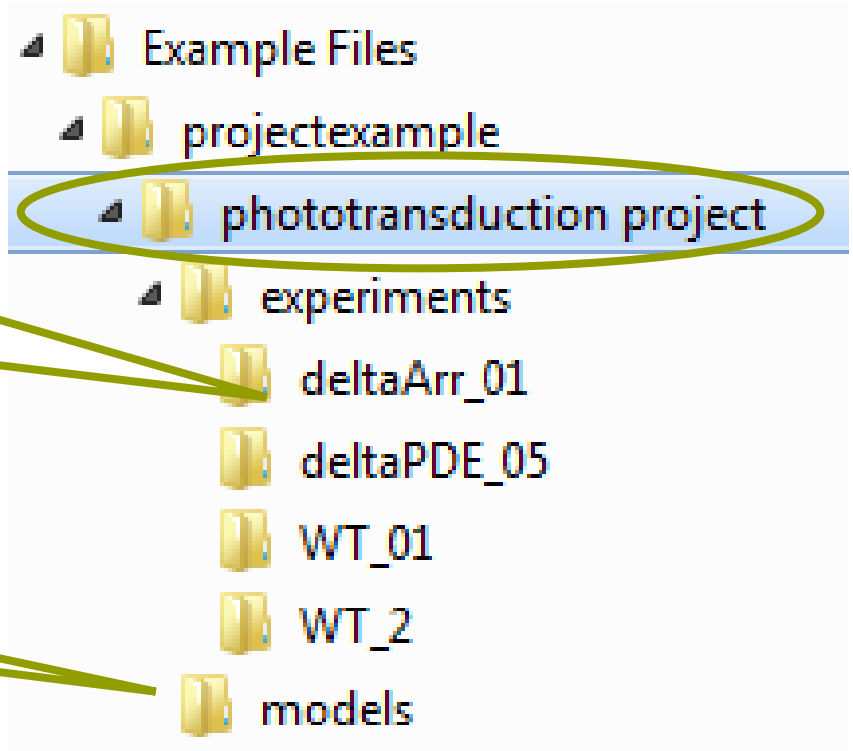
```
>> edit deltaArr_0point1_stimulation.exp  % or right-click => “open as text”
```

Representation of IQMprojectsSB

- Folder structure, containing
  - Models
  - Experiments
  - Measurements

Experiment folders containing **experiment description file** and corresponding **measurement data**

Folder containing **model(s)** TEXT (ODE and/or BC) and/or SBML



Use telling names for the experiment folders. It will help you later!

henning.schmidt@intiquan.com

Projects – Import and Information

- Change into the „**example files/projectexample**“ folder
- Import of the project

```
>> sbp = IQMprojectSB('phototransduction project')
     IQMprojectSB
     ============
     Name: phototransduction project
     Number Models:                          1
     Number Experiments:     4
     Number Measurements:    4
     Number Estimations:     0
```

- Project information

```
>> IQMinfo(sbp)
===PROJECT INFO=================================================
Project name: phototransduction project
---NOTES---------------------------------------------------------
Project notes: This is an example project for the IQM Tools tutorial

In this notes.txt file you can document your project. Additionally,
you can add all kinds of additional documents in the different folders.
---MODELS--------------------------------------------------------
Model 1: Project_example_model
---EXPERIMENTS---------------------------------------------------
Experiment 1: WT_0point1_stimulation
      Measurement 1: WT_0point1_stimulation
Experiment 2: WT_2_stimulation
      Measurement 1: WT_2_stimulation
Experiment 3: deltaARR_0point1_stimulation
      Measurement 1: deltaArr_0point1_stimulation
Experiment 4: deltaPDE_0point5_stimulation
      Measurement 1: deltaPDE_0point5_stimulation
---ESTIMATIONS---------------------------------------------------
0 estimations present
================================================================
```

Adding New Experiment Results

1. Create new folder in „experiments“ folder
2. Copy measurement data in (Excel or CSV file)
3. Create an experiment description

DONE!

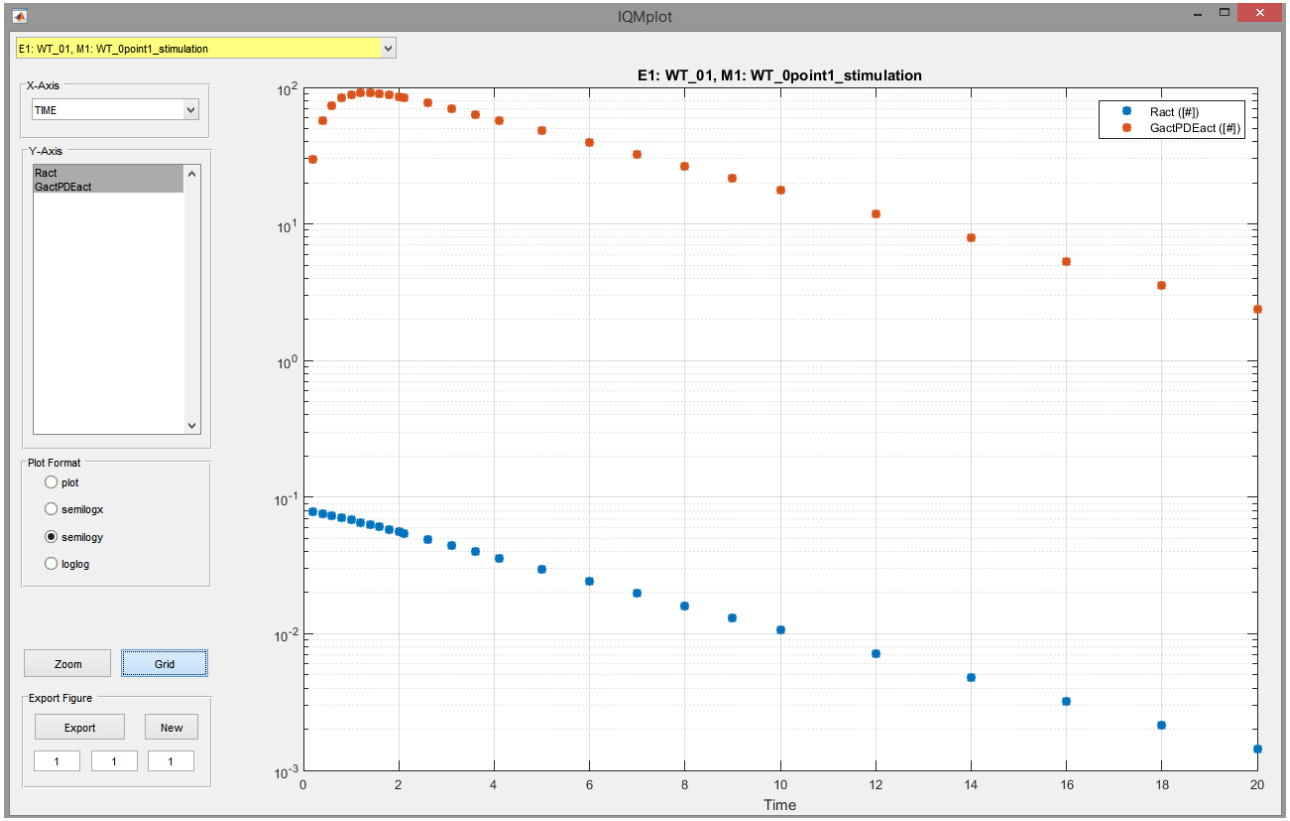
- **Names in measurement files and experiment descriptions have to be the same as for the corresponding elements in the model**

henning.schmidt@intiquan.com

Projects – Visualize Measurement Data

- Measurement data can be plotted by

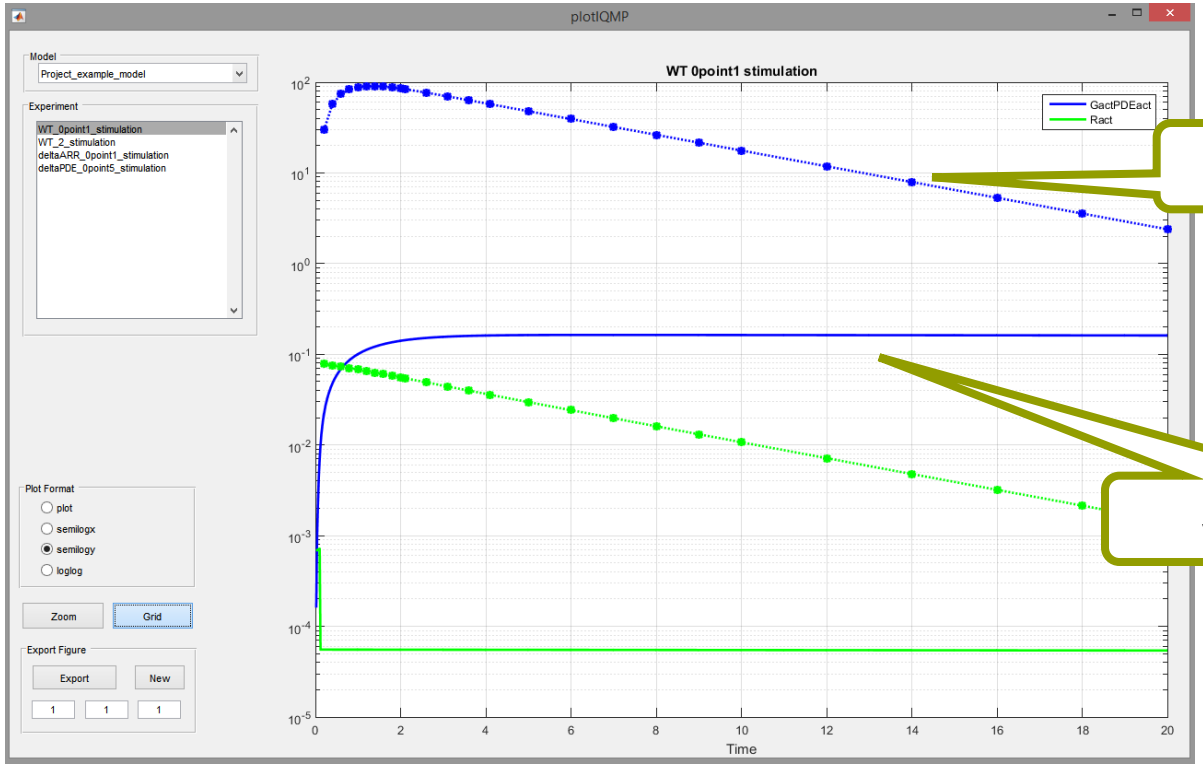
```
>> IQMplotmeasurements(sbp)
```



Projects – Simple Simulation

- Simulation of the experiments in the project
- Comparison with the measurement data

```
>> IQMcomparemeasurements(sbp)
```



Measurements (-o-)

Simulations (–)

henning.schmidt@intiquan.com

IntiQuan

Projects – Export

- Saving a project as single binary file

```
>> IQMsaveproject(sbp,'projectfilename')
```

  - Saves the project in the file: **projectfilename.iqmp**
  - Can be loaded again by

```
>> sbp = IQMprojectSB('projectfilename.iqmp')
```

- Exporting the project to a folder structure

```
>> IQMexportproject(sbp, 'projectfoldername')
```

  - During export measurement data are always saved as CSV files!

henning.schmidt@intiquan.com

- Parameter estimation / manual tuning / parameter fit analysis / identifiability analysis

IQMparamestGUI++

henning.schmidt@intiquan.com

IntiQuan

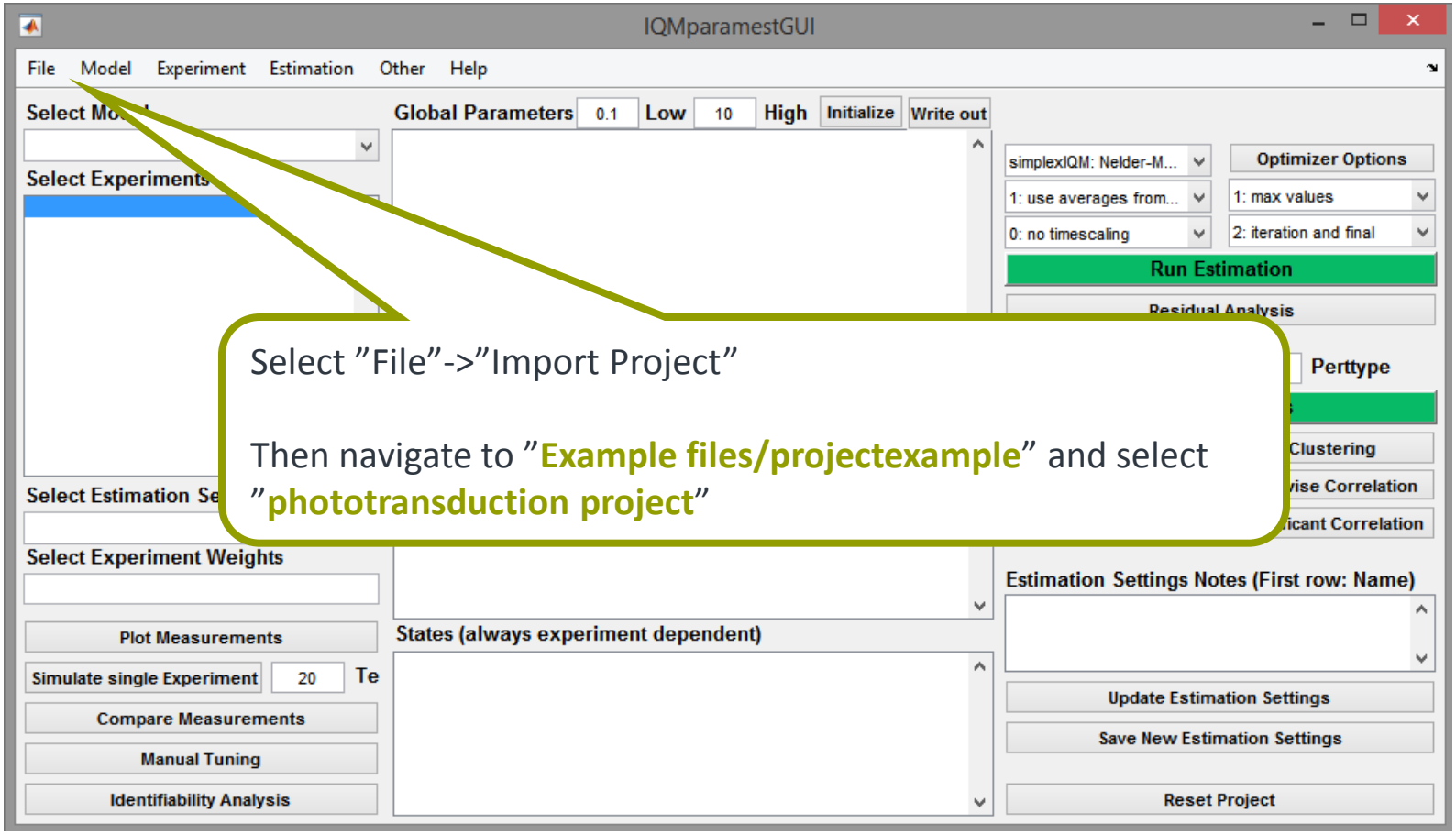
Project is Defined – What Now?

- Several possibilities to perform parameter estimation, etc.
  - All from command line (messy)
  - Using a „**runEstimation**“ script (easy ... we will see it later)
  - Using a graphical user interface (easy)

- Start the GUI

```
>> IQMparamestGUI
```

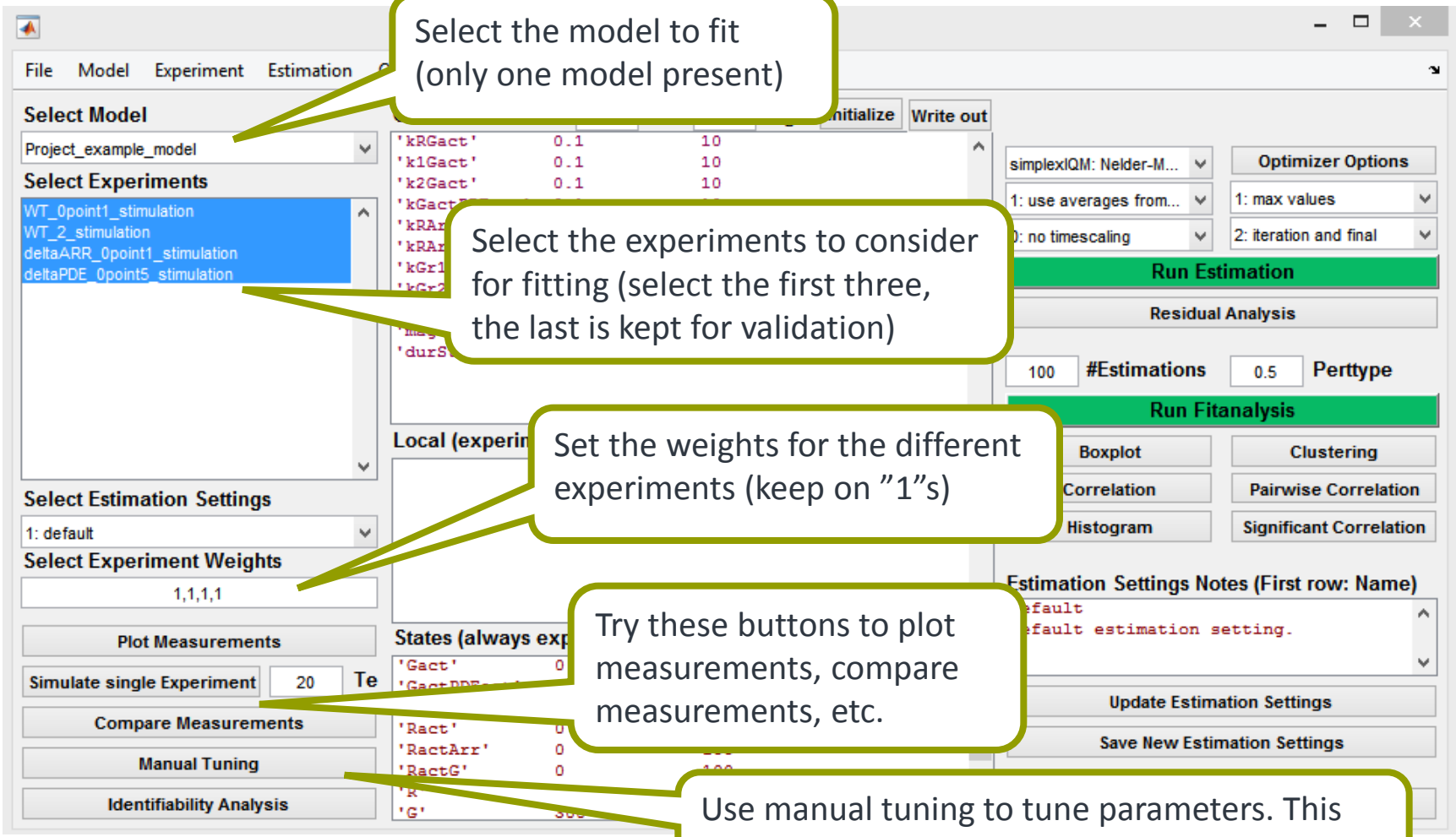
IQMparamestGUI



henning.schmidt@intiquan.com

IntiQuan

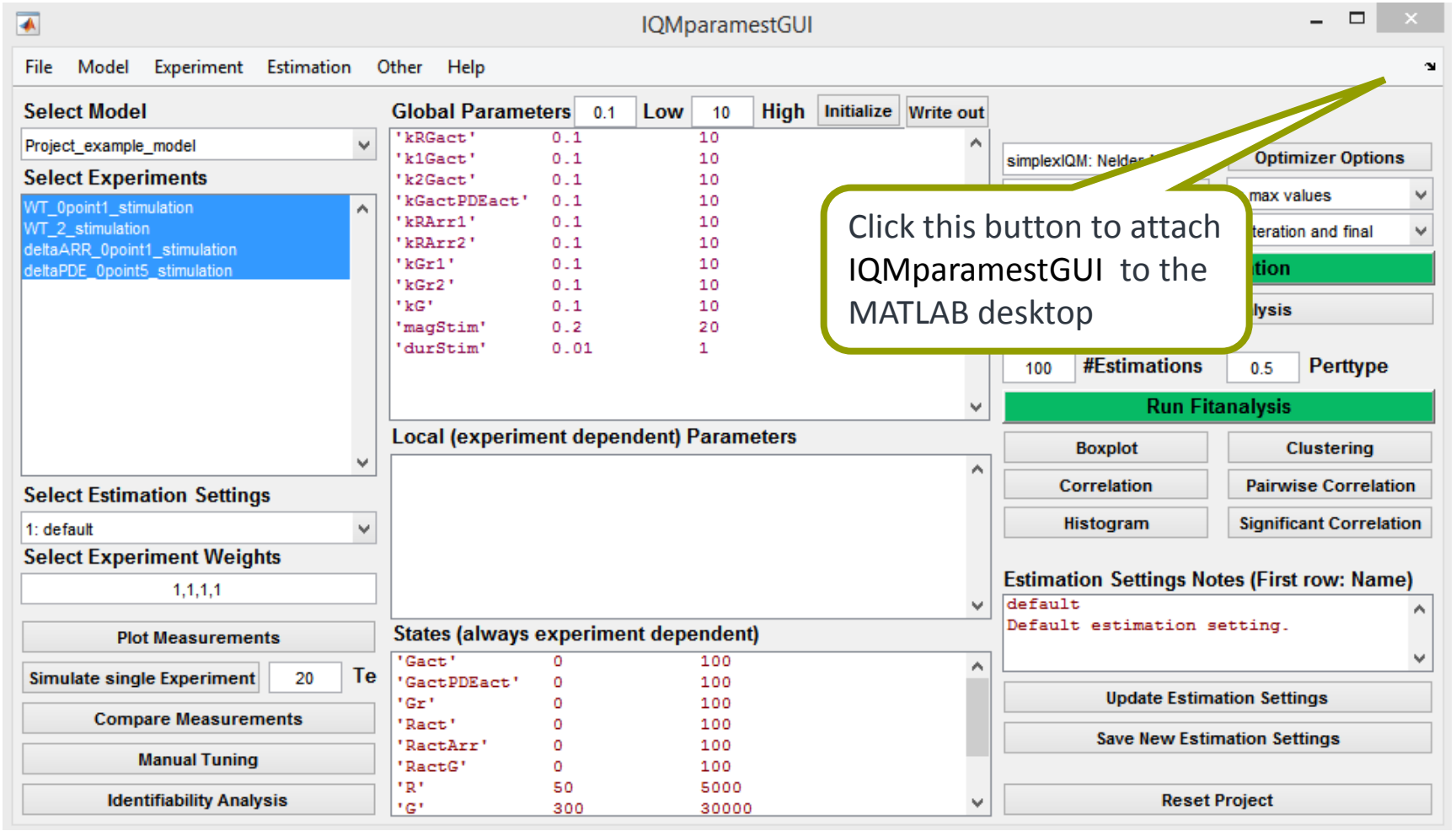
IQMparamestGUI



henning.schmidt@intiquan.com

IntiQuan

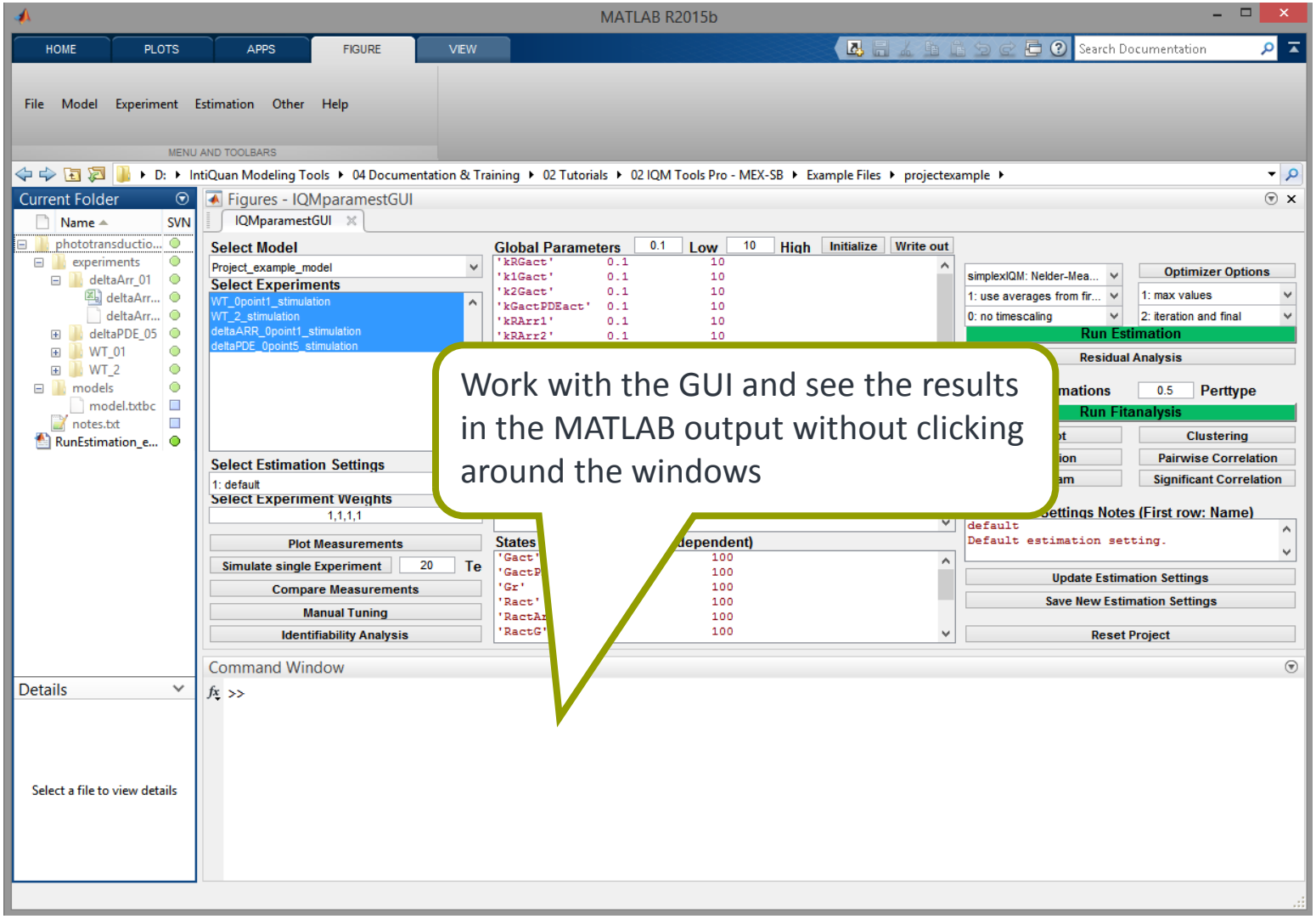
IQMparamestGUI



henning.schmidt@intiquan.com

IntiQuan

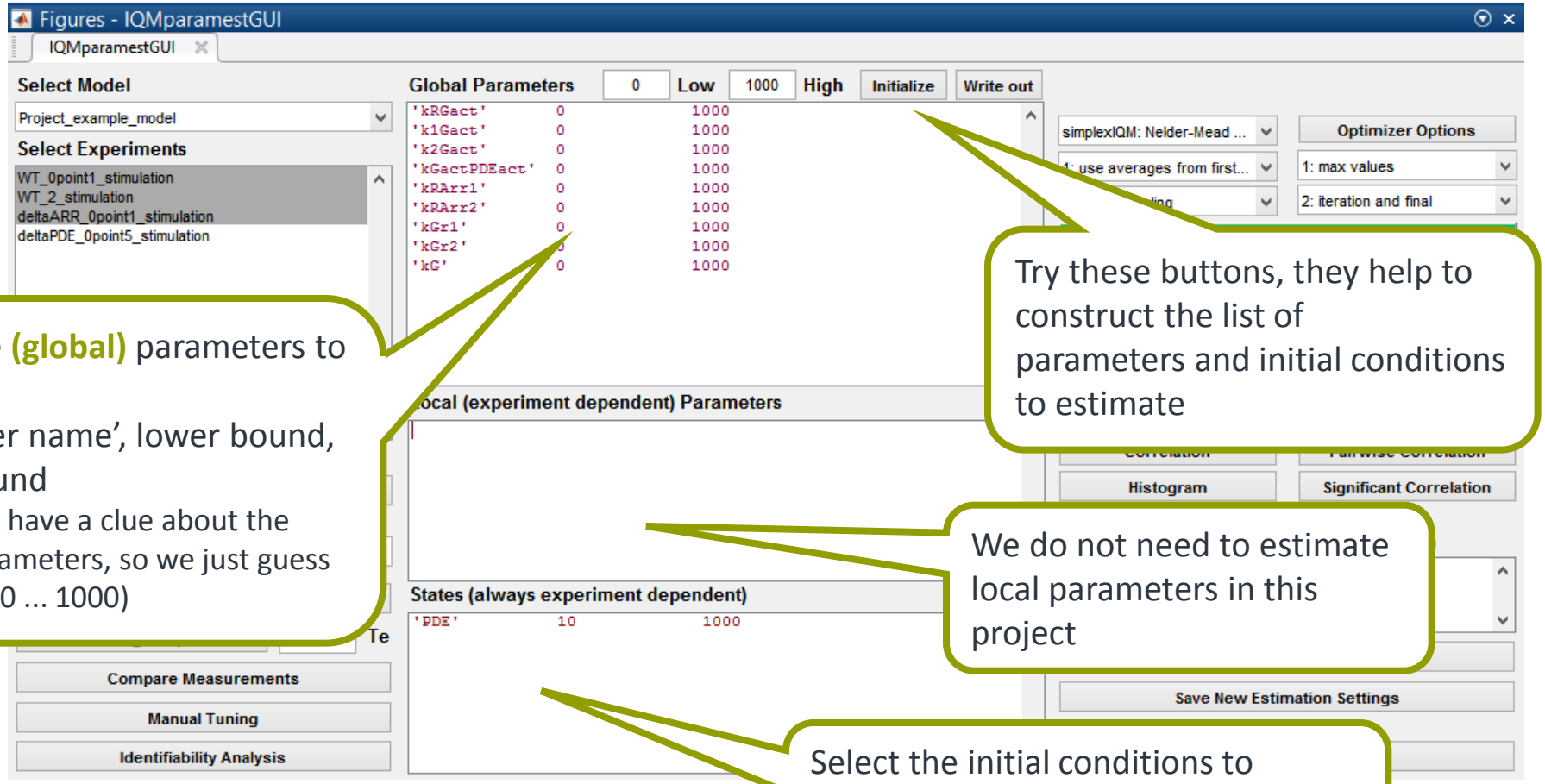
IQMparamestGUI



henning.schmidt@intiquan.com

IntiQuan

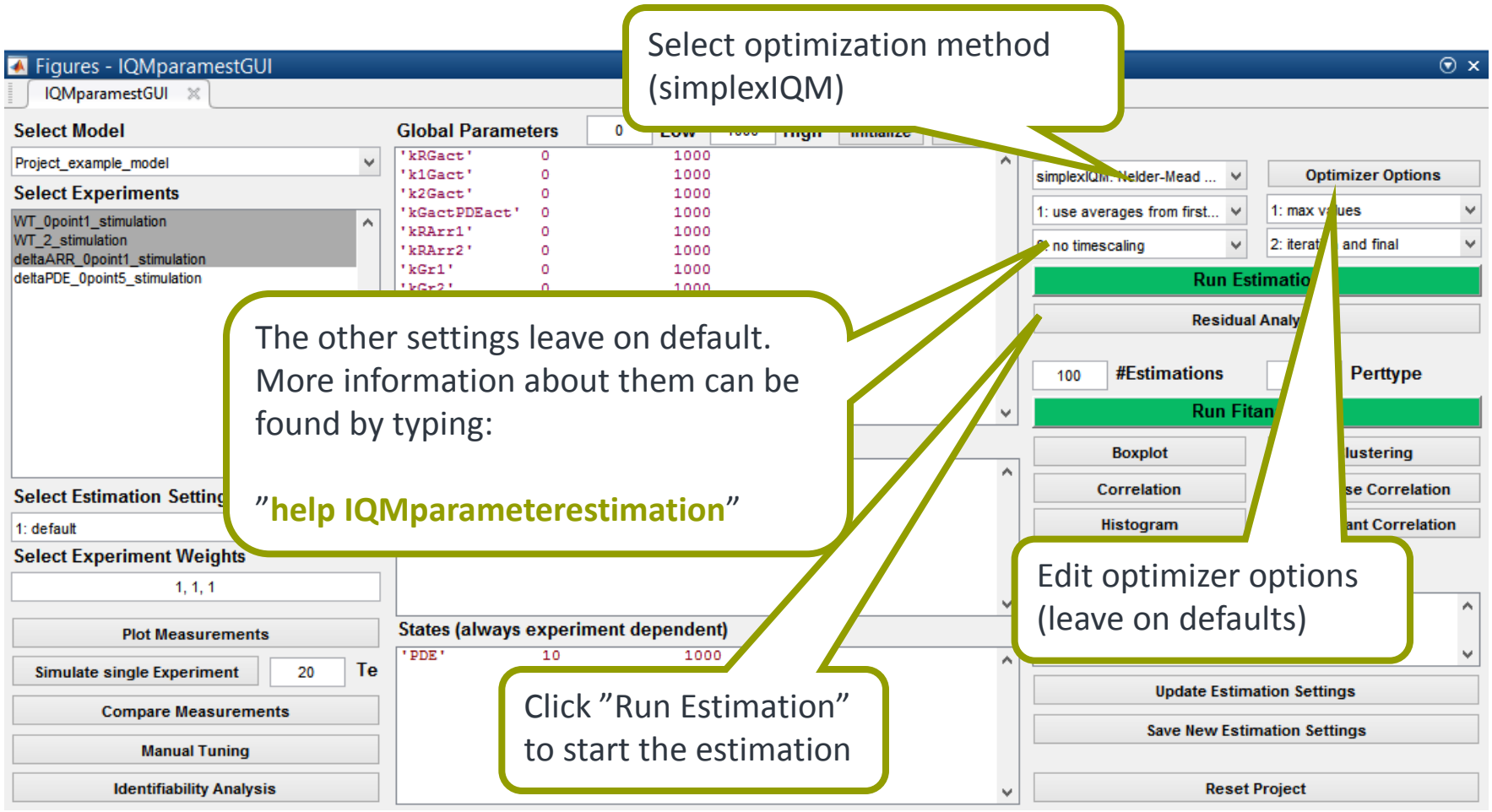
IQMparamestGUI



henning.schmidt@intiquan.com

IntiQuan

IQMparamestGUI



henning.schmidt@intiquan.com

IntiQuan

Parameter estimation

≠

Running an optimization and wait

henning.schmidt@intiquan.com

Parameter Estimation

- Different optimization methods are useful for different problems
- Many options for optimization methods => translation of **parameter estimation** into **optimizer options estimation**

- Simple guideline
  - Start with a local optimimzation method and see how good it gets
  - Restart optimization with a global method (here use fSSmIQM if you have installed the SSm GO toolbox, otherwise use, e.g., pswarmIQM)
  - Iterate between optimization and checking the results using „Compare Measurements“ or „Manual Tuning“
  - „Simulate Single Experiment“ helps in understanding what happens to the unmeasured parts in the model

Parameter Estimation 1st Preliminary Result

```
Estimated parameters
====================
kRGact = 0.76107
k1Gact = 999.989
k2Gact = 0.537476
kGactPDEact = 485.473
kRArr1 = 0.109028
kRArr2 = 12.9209
kGr1 = 0.0549069
kGr2 = 1.47609
kG = 0.466008

Estimated initial conditions
============================
PDE=115.492 (Experiment 1)
PDE=296.076 (Experiment 2)
PDE=921.481 (Experiment 3)

Optimal cost: 0.036001
```

VERY close to the upper bound => lets increase the upper bound to 10000

=> restart optimization

(preferably using first a local method: simplexIQM, alternating with fSSmIQM)

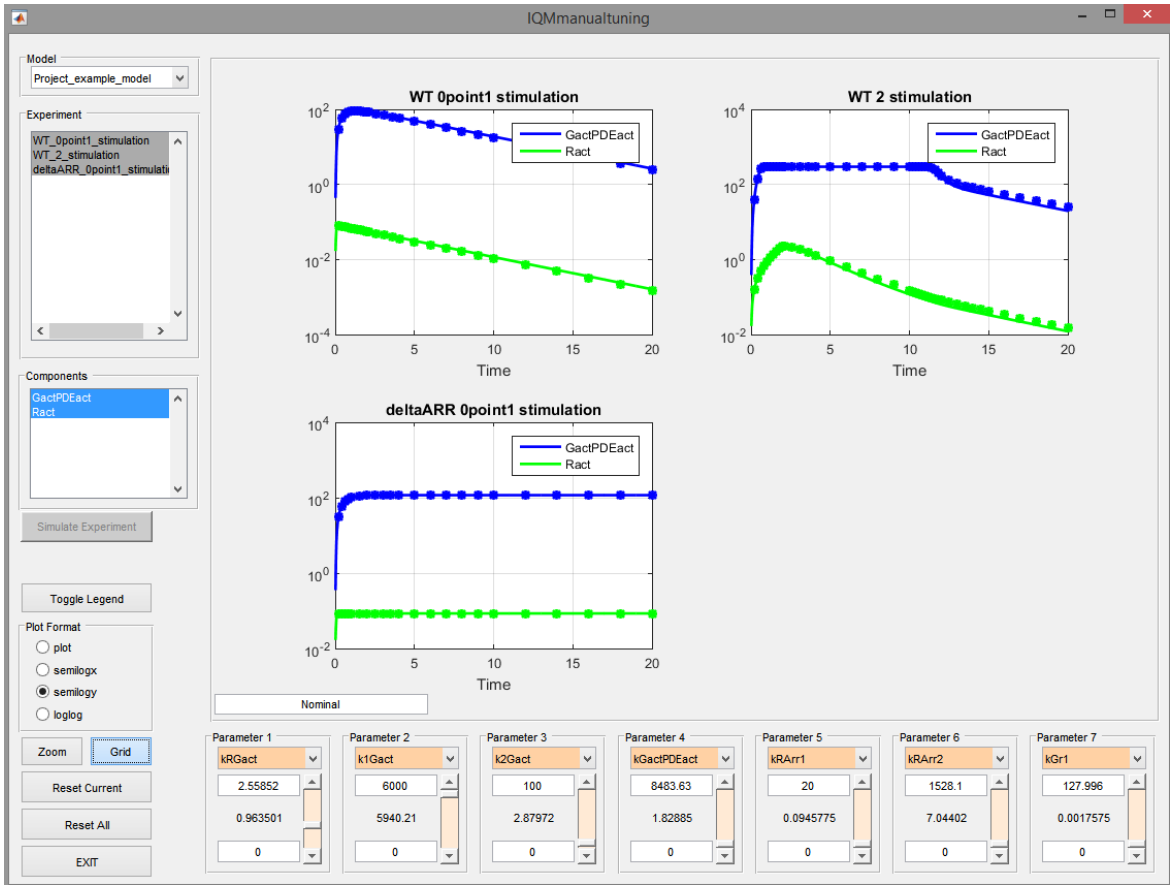
henning.schmidt@intiquan.com

Parameter Estimation 2nd Preliminary Result

```
Estimated parameters
====================
kRGact = 1.19197
k1Gact = 6277.36
k2Gact = 2.56596
kGactPDEact = 0.362095
kRArr1 = 0.104435
kRArr2 = 0.289185
kGr1 = 2.92379e-018
kGr2 = 2.31828
kG = 1.40081

Estimated initial conditions
============================
PDE=122.636 (Experiment 1)
PDE=300.322 (Experiment 2)
PDE=141.934 (Experiment 3)
```

Optimal cost: 0.000615584



Use „Manual Tuning“ or „Compare Measurements“
=> Fit seems acceptably nice

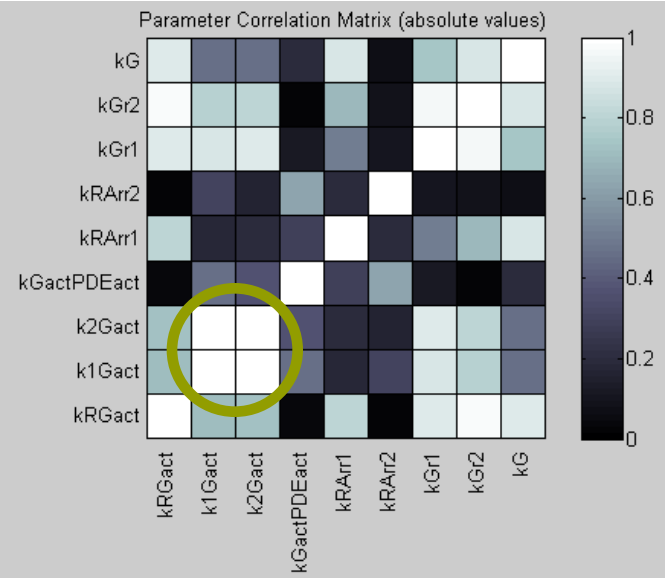
henning.schmidt@intiquan.com

IntiQuan

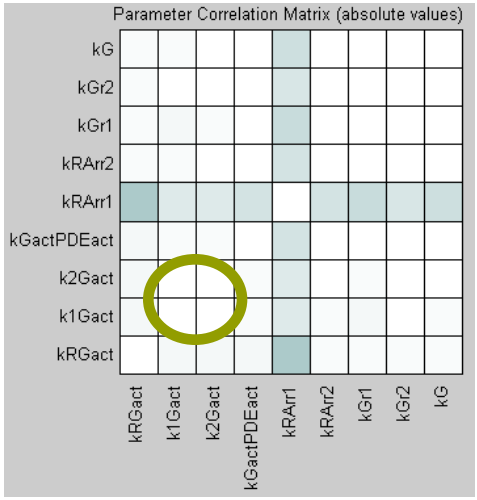
A Posteriori Identifiability Analysis

- Select one experiment at a time and click the „Identifiability Analysis“ button

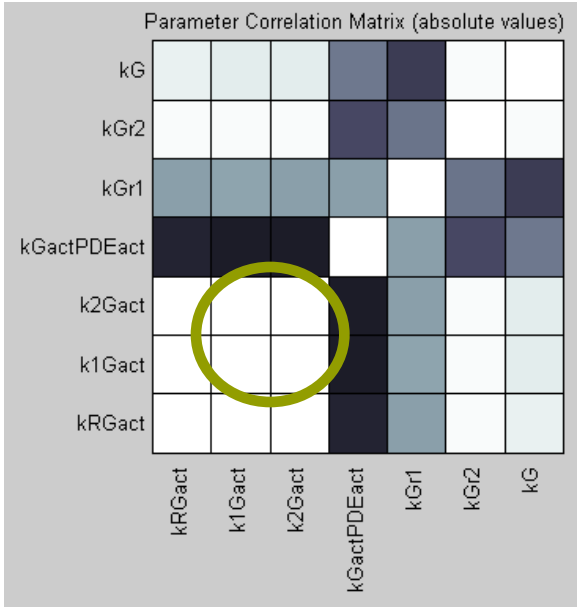
Experiment 1



Experiment 2



Experiment 3

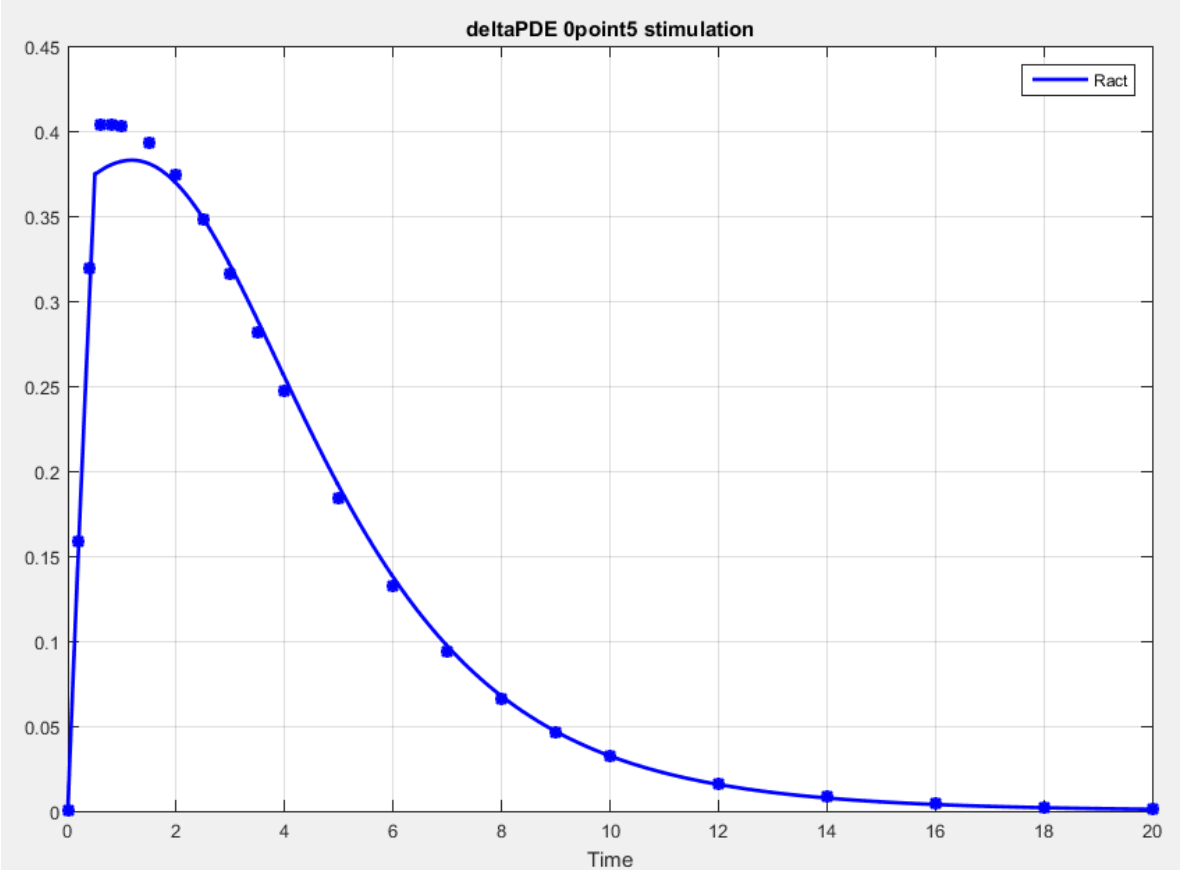


henning.schmidt@intiquan.com

IntiQuan

Validation of Models‘ Predictiveness

- Simulate the last experiment, which was not used for estimation



- Not perfect, but we have seen worse in other projects ☺

henning.schmidt@intiquan.com

IntiQuan

Don‘t Waste Measurement Data

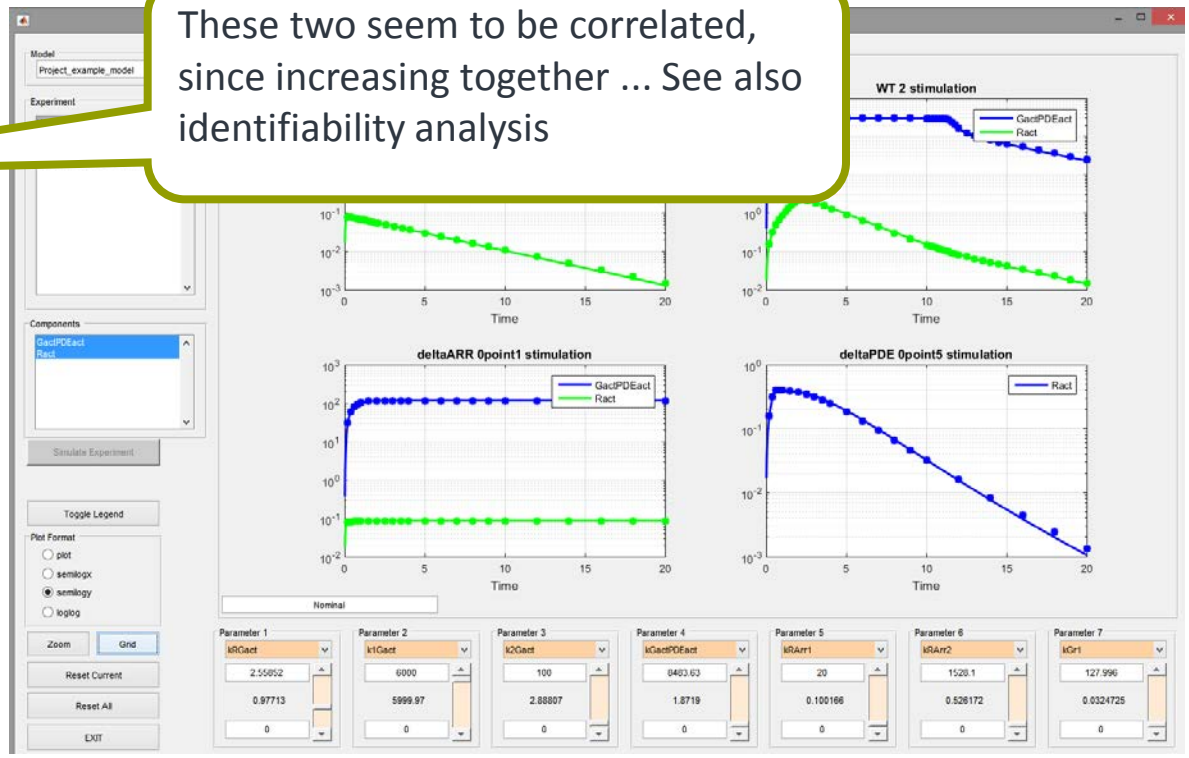
- One we have seen that the model is able to predict our validation data we should use these data to improve the model
- Reduce the upper bounds according to the current optimized values
- Again use simplexIQM and fSSmIQM sequentially

```
Estimated parameters
====================
kRGact = 1.15041
k1Gact = 9994.95
k2Gact = 4.18729
kGactPDEact = 0.373298
kRArr1 = 0.101745
kRArr2 = 0.397964
kGr1 = 0.0171438
kGr2 = 2.30743
kG = 2.36096

Estimated initial conditions
============================
PDE=128.758 (Experiment 1)
PDE=302.294 (Experiment 2)
PDE=152.618 (Experiment 3)
PDE=39.4611 (Experiment 4)

Optimal cost: 0.000423773
```

These two seem to be correlated, since increasing together ... See also identifiability analysis



henning.schmidt@intiquan.com

IntiQuan

Save Optimized Project

- To export the optimized project do the following
  - Select „**File**“->“**Export Project (Folder)**“
  - Navigate to the projectexample folder
  - Click OK
  - Type new name: „**phototransduction_project_optimized**”

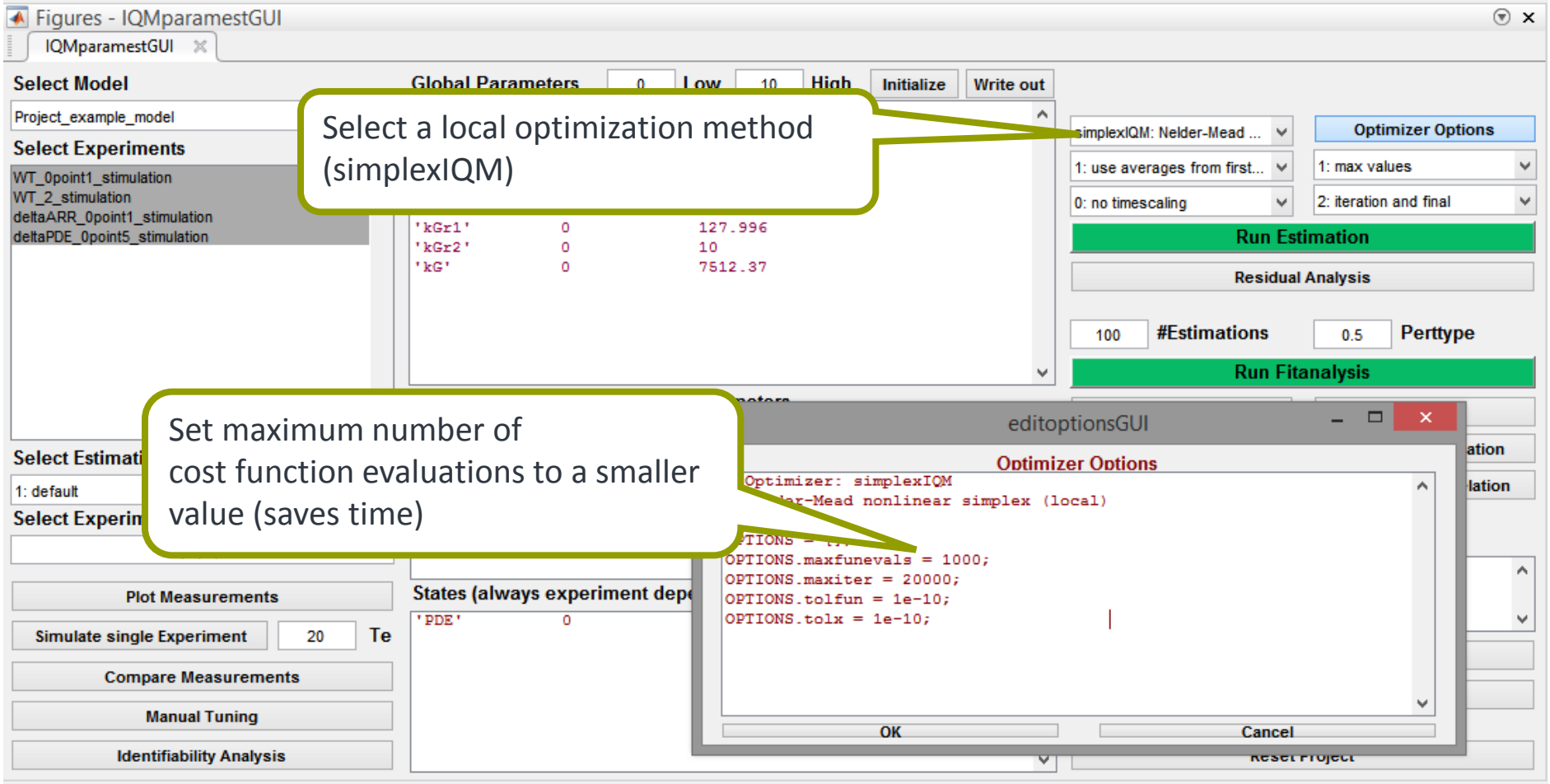
- Optimized global variables are stored in the model
- Optimized initial conditions and local variables are stored in the experiment decriptions
- Browse the newly created project folder and check the experiment descriptions and the model

Parameter Fit Analysis

- Idea:
  1. randomly perturb starting conditions for estimation around current optimum
  2. perform optimization and collect optimal values
  3. repeat N times
  4. analyze results

- Gives information about:
  - parameter correlations
  - local optima

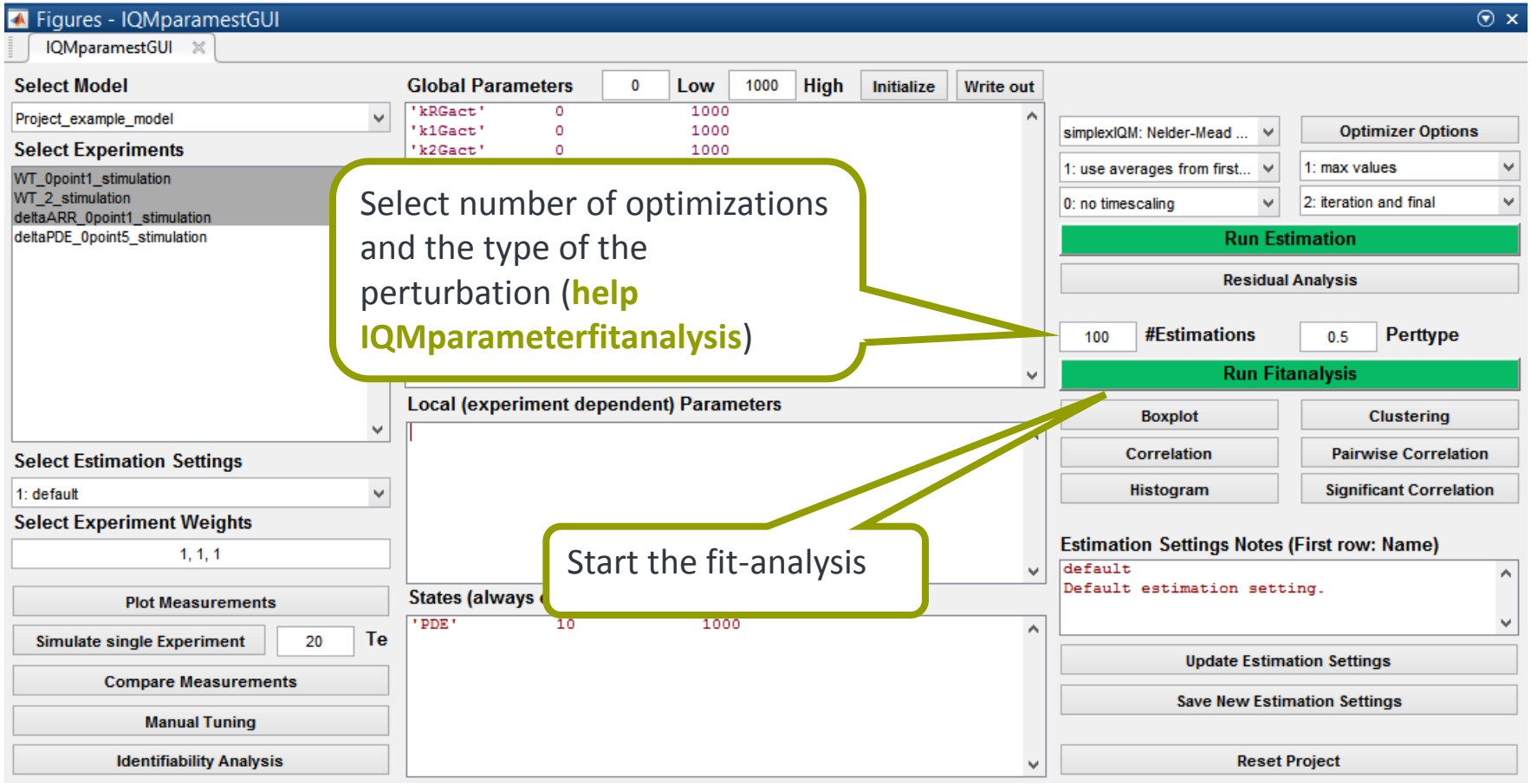
Parameter Fit Analysis



henning.schmidt@intiquan.com

IntiQuan

Parameter Fit Analysis

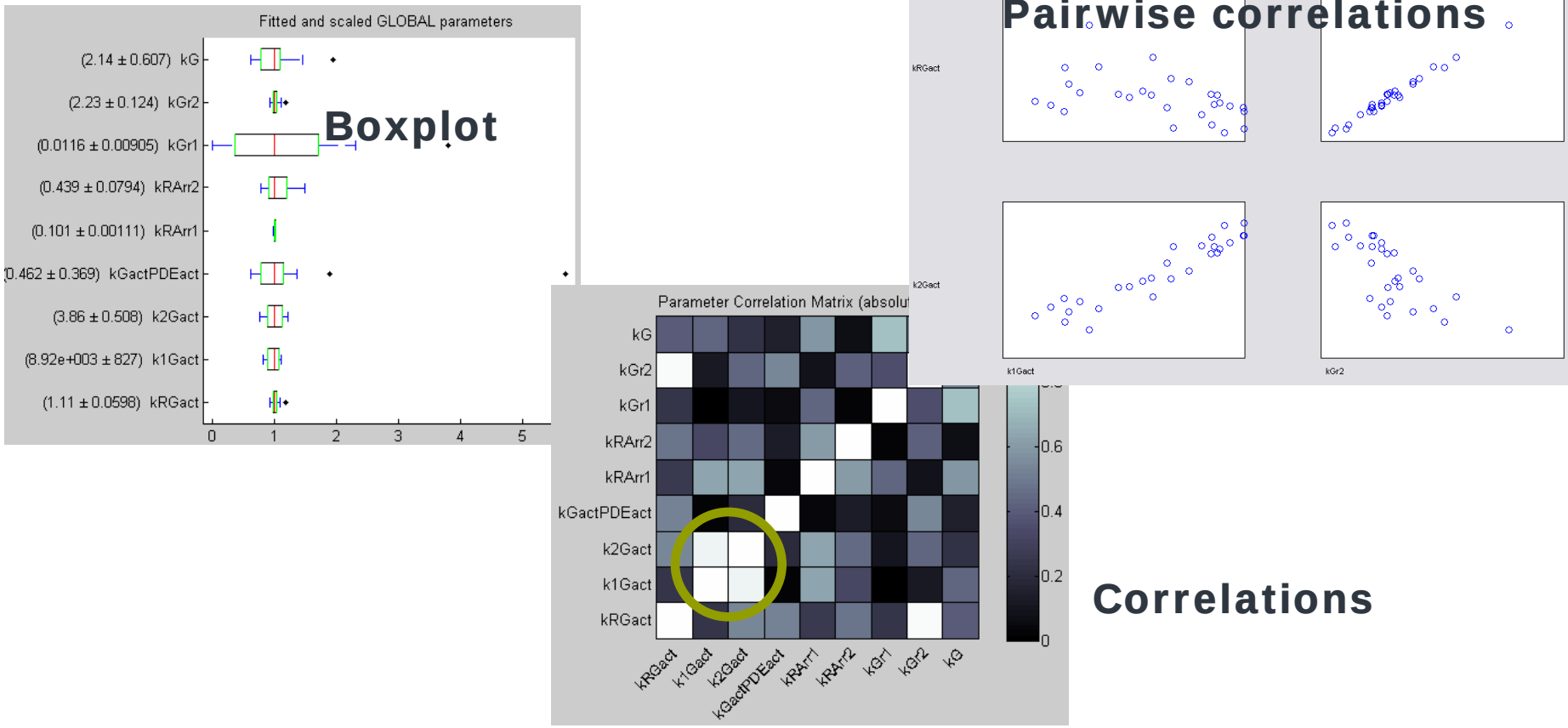


henning.schmidt@intiquan.com

IntiQuan

Parameter Fit Analysis

- Correlation analysis: Even here k1Gact and k2Gact are highly correlated



henning.schmidt@intiquan.com

IntiQuan

Non GUI Estimation etc.

- Parameter estimation does not require IQMparamestGUI
- All functions are available on the command line
- Using MATLABs Cell Mode => „comfortable“ estimation

- Create a **RunEstimation** script
  - In IQMparamestGUI choose „**Other**“->“**Create RunEstimation Script**“
  - Navigate to the **projectexample** folder
  - Select **RunEstimation** as name and save

- Close IQMparamestGUI

```
>> edit RunEstimation

% Delete line 11
% Change line 12 to sbp = IQMprojectSB('phototransduction_project_optimized');
```

henning.schmidt@intiquan.com

- Model Reduction

IQMreduceratexpressions

Schmidt, H. et al. (2008) Complexity Reduction of Biochemical Rate Expressions, Bioinformatics, doi: 10.1093/bioinformatics/btn035

henning.schmidt@intiquan.com

IntiQuan

Model Reduction

- Kinetic rate expressions can be very complex
- Not always they do need to be so complex in order to describe the behavior of interest

$$r = \frac{V_{max}\frac{[Glc_x]}{K_{Glc}}}{1+\frac{[Glc_x]}{K_{Glc}}+\frac{P\cdot\frac{[Glc_x]}{K_{Glc}}+1}{P\cdot\frac{[Glc]}{K_{Glc}}+1}\left(1+\frac{[Glc]}{K_{Glc}}+\frac{[G6P]}{K_{iG6P}}+\frac{[Glc]}{K_{Glc}}\frac{[G6P]}{K_{iiG6P}}\right)} - \frac{V_{max}\frac{[Glc]}{K_{Glc}}}{1+\frac{[Glc]}{K_{Glc}}+\frac{[G6P]}{K_{iG6P}}+\frac{[Glc]}{K_{Glc}}\frac{[G6P]}{K_{iiG6P}}+\frac{P\cdot\frac{[Glc]}{K_{Glc}}+1}{P\cdot\frac{[Glc_x]}{K_{Glc}}+1}\left(1+\frac{[Glc_x]}{K_{Glc}}\right)}.$$

$$r_{red} = \frac{K_4[Glc]+K_3[Glc_x]}{1+K_1[Glc][Glc_x]+K_2[G6P][Glc_x]}.$$

henning.schmidt@intiquan.com

Model Reduction

- The function **`IQMreducerateexpressions`** allows to stepwise reduce complex kinetic expressions

- Here we demonstrate its use based on the previous modeling example, where we unnecessarily used a Michaelis Menten term

$$r = \frac{V_{\max}\frac{[\mathrm{Glc_x}]}{K_{\mathrm{Glc}}}}{1+\frac{[\mathrm{Glc_x}]}{K_{\mathrm{Glc}}}+\frac{P\cdot\frac{[\mathrm{Glc_x}]}{K_{\mathrm{Glc}}}+1}{P\cdot\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}+1}\left(1+\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}+\frac{[\mathrm{G6P}]}{K_{\mathrm{iG6P}}}+\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}\frac{[\mathrm{G6P}]}{K_{\mathrm{iiG6P}}}\right)} - \frac{V_{\max}\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}}{1+\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}+\frac{[\mathrm{G6P}]}{K_{\mathrm{iG6P}}}+\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}\frac{[\mathrm{G6P}]}{K_{\mathrm{iiG6P}}}+\frac{P\cdot\frac{[\mathrm{Glc}]}{K_{\mathrm{Glc}}}+1}{P\cdot\frac{[\mathrm{Glc_x}]}{K_{\mathrm{Glc}}}+1}\left(1+\frac{[\mathrm{Glc_x}]}{K_{\mathrm{Glc}}}\right)}.$$

henning.schmidt@intiquan.com

Model Reduction

- Load the optimized project

```
>> sbp = IQMprojectSB('phototransduction_project_optimized')
```

- Run the model reduction

```
>> IQMreducerateexpressionsProject(sbp)
```

Works only when the symbolic toolbox is available

- **RESULT: Mass action kinetics is sufficient**

- More information

```
>> help IQMreducerateexpressionsProject
```

henning.schmidt@intiquan.com

- Simulation of IQMprojectsSB

Simulation of Projects

- Change into the „**Example Files/projectexample**“ folder

```
>> sbp = IQMprojectSB('phototransduction_project_optimized')   % import the project
```

- Compare measurements

```
>> IQMcomparemeasurements(sbp)    % simulate all experiments and model in the project
                                  % and compare results to measurements

>> help IQMcomparemeasurements
```

- Manual tuning

```
>> IQMmanualtuning(sbp)    % simulate selected experiments and model in the project
                           % and compare results to measurements + allow tuning

>> help IQMmanualtuning
```

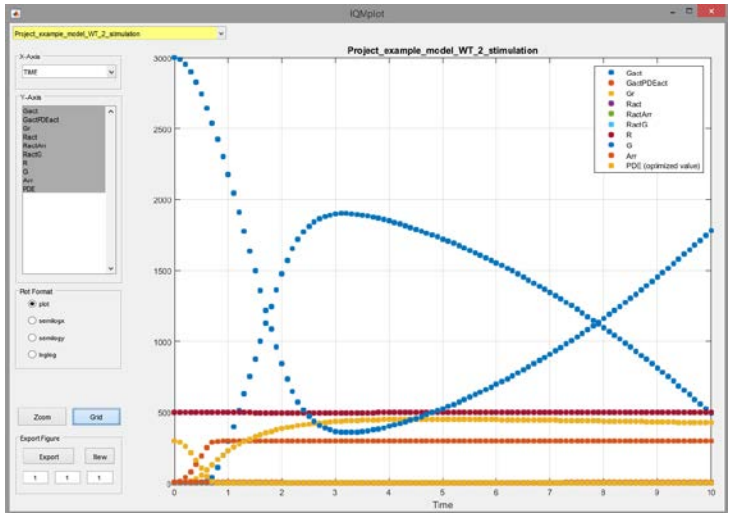
- These commands take a while to execute, since compilation is done

Simulation of Projects

- Simulating *in silico* experiments

```
>> model = IQMgetmodel(sbp,1)    % get first model from project sbp
>> experiment = IQMgetexperiment(sbp,2)   % get second experiment from project sbp
>> IQMinsilicoexp(model,experiment,[0:0.1:10])
```

- Per default the result is plotted



- Optionally, the result can be saved
  - CSV file
  - Excel file

- Or as **IQMmeasurement** be returned to the workspace

```
>> help IQMinsilicoexp
```

henning.schmidt@intiquan.com

Simulation of Projects

- Instead of extracting models and experiments in silico experiments can directly be performed on projects:

```
>> modelindex = 1              % first model from project sbp
>> experimentindex = 2         % second experiment from project sbp
>> IQMinsilicoexpproj(sbp,modelindex,experimentindex,[0:2:400])   % perform experiment
```

- Per default the result is **saved in a CSV measurement file**

- Optionally, the result can be
  - displayed using IQMplot
  - saved as an Excel measurement file
  - returned to the workspace as an **`IQMmeasurement`**

```
>> help IQMinsilicoexpproj
```

henning.schmidt@intiquan.com

- Commenting of projects

Commenting a Project

- A project folder can contain a `notes.txt` file
  - General information about the project
  - Focus of the project: why modeling, what to do with the model, etc.

- Every folder in a project can contain additional files. For example:
  - Word(etc.) documents in the model or experiment folder(s), describing things more in detail
  - Figures, spreadsheets, raw data files, ...

- In this way the complete information about a modeling project can be contained in the folder structure of an IQMprojectSB

- LIMITATION: Experiment folders should only contain Excel or CSV files that contain measurement data. However, these folders can contain additional folders in which all kinds of information can be stored.

Documenting the Validity of a Model

- A model alone is NOT very useful
  - SBML model
  - *.txt or *.txtbc model

- A model becomes useful if the following is known about it
  - For what purpose has the model been build?
  - Which experiments have been performed to generate data for fitting and validation?
  - Under what conditions have the experiments been performed?
  - Etc.

Documenting the Validity of a Model

- Where to store that information in the framework of the IQM Tools Suite?
  - In the projects
    - The notes.txt can contain the purpose of the model and general assumptions
    - The model files can contain additional information
    - The experiment descriptions define the performed experiments and the conditions under which these experiments have been performed
- Don‘t exchange models! Exchange projects!

Tutorial Goal: „You should now be able to“

- Use MEX/C-code models for fast simulation
- Calculate sensitivity trajectories with MEX models
- Set up your own parameter estimation project
  - Model description
  - Measurement data
  - Experiment descriptions
- Perform parameter estimation, identifiability analysis, etc.
- Analyze the resulting model

THE END

Thank you for your participation and interest!

The tutorial continues in Part 3 (IQM Tools Pro / Basic Pharmacometrics)

henning.schmidt@intiquan.com