Tutorial on IQM Tools Pro

Linking Systems Pharmacology Models to Clinical Data

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

Henning Schmidt, IntiQuan GmbH

Tutorial Outline

- **General introduction to the IQM Tools Suite**
- **Compliance mode**
- **Models: dosing, output and parameter definitions**
  - Define dosing inputs
  - Define observation outputs
  - Define parameters to estimate and to use as regression parameters
- **Dosing description and simulation**
  - Definition of dosing schemes
  - Simulation of dosing schemes

henning.schmidt@intiquan.com

Tutorial Outline

- **Import and export of SBML models**
  - General import / export
  - Handling cases where SBML models are incompletely defined
  - Handling cases where SBML IDs are randomly generated (e.g. Simbiology, CellDesigner)

- **Important things to consider for systems pharmacology models**
  - Bad examples
  - Suggestions for good modeling practice
    - Model features to avoid in order to allow use of standard NLME parameter estimation tools
    - Steady-state implementation
    - Clear input and output definitions

henning.schmidt@intiquan.com

Tutorial Outline

- **Interface to NLME Tools**
  - Conversion of IQMmodels to NONMEM and MONOLIX
  - Running NONMEM and MONOLIX from IQM Tools
  - Fitanalysis results – overview
  - Robustness analysis

- **General representation of clinical data**
  - Generalized data format for clinical data used in IQM Tools
  - Checking of data consistency
  - Conversion to task specific dataset format
  - Graphical exploration
  - Conversion to dataset for nonlinear mixed effect parameter estimation

- **Examples**

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(Ad)= -ka*Ad + INPUT1
d/dt(Ac)=  ka*Ad – CL/Vc*Ac
********** MODEL PARAMETERS
ka = 0.2
CL = 0.3
Vc = 6
********** MODEL VARIABLES
Cc = Ac/Vc
```

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“

- Understand and use the compliance mode – or switch it off
- Define IQMmodels with dosing inputs and simulate dosing scenarios
- Import models from SBML to IQMmodels, add inputs and outputs for simulation and NLME approaches
- Understand the requirements on model definition for use of NONMEM or MONOLIX as parameter estimation tools
- Convert models to NONMEM and MONOLIX and estimate parameters
- Understand and handle the general dataset specification

Tutorial Outline

- **General introduction to the IQM Tools Suite**
- Compliance mode
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- General representation of clinical data
- Examples

henning.schmidt@intiquan.com

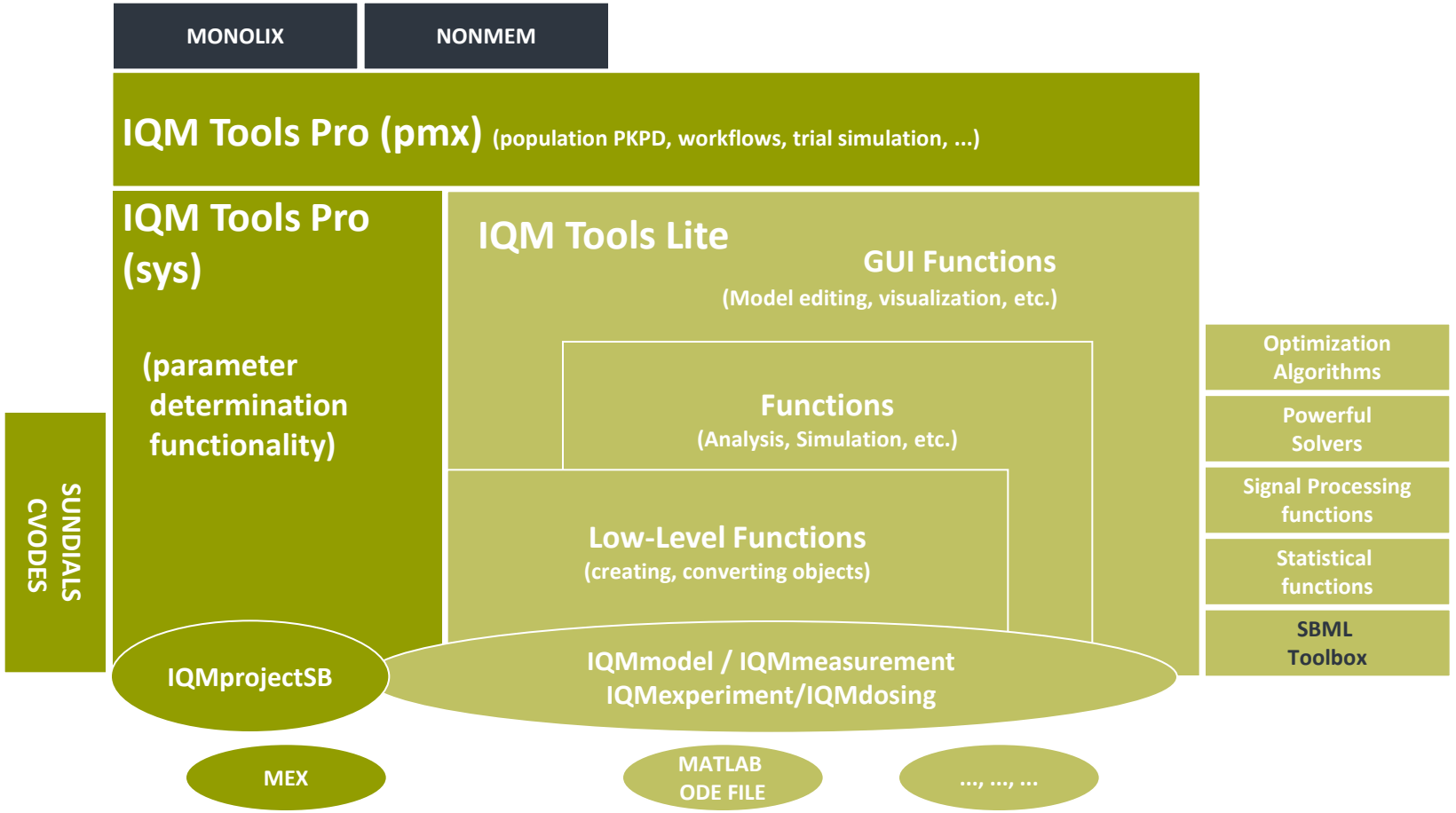
Introduction to the "IQM Tools Suite"

- 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)**
  - Optional but very useful: Parallel Toolbox to allow for better runtimes of the examples

- **Required 3rd party software**
  - SBML Toolbox (included in IQM Tools for Windows)
  - Monolix Version 4.2.3 (or later) (http://www.lixoft.org)
  - NONMEM Version 7.2 (or later) (http://www.iconplc.com)

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 3rd party software

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

- Update the **IQMpro/SETUP_PATHS_TOOLS_IQMPRO.m** file with the relevant information, as shown below for an example implementation

```
% NONMEM (currently tested version: 7.2)
PATH_SYSTEM_NONMEM_WINDOWS                  = 'nmfe73';
PATH_SYSTEM_NONMEM_UNIX                     = 'nmfe72';

% NONMEM PARALLEL
PATH_SYSTEM_NONMEM_PARALLEL_WINDOWS         = '';
PATH_SYSTEM_NONMEM_PARALLEL_UNIX            = 'nmfe72par';

% MONOLIX STANDALONE (version >= 4.3.2)
PATH_SYSTEM_MONOLIX_WINDOWS                 = 'C:\LOCAL\Monolix\Monolix432s\bin\Monolix.bat';
PATH_SYSTEM_MONOLIX_UNIX                    = '/CHBS/apps/dev_apps/Monolix/4.3.2/Standalone/Monolix432grid-Matlab2008/bin/Monolix.sh';

% MONOLIX STANDALONE PARALLEL (version >= 4.3.2)
PATH_SYSTEM_MONOLIX_PARALLEL_WINDOWS        = '';
PATH_SYSTEM_MONOLIX_PARALLEL_UNIX           = '';
```

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
- **Compliance mode**
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- General representation of clinical data
- Examples

Compliance Mode

- The Pharma World is highly regulated. For every modeler it is important to ensure that the complete modeling project (models, scripts, data, outputs) are well documented and all results are easily reproducible
- IQM Tools allows to annotate each produced figure and each produced table or text file by the username of the modeler, date and time, and the names of the scripts that were run to produce the output
  - This is accomplished by generating a .log file for each output (figure and text / table file)
  - The IQM Tools Reporting package will link this log information with the table or figure in the generated report
  - This feature is called „Compliance Mode“
- **The Compliance Mode can be turned on or off in the IQMlite/SETUP_PATHS_TOOLS_IQMLITE.m setup file.** When using IQM Tools for work on clinical projects, it should always be turned on.
- IQM Tools additionally allows a very structured approach to coding / organization of models, scripts, output, data that allows reproducibility of results by any person

Setup the Compliance Mode

- Change into the **IQMlite folder**
- Switch on the Compliance Mode by setting the corresponding variable to 1

```
% that this name is defined and if not in the list of functions that will
% lead to the output generation it will add it. If undefined, there will be
% an error, explaining that this variable needs to be set.
COMPLIANCE_OUTPUT_MODE                = 1;  % 1=enabled, 0=off
```

- **Or switch it off by setting the value to 0**

- The log files are created when figures and text files are generated by the following IQM Tools commands:
  - `IQMprintFigure`
  - `IQMwriteText2File`

Example for Compliance Mode

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

```
>> IQMnewscript('testscript')  % Creates a template analysis script
>> edit testscript.m
```

- „testscript.m“ is a template analysis script that as first step clears the workspace and installs IQM Tools and in a second step initializes the compliance mode

```
% The input argument to the "IQMinitializeCompliance" function needs to be
% the name of the script file.
IQMinitializeCompliance('testscript');
```

henning.schmidt@intiquan.com

Example for Compliance Mode

- Add the following text into the testscript.m

```
%% Here your code starts

% Write example text to text file
text = 'Hello World!';
IQMwriteText2File(text,'textfile.txt')

% Write example figure to PNG file
t = [0:1:100];
plot(t,sin(0.1*t));
IQMprintFigure(gcf,'figurefile','png')
```

- Execute the whole script by pressing „F5“
- => Output files and associated .log files are created

```
>> edit textfile.txt.log
>> edit figurefile.png.log
```

Example for Compliance Mode

- Comment out the IQMinitializeCompliance call

```
% The input argument to the "IQMinitializeCompliance" function needs to be
% the name of the script file.
% IQMinitializeCompliance('testscript');
```

- Rerun the script by pressing „F5“

```
Error using getFunctionCallInformationIQM (line 48)
Compliance mode is on. Please use the function "IQMinitializeCompliance" at the beginning of each main working script.
Do not use "clear all" within a script - or rerun the "IQMinitializeCompliance" function after each call to "clear all".
```

- An error appears, since the compliance mode is set to on but no call to IQMintializeCompliance has been done after the last „clear all“
- Edit the argument to the IQMinitializeCompliance call and rerun by „F5“

```
% The input argument to the "IQMinitializeCompliance" function needs to be
% the name of the script file.
IQMinitializeCompliance('wrongname');
```

```
Error using IQMinitializeCompliance (line 51)
Input argument to "IQMinitializeCompliance" does not match the filename of this script in which this command is run.
```

- The name of the script and the input argument to IQMinitializeCompliance need to match

henning.schmidt@intiquan.com

Additional Compliance / Reproducibility Considerations

- Typical pharmacometric modeling and analysis code is „messy“ and very difficult to audit
- IQM Tools allows a good organization of scripts, models, data, and outputs that is easy to understand
  - In the workflow tools IQM Tools imposes such a structure
  - In free modeling activities it is suggested that the modeler uses the following structure

    - **Root folder**
      - **„Scripts“:** Containing analysis scripts with numeric ordering, e.g.:
        - SCRIPT_01_data_preparation
        - SCRIPT_02_popPK
      - **„Data“:** The datasets
      - **„Models“:** The generated IQMmodels or NLME models
      - **„Output“:** The generated output (figures, tables, etc.)

This structure will be used in the following examples (where applicable)

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- **Models: dosing, output and parameter definitions**
  - Define dosing inputs
  - Define observation outputs
  - Define parameters to estimate and to use as regression parameters
- Dosing description and simulation
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- General representation of clinical data
- Examples

IQMmodel syntax linking the model to dosing inputs and observations in clinical data (for parameter estimation)

- Same syntax as for **IQMmodel**s
  - ODEs or biochemical representation can be used
- Additional elements
  - INPUT*: Allows to define dosing inputs – linking to doses in datasets
  - OUTPUT*: Allows to define outputs – linking to clinical observations in datasets

Example model

```
********** MODEL NAME
Linear two compartmental distribution PK model
********** MODEL STATES
d/dt(Ac) = -CL/Vc*Ac – Q/Vc*Ac + Q/Vp*Ap + INPUT1
d/dt(Ap) =           + Q/Vc*Ac - Q/Vp*Ap
********** MODEL PARAMETERS
CL = 20     % (L/hour)  Clearance
Vc = 32     % (L)       Central volume
Q  = 12     % (L/hour)  Intercompartmental clearance
Vp = 1450   % (L)       Peripheral volume
********** MODEL VARIABLES
Cc = Ac/Vc  % Calculation of plasma concentration
OUTPUT1 = Cc
```

henning.schmidt@intiquan.com

IQMmodel Syntax - INPUTs

- **INPUT Definitions in IQMmodels**
  - Please open the example file **inputExamples.m** in the folder **Example Files/MODEL_SYNTAX** and work it through
  - Example for a single input definition in an ODE based model (**model1.txt**)

```
********** MODEL STATES
d/dt(Ac)  = -VMAX*Cc/(KM+Cc) + F*INPUT1
```

  - Example for a single input definition in an model based on biochemical reaction equations (**model2.txtbc**)

```
********** MODEL REACTIONS
   => Ac : v_input
    vf = F*INPUT1

Ac =>  : v_clearance
    vf = VMAX*Cc/(KM+Cc)
```

  - Both models above are mathematically identical.
  - The **inputExamples.m** file contains also a more complex input example

IQMmodel Syntax - INPUTs

- A model can contain arbitrarily many input definitions
- INPUT* where "*": 1,2,3,4,5. Indices do not need to be sequential and do not need to start at 1, but they need to be numeric
- Input definitions are only allowed in differential equations and in reactions
- In the latter case the reaction name in the ODE is replaced by the reaction expression to have the input definition in the ODE
- A prefactor is allowed, e.g. 1.5*INPUT1 or (k1+k2)*INPUT2 or F*INPUT4 No postfactors are allowed. A "*" character has to stand between prefactor and INPUT* definition
- Input terms (INPUT* identifier and prefactor) are only allowed to be added (+) to the differential equation. But they can be the first term in the ODE, not requiring a leading "+" sign
- If a parameter with the name "INPUT*" is defined in the model, its value will not be changed during import. If the model does not contain an INPUT* parameter, it is added and set by default to "0". INPUT* is only allowed to be defined as a parameter. Not as a variable and not as a reaction
- Only parameters (and numerical values) are allowed to be used in the definition of INPUT* prefactors. But no states, variables and reactions
- INPUT* definitions are not allowed to be enclosed in any parentheses

IQMmodel Syntax - OUTPUTs

- **OUTPUT Definitions in IQMmodels**
  - Please open the example file **inputExamples.m** in the folder **Example Files/MODEL_SYNTAX** and work it through
  - Example for a single output definition in an IQMmodel (**model1.txt** and **model2.txtbc**)

```
********** MODEL VARIABLES
Cc      = Ac/Vc % (ng/ml) Plasma concentration
OUTPUT1 = Cc    % (ng/ml) Output variable
```

- The reason for the availability of these OUTPUT* definitions is that IQMtools uses these when interfacing IQM Tools with pharmacometric parameter estimation tools (NONMEM and MONOLIX)

- The right hand side of OUTPUT* definitions should NOT contain mathematical expressions, but only get assigned previously defined variables (in this example: Cc)

IQMmodel Syntax - OUTPUTs

- OUTPUT* where "*": 1,2,3,4,5, ... In contrast to INPUT* definitions, the OUTPUT indices NEED TO BE sequential, starting from 1. No number is allowed to be excluded
- Output definitions are only allowed to appear in the model variables section
- The right hand side of OUTPUT* definitions should NOT contain mathematical expressions, but only get assigned previously defined variables
- OUTPUT* variables are NOT allowed to depend on INPUT* components

IQMmodel Syntax – Parameter Info

- The last elements of the IQMmodel syntax are flags that indicate if a certain parameter should be estimated or obtained from a dataset (called: regression parameter)
- This is done by adding in the comment section of a parameter the reserved words **<estimate>** or **<regression>**
- If a parameter is neither estimated nor provided as regression parameter, its value will be kept on the value defined in the model itself

Example Files / Model_Syntax / model_estimation.txt

```
********** MODEL PARAMETERS
% PK parameters obtained from dataset
F_subcut  = 0.5     % <regression> (.)
CL        = 0.303   % <regression> (L/day)
Vc        = 2.83    % <regression> (L)
Q         = 0.724   % <regression> (L/day)
Vp        = 4.43    % <regression> (L)
VMX       = 0.5     % <regression> (mg/day)
KM        = 1.86    % <regression> (ug/ml)

% PD parameters to be estimated
BASELINE = 1        % (.)
kdeg      = 0.1     % <estimate> (1/day)
EMAX      = 1       % <estimate> (.)
EC50      = 1       % <estimate> (ug/ml)
```

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- Models: dosing, output and parameter definitions
- **Dosing description and simulation**
  - Definition of dosing schemes
  - Simulation of dosing schemes
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- General representation of clinical data
- Examples

henning.schmidt@intiquan.com

IQMdosing Syntax

- The IQM Tools allow to define dosing schedules in a simple but efficient format
- The different dosing "inputs" then can be automatically applied to the inputs, defined in IQMmodels, using the INPUT* identifier
- In this section we will explain how to define dosing schemes and how to merge them with models to simulate the desired dosing scheme on the desired model

IQMdosing Syntax

- An example for a dosing scheme is the following

```
********** INPUT1
type:           INFUSION
time:           0       % (days) time for first application
deltaT:         14      % (days) time inbetween applications
nr_repetitions: 5       % number of applications
D:              10      % (unit) dose
Tinf:           2/24    % (days) 2 hours
```

- This dosing scheme defines that an INPUT1, defined in the model, should be implementing an infusion with a 14 day dosing interval, starting at time 0 with 5 repetitions, each dose 10 units and the infusion duration should be 2 hours

IQMdosing Syntax

- Dosing schedules can be defined in several different ways, also different doses can be given at different times. The following dose-application-types can be defined:
  - Bolus
  - Infusion (both defining infusion rate or infusion time)
  - 0th-order Absorption
  - 1st-order Absorption
- Optionally, each type can get a lag time assigned to it. Each dosing definition can define single doses or multiple doses

IQMdosing Syntax

- Dosing definitions are contained in dosing files with the extension ".dos"
  - ASCII text files which have a certain format and can contain an arbitrary number of input definitions
  - Limitation: Each input definition needs to have a different name. For example "INPUT1" is not allowed to be defined more than once

- Combining Models with Dosing Schedules
  - The underlying idea is that a model, containing inputs (INPUT1, INPUT2, etc.) can be merged with a dosing description that defines these inputs to implement the corresponding dosing definitions. The result of this merge is again a model that now can be simulated

  - Please open the example file **dosingExamples.m** in the folder **Example Files/Dosing_Syntax** and work it through

IQMdosing Syntax - Example

- Assume you have the following model
- And assume further that you want to simulate a first order absorption in the central compartment Ac
- Then this is your dosing description

```
********** MODEL STATES
d/dt(Ac)  = -VMAX*Cc/(KM+Cc) + F*INPUT1
********** MODEL PARAMETERS
VMAX = 1      % (ug/hour)
KM   = 1      % (ng/ml)
Vc   = 1      % (L)
F    = 0.6    % (.)
********** MODEL VARIABLES
Cc      = Ac/Vc % (ng/ml)
OUTPUT1 = Cc    % (ng/ml)
```

```
********** INPUT1
type:           ABSORPTION1
time:           0  % (hours)
D:              10 % (ug)
ka:             1  % (1/hour)
```

- To merge model with dosing definition we need to
  - 1) Import the model
  - 2) Import the dosing definition
  - 3) Merge model with dosing definition to obtain a model for simulation

henning.schmidt@intiquan.com

IQMdosing Syntax - Example

- Change into the **Example Files/Dosing_Syntax** folder

```
>> model = IQMmodel('model1.txt');                % Import of model
>> dosing = IQMdosing('dosing2.dos');             % Import of dosing description
>> modeldosing = IQMmergemoddos(model,dosing);    % Merge model and dosing description:
```

- Note the added elements and the naming of ka_input1, ...

```
********** MODEL STATES
d/dt(Ac) = -VMAX*Cc/(KM+Cc)+vAbsorption_input1  %(ug/hour)
d/dt(Comp_input1) = input1-vAbsorption_input1
********** MODEL PARAMETERS
VMAX = 1  %(ug/hour) Maximum rate of elimination
KM = 1    %(ng/ml)   At this concentration elimiation rate is half VMAX
Vc = 1    %(L)       Central volume
F = 0.59999999999999998  %(.)       Relative bioavailability
Dose_input1 = 10
Time_input1 = 0
DeltaT_input1 = 0.0001
ka_input1 = 1
********** MODEL VARIABLES
Cc = Ac/Vc  %(ng/ml) Plasma concentration
OUTPUT1 = Cc  %(ng/ml) Output variable

vAbsorption_input1 = ka_input1*Comp_input1
input1 = +F*Dose_input1/DeltaT_input1 *
          piecewiseIQM(1,andIQM(ge(time,Time_input1),lt(time,Time_input1+DeltaT_input1)),0)
```

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- **Import and export of SBML models**
  - General import / export
  - Handling cases where SBML models are incompletely defined
  - Handling cases where SBML IDs are randomly generated (e.g. Simbiology, CellDesigner)
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- General representation of clinical data
- Examples

henning.schmidt@intiquan.com

Import of SBML Models

- SBML Level Version 1,2,3 & 4 models can be imported
- **The SBML Toolbox needs to be present**
- Change into the "**Example Files**" folder

```
>> model = IQMmodel('CellCycle.xml')    % IQMmodel additionally also import SBML models
    IQMmodel
    ========
    Name: CellCycle
    Number States:              13
    Number Variables:           2
    Number Parameters:          41
    Number Reactions:           23
    Number Functions:           0

>> IQMedit(model)
```

Increase simulation time to 400 and click "Simulate"

henning.schmidt@intiquan.com

Import of SBML Models

Import of Incompletely Defined SBML Models

- Per default IQM Tools requires an SBML model to be completely defined before it is correctly imported

- To allow the user to import incompletely defined models the following optional call to IQMmodel exists:

```
% File located in the “Example Files” folder
>> model = IQMmodel('SBMLfileIncomplete.xml',1);
>> IQMedit(model)
```

  - `IQMmodel` then does not require that an SBML model is fully defined in terms of parameter values, rate equations, kinetic parameters, etc.
  - **However, when trying to „Exit“ IQMedit or IQMeditBC model correctness is checked. To still exit on Windows click the red cross in the upper right corner of the window. On Unix/Linux/Mac also click the corresponding symbol ...**

Import of SBML Models

Name to ID Conversion

- 'Name' to 'id' conversion
  - E.g.: Not all models in Biomodels.net have informative IDs, and CellDesigner or Simbiology choose the ids of certain elements (e.g., species and reactions) automatically and the user only can choose names that need not be unique

```
********** MODEL REACTIONS
reaction_0000001 = compartment_0000002 * (parameter_0000027 * parameter_0000021 +
       parameter_0000031)
reaction_0000002 = compartment_0000001 * (parameter_0000028 * parameter_0000020 +
       parameter_0000032)
reaction_0000003 = compartment_0000001 * delay(parameter_0000029 * parameter_0000022 +
       parameter_0000034, parameter_0000039)
reaction_0000004 = compartment_0000001 * (parameter_0000030 * (power(species_0000009, 2) /
       (power(species_0000009, 2) + power(parameter_0000010, 2))) * (power(parameter_0000008, 2) /
       (power(species_0000007, 2) + power(parameter_0000008, 2))) + parameter_0000033)
```

  - In IQM Tools the ids are used as names for states, variables, etc. (since they are unique)
  - => an optional call to IQMmodel (see box below) allows to use the SBML names instead of the IDs, but making them unique by numerical extensions

```
>> model = IQMmodel('SBMLfileIncomplete.xml',1,1);
```

  - More information, see the help text:

```
>> help IQMmodel
```

henning.schmidt@intiquan.com

Export of SBML Models

- Export only to SBML Level 2 Version 1
- IQMmodels do not necessarily contain all needed information for export
  - Additional information is required

```
>> help IQMexportSBML          % for more information on the additional information
```

- Location of additional information (in IQMmodel internal data structure)

states.type	**algebraic.type**
states.compartment	**algebraic.compartment**
states.unittype	**algebraic.unittype**
parameters.type	**variables.type**
parameters.compartment	**variables.compartment**
parameters.unittype	**variables.unittype**

***.type:**	'isSpecie'	'isParameter'	'isCompartment'
***.compartment:**	compartment	-	outside compartment
***.unittype:**	'amount' or 'concentration'	-	-

henning.schmidt@intiquan.com

Export of SBML Models

Additional Information in the TEXT Description

- Additional information can be added in the text / textbc format

```
>> help IQMedit
>> help IQMeditBC
```

- After SBML import the additional information is already present

```
********** MODEL NAME
example

********** MODEL NOTES


********** MODEL STATES
d/dt(s1) = -re1 {isSpecie:cytosol:amount} % comment
d/dt(s2) = +re1 {isSpecie:cytosol:amount}
d/dt(s3) = -re2+re4 {isSpecie:cytosol:amount}
d/dt(s4) = +re2-re5 {isSpecie:cytosol:amount}
d/dt(s5) = (-re3)/nucleus {isSpecie:nucleus:concentration}
d/dt(s6) = +re3 {isSpecie:nucleus:amount}

s1(0) = 1

********** MODEL PARAMETERS
s7 = 1 {isParameter} % comment
s8 = 1 {isParameter}
cytosol = 1 {isCompartment:}
nucleus = 0.1 {isCompartment:cytosol}


********** MODEL VARIABLES


********** MODEL REACTIONS
re1 = 3 * s1 - 2 * s2 {reversible} % comment
re2 = s3
re3 = s4 * (s5 - s6) {reversible}
re4 = 1
re5 = s4
```

henning.schmidt@intiquan.com

Export of SBML Models

Automatic Determination of Additional Information

- Even if no additional information is given, the toolbox is under certain conditions able to determine the correct information.

novaktyson1.txt

```
d/dt(Cyclin) = R1-R2-R3
d/dt(YT) = R4-R5-R6-R7+R8+R3
d/dt(PYT) = R5-R8-R9-R10+R11
d/dt(PYTP) = R12-R11-R13-R14+R9
d/dt(MPF) = R6-R4-R12-R15+R13
d/dt(Cdc25P) = R16
d/dt(Wee1P) = R17
d/dt(IEP) = R18
d/dt(APCstar) = R19
```

=> Species in Amount units

novaktyson2.txt

```
d/dt(Cyclin) = (R1-R2-R3)/compartment
d/dt(YT) = (R4-R5-R6-R7+R8+R3)/compartment
d/dt(PYT) = (R5-R8-R9-R10+R11)/compartment
d/dt(PYTP) = (R12-R11-R13-R14+R9)/compartment
d/dt(MPF) = (R6-R4-R12-R15+R13)/compartment
d/dt(Cdc25P) = (R16)/compartment
d/dt(Wee1P) = (R17)/compartment
d/dt(IEP) = (R18)/compartment
d/dt(APCstar) = (R19)/compartment
```

=> Species in Concentration units

- Remember: SBML assumes reaction rates defined in amount/time

- Here it is important that the elements on the RHS are defined under the „**MODEL Reactions**“ header

henning.schmidt@intiquan.com

Export of SBML Models

Automatic Determination of Additional Information

- Example

```
>> model = IQMmodel('novaktyson1.txt')  % file located in ”example files” folder
>> IQMexportSBML(model)                 % choose sbmlmodel1 as name for the file

>> model = IQMmodel('novaktyson2.txt’)  % file located in ”example files” folder
>> IQMexportSBML(model)                 % choose sbmlmodel2 as name for the file
```

- Have a look at both files (novaktyson1.txt and ...2.txt)

- Have a look at the exported SBML files
  - => species, reactions, compartments and unittypes are correctly determined

- This does not work for all possible model definitions, and thus the manually added information will override the automatically generated one

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- Import and export of SBML models
- **Important things to consider for systems pharmacology models**
  - Bad examples
  - Suggestions for good modeling practice
    - Model features to avoid in order to allow use of standard NLME parameter estimation tools
    - Steady-state implementation
    - Clear input and output definitions
- Interface to NLME Tools
- General representation of clinical data
- Examples

henning.schmidt@intiquan.com

Disclaimer

- „Bad Examples“ does not mean that the models are „bad“
- It is used here in the context of model „re-usability“ and potential to linking with NLME parameter estimation tools

Some bad ... examples

- Change into the folder **Example Files/Bad_Examples**
- It conains 3 SBML models for the glucose-insulin system

- Lets load the first one and export it to an IQMmodel ODE representation

```
>> model = IQMmodel('BIOMD0000000379.xml',0,0,1);
>> IQMcreateTEXTfile(model,'BIOMD379');
>> edit BIOMD379.txt
```

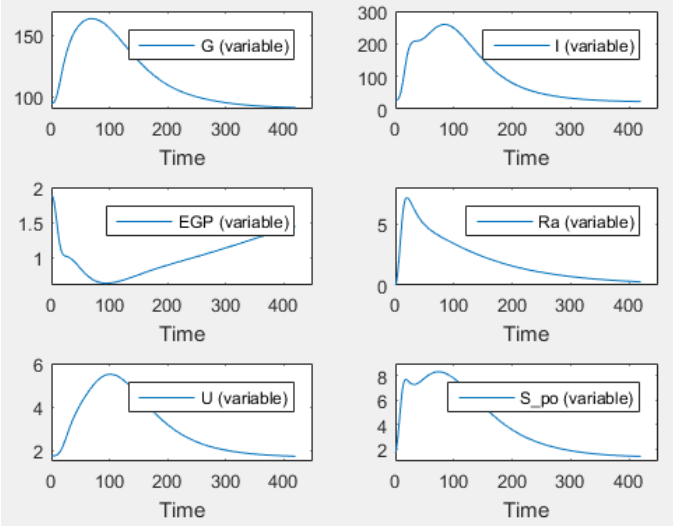
- Lets simulate the model

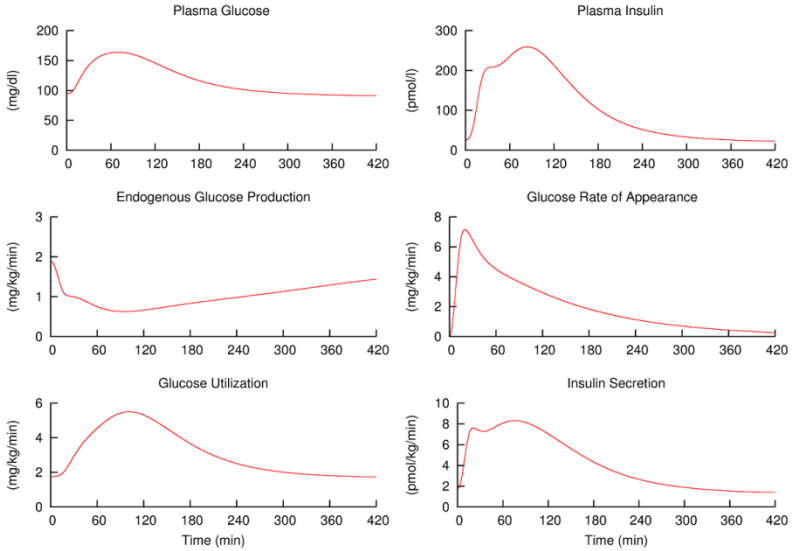
```
>> IQMsimulate(model,420)
```

henning.schmidt@intiquan.com

Some bad ... examples

- The model reproduces the plots in the paper and on the biomodels.net webpage (http://www.ebi.ac.uk/biomodels-main/BIOMD0000000379)





- So what is the problem?

Some bad ... examples

- The model is a „this particular simulation only“ model

- No „Inputs“ to the model are defined that would allow to apply „glucose“ or „insulin“ doses
- Specific time course of simulation coded in initial conditions and parameters

```
********** MODEL STATES
d/dt(G_p) = (EGP+Ra-E-U_ii-k_1*(G_p/Compartment1)+k_2*(G_t/Compartment1))*Compartment1
d/dt(G_t) = (-U_id+k_1*(G_p/Compartment1)-k_2*(G_t/Compartment1))*Compartment1
d/dt(I_l) = (-m_1*(I_l/Compartment1)-m_3*(I_l/Compartment1)+m_2*(I_p/Compartment1)+S)*Compartment1
d/dt(I_p) = (-m_2*(I_p/Compartment1)-m_4*(I_p/Compartment1)+m_1*(I_l/Compartment1))*Compartment1
d/dt(Q_sto1) = (-k_gri*(Q_sto1/Compartment1))*Compartment1
d/dt(Q_gut) = (-k_abs*(Q_gut/Compartment1)+k_empt*(Q_sto2/Compartment1))*Compartment1
d/dt(I_1) = (-k_i*((I_1/Compartment1)-I))*Compartment1
d/dt(I_d) = (-k_i*((I_d/Compartment1)-(I_1/Compartment1)))*Compartment1
d/dt(X) = (-p_2U*(X/Compartment1)+p_2U*(I-I_b))*Compartment1
d/dt(I_po) = (-gamma*(I_po/Compartment1)+S_po)*Compartment1
d/dt(Y) = (-alpha*((Y/Compartment1)-beta*(G-G_b)))*Compartment1
d/dt(Q_sto2) = (-k_empt*(Q_sto2/Compartment1)+k_gri*(Q_sto1/Compartment1))*Compartment1

G_p(0) = 178
G_t(0) = 135
I_l(0) = 4.5
I_p(0) = 1.25
Q_sto1(0) = 78000
Q_gut(0) = 0
I_1(0) = 25
I_d(0) = 25
X(0) = 0
I_po(0) = 3.6
Y(0) = 0
Q_sto2(0) = 0
```

- Potential for improvement:
  - Code initial conditions such that model is in steady-state
  - Use input definitions to allow application of glucose and/or insulin, etc.
  - Document the model in the Notes section with respect to time units, meaning of model components ... What is Y?

Some bad ... examples

- Lets load the second model and export it to an IQMmodel ODE representation

```
>> model = IQMmodel('Cobelli Glucose-Insulin System.xml', 1,1,1);
>> IQMcreateTEXTfile(model,'Cobelli');
>> edit Cobelli.txt
```

- Lets simulate the model

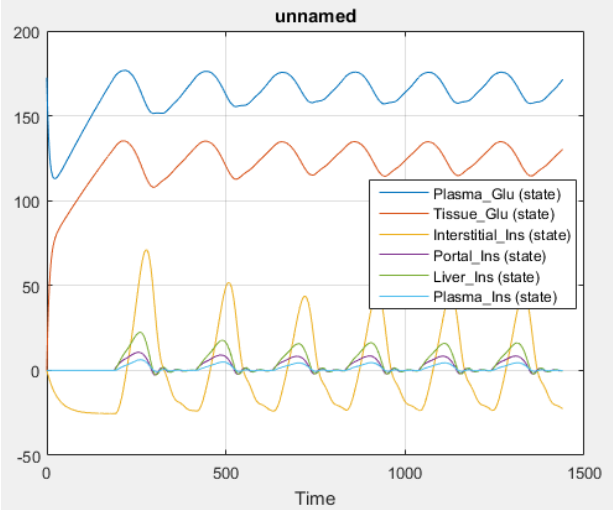
```
>> IQMsimulate(model,24*60)
```

- **Does it work? .... NO**

- The reason for not working is that the variables are not ordered correctly, for example, the variable „a“ is defined after it is used on the RHS for the variable „kempt“
  - SBML does not enforce ordering, but many simulation tools do ... Including the parameter estimation tools NONMEM and MONOLIX

Some bad ... examples

- Once the order of the variables has been manually corrected (several iterations), the model can be simulated
- This model as well is a „this particular simulation only“ model, although some dose might be possible to apply but the model lacks internal documentation to understand what could be done or not



- Another issue of the model wrt potential use in NLME parameter estimation scenarios is the presence of Events
  - Neither NONMEM nor MONOLIX can handle events
  - For this model here all events, except the Apply_dose event could have been coded with the SBML piecewise statement, which in NONMEM and MONOLIX can be translated to IF THEN ELSE statements

```
********** MODEL EVENTS
Apply_dose = andIQM(gt(time,timeSmall),gt((Dose/Glucose_appearance),0*(Dose/Glucose_appearan
Insulin_secretion_activation = andIQM(gt(Plasma_Glu_Conc_Rate,0*Plasma_Glu_Conc_Rate),gt(Pla
Insulin_secretion_deactivation = orIQM(le(Plasma_Glu_Conc_Rate,0*Plasma_Glu_Conc_Rate),lt(Pl
Glucose_insulin_signal_activation = ge(beta*(Plasma_Glu_Conc-Basal_Plasma_Glu_Conc),-Basal_I
Glucose_insulin_signal_deactivation = lt(beta*(Plasma_Glu_Conc-Basal_Plasma_Glu_Conc),-Basal
Renal_excretion_activation = gt((Plasma_Glu/Glucose_appearance),ke2),Glu_Excretion_Mode,1
Renal_excretion_deactivation = le((Plasma_Glu/Glucose_appearance),ke2),Glu_Excretion_Mode,0
Reduced_beta_cell_responsivity_after_lunch = ge(time,timeLunch),beta_par,0.75*beta
Reduced_insulin_sensitivity_after_evening_meal = ge(time,timeEveningMeal),Vmx_par,0.75*Vmx
```

henning.schmidt@intiquan.com

IntiQuan

Some bad ... examples

- Lets load the third model and export it to an IQMmodel ODE representation

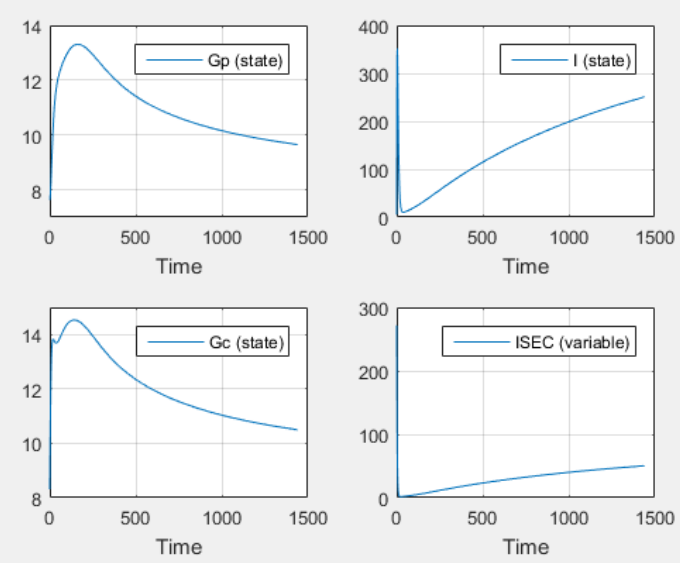
```
>> model = IQMmodel('MODEL1112110004_Silber2007', 0,0,1);
>> IQMcreateTEXTfile(model,'Silber');
>> edit Silber.txt
```

- Lets simulate the model

```
>> IQMsimulate(model,24*60)
```

- The model simulates fine

- **Whats the problem?**



henning.schmidt@intiquan.com

IntiQuan

Some bad ... examples

- **It looks like the ODE for GE2 is wrong**
  - **Missing „*GE2“**
- The model is not yet curated on Biomodels.net, which might explain the problem

```
********** MODEL STATES
d/dt(Gp) = Q/Vg*Gc-Q/Vp*Gp
d/dt(GE1) = KGE1*(Gc/Vg)-KGE1*GE1
d/dt(GE2) = KGE2*(Gc/Vg)-KGE2*GE2
d/dt(Gc) = GPROD+Q/Vp*Gp-(CLG/Vg+CLGI/Vg*IE+Q/Vg)*Gc
d/dt(I) = ISEC-CLI/VI*I
d/dt(IE) = KIE*(I/VI)-KIE*IE
d/dt(IFPS) = -KIS*IFPS

Gp(0) = Gss*Vp
GE1(0) = Gss
GE2(0) = Gss
Gc(0) = Gss*Vg
I(0) = Iss*VI
IE(0) = Iss
IFPS(0) = FPS
```

- The model again is a „this simulation only model“ - but the closest of the three examples to a useful one for later linking to NLME parameter estimation
- Input definitions missing as well
- The model code might have issues

henning.schmidt@intiquan.com

Suggestions for good modeling practice

To allow for using the model with NLME parameter estimation tools

- Use state, parameter, variable and reaction names that somewhat explain what these components are
  - For export to NONMEM these names should not be too long. MONOLIX can handle long names and anyway will be the better choice for systems pharmacology type of models, due to its far better numerical properties
  - For export to NONMEM, do not define reactions in the IQMmodel. For MONOLIX this is possible
  - Names of the parameters that you might want to consider for parameter estimation should not contain an underscore ('_')
- Do not use Events in the model
  - Often they can be avoided by using the piecewiseIQM statement, which does the same as the SBML piecewise function. For NONMEM and MONOLIX these are converted to IF THEN ELSE statements
- Define initial conditions for the state variables to ensure the model is in steady-state
  - Use non-numerical initial conditions to make them dependent on the parameterization of the model
  - Choose them such that the model is in steady state if all external stimuli / inputs of interest are 0
- Add well defined inputs to your model that allow to apply "external stimuli" to perform simulations of different scenarios. These inputs might later be defined from an NLME dataset for parameter estimation purposes
- Document the purpose of your model in the Notes section, including time units, units of inputs and outputs

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- **Interface to NLME Tools**
  - Conversion of IQMmodels to NONMEM and MONOLIX
  - Running NONMEM and MONOLIX from IQM Tools
  - Fitanalysis results – overview
  - Robustness analysis
- General representation of clinical data
- Examples

henning.schmidt@intiquan.com

Conversion of IQMmodels to NONMEM and MONOLIX

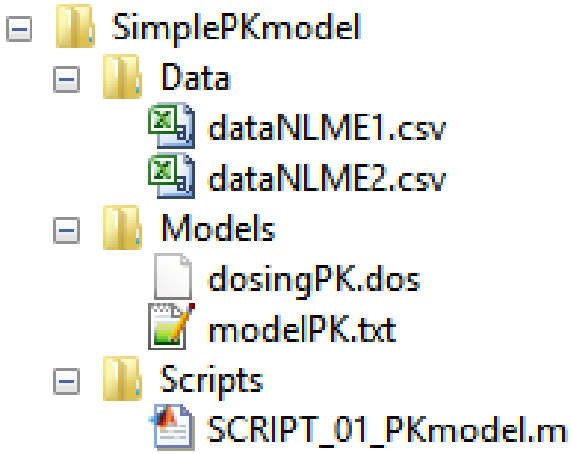
- IQM Tools allows to easily generate MONOLIX and NONMEM code for parameter estimation

- In the following several examples will be provided, based on
  - Theophylline data

- The goal in this tutorial is not to perform a complete model building, but to show how a model of interest can be converted to the parameter estimation tool of choice and perform the parameter estimation
  - We limit ourselves to a PK model
  - In later tutorials PKPD models, regression parameters, etc. will be covered

Theophylline Example

- Change into **Example Files/SimplePKmodel** folder



- Have a look at the
  - two data files
  - modelPK.txt file
  - dosingPK.dos file

- The two data files contain the same quantitative information, however the dataNLME1.csv file contains additional annotation that you might consider quite useful for the purpose of understanding of its contents
- The modelPK.txt file contains the structural PK model that we want to convert to NONMEM and MONOLIX
- The dosingPK.dos file contains information about the type of INPUT1 in the model

Required Dataset Columns (dataNLME2.csv)

Column Name	Meaning
IXGDF	Indices for records in the dataset. If dataset derived from a general dataset format each entry on IXGDF corresponds to the entry for IXGDF for this record in the general data format. If not derived, column still required.
ID	Unique numeric subject ID, used by NONMEM and MONOLIX
USUBJID	Unique subject ID, used in the clinical database
TIME	Time of event in a row with 0 determining time of first dose
TIMEPOS	Time of event in a row with 0 the time of the first event (needed for NONMEM, since crashing for negative times)
NT	Nominal time of the event (per protocol). Useful for binning
YTYPE	0 for dose events, otherwise containing the number of the model OUTPUT with which this observation should be compared
DV	Observed value of this observation
CENS	If 1 then observation assumed to be below lower limit of quantification. DV then needs to contain the LLOQ. Otherwise CENS=0
MDV	=1 for doses, =0 for observations, =1 for „missing observations“ or observations that should not be considered for parameter estimation
EVID	=1 for doses, =0 for observations
AMT	=0 for observations, =value of dose for dosing events
ADM	=0 for observations, otherwise containing the number of the model INPUT to which this dose is to be applied
RATE	Rate of dose, 0 for bolus, AMT/infusion time for infusion, etc.

Useful Dataset Columns (dataNLME1.csv)

Column Name	Meaning
NAME	Name of the event in a row. Unique per YTYPE. Gives information about the actual readout. YTYPE=1 does not say much. NAME = „Theophylline Concentration“ is more informative. Will be used by IQM Tools, e.g., for annotation of simulations, VPCs, etc. The reason for „:::“ in the NAME column is that NONMEM is unable to deal with spaces in text. IQM Tools will replace „:::“ by spaces for display purposes
UNIT	Unit of the observation or dose in a row. Will be used by IQM Tools, e.g., for annotation of simulations, VPCs, etc.
TIMEUNIT	Unit of all time related information in the dataset (including RATE). Will be used by IQM Tools, e.g., for annotation of simulations, VPCs, etc.
TRTNAME	Treatment group name. In the Theopylline Example only a single treatment group is present, but still, it is informative. Will be used by IQM Tools, e.g., for annotation of stratified VPCs
TRT	Numeric equivalent to TRTNAME, can be used as a covariate.
ROUTE	Information about the route of administration

- These additional columns make the dataset more complete and self explaining
- In Part 5 of the tutorials we will learn about the general dataset format that is used by IQM Tools, which contains even more additional and very useful columns
- It is sad but true that most data files in the pharmacometric world look like dataNLME2.csv

The Model

- The modelPK.txt file is a standard IQMmodel, containing
  - INPUT1 (to be linked to doses in data with ADM=1)
  - OUTPUT1 (to be linked to observation records with YTYPE=1)
  - 3 parameters to be estimated (<estimate>)

```
1   ********** MODEL NAME
2   modelPK
3   ********** MODEL NOTES
4   Example model for IQM Tools Tutorial
5   Simple linear one compartment PK model with 1st order absorption.
6   ********** MODEL STATES
7   d/dt(Ad) = -ka*Ad + INPUT1
8   d/dt(Ac) =  ka*Ad - CL/Vc*Ac
9   ********** MODEL PARAMETERS
10  ka = 1    % <estimate>
11  CL = 3    % <estimate>
12  Vc = 60   % <estimate>
13  ********** MODEL VARIABLES
14  Cc = Ac/Vc
15  OUTPUT1 = Cc
```

henning.schmidt@intiquan.com

Theophylline Example

- Change into **Example Files/SimplePKmodel/Scripts** folder
- Open **SCRIPT_01_PKmodel.m** in the MATLAB editor
  - This script follows the „Compliance Guideline“ of IQM Tools
    - Clean MATLAB workspace, path, etc
    - Install IQM Tools
    - Setup Compliance Mode

- Please update line number 12 with the path of your IQM Tools installation:

```
10      % In the next line, please enter the path to your IQM Tools folder. It can
11      % be a relative or an absolute path.
12 -    PATH_IQM            = 'D:\IntiQuan Modeling Tools\01 IQM Tools Suite';
```

- Execute lines 4-16 in the script (highlight them and press „F9“)

henning.schmidt@intiquan.com

Theophylline Example

- The following code in **SCRIPT_01_PKmodel.txt** defines the minimally needed information for model conversion to MONOLIX and NONMEM

```
%% Define minimally needed information for NLME model generation
model                        = '../Models/modelPK.txt';
dosing                       = IQMcreateDOSING('BOLUS');
data                         = [];
data.dataRelPathFromProject = '../../Data';
data.dataFileName            = 'dataNLME1.csv';
data.dataHeaderIdent         = IQMgetNLMEdataHeader('../Data/dataNLME1.csv');
```

model	Relative path from the Scripts folder to the IQMmodel text file that should be used for conversion.
dosing	We need to define which INPUT type is desired. In the model only a single input INPUT1 is defined. Underlying assumption: INPUT1 is a bolus into the dosing compartment. Thus „dosing“ needs to have as first and only type „BOLUS“.
data	This is a MATLAB structure which contains 3 different fields related to the dataset location and its structure.
data.dataRelPathFromProject	Defined the relative path from the desired project folder location to the dataset. We plan to create the NLME project in ../Models/“projectname“.
data.dataFileName	Name of the dataset to be used (without path).
data.dataHeaderIdent	Cell-array with identifiers, defining the meaning of the columns in the dataset.

- **Execute the code above**

data.dataHeaderIdent

- In this example:

  data.dataHeaderIdent = {ID,IGNORE,IGNORE,IGNORE,TIME,TIMEPOS,IGNORE,IGNORE,YTYPE,IGNORE,DV,IGNORE,CENS,MDV,EVID,AMT,ADM,IGNORE,RATE,IGNORE,IGNORE,IGNORE,IGNORE}

- MONOLIX users will understand what it means
- If you are a NONMEM user, then please have a look in the MONOLIX manual

- The good thing is that the function **IQMgetNLMEdataHeader** generates this header automatically and correctly

```
>> help IQMgetNLMEdataHeader
```

```
Column name   | Inferred Type
------------------------------
ID            | ID
USUBJID       | IGNORE
TRT           | IGNORE
TRT_NAME      | IGNORE
TIME          | TIME
TIMEPOS       | TIMEPOS
NOMINAL_TIME  | IGNORE
TIME_UNIT     | IGNORE
YTYPE         | YTYPE
NAME          | IGNORE
DV            | DV
UNIT          | IGNORE
CENS          | CENS
MDV           | MDV
EVID          | EVID
AMT           | AMT
ADM           | ADM
ROUTE         | IGNORE
RATE          | RATE
TINF          | IGNORE
DOSE          | IGNORE
WT0           | IGNORE
SEX           | IGNORE
```

henning.schmidt@intiquan.com

MONOLIX and NONMEM Model Creation

- The function **IQMcreateNLMEproject** creates NONMEM or MONOLIX models
- Additionally to the previously defined information the „projectPath“ needs to be defined. This is the path to the folder in which to create the NLME model

```
%% Generate a MONOLIX model
projectPath                    = '../Models/MODEL_MONOLIX';
IQMcreateNLMEproject('MONOLIX',model,dosing,data,projectPath)
```

```
%% Generate a NONMEM model
projectPath                    = '../Models/MODEL_NONMEM';
IQMcreateNLMEproject('NONMEM',model,dosing,data,projectPath)
```

- **Execute the code above**

MONOLIX and NONMEM Model Creation

- Change into **Example Files/SimplePKmodel/Models** folder
- Have a look at the generated NONMEM and MONOLIX code

- Default settings for **IQMcreateNLMEproject**
  - NONMEM: SAEM
  - Lognormal distribution for individual parameters
  - Additive error model
  - Diagonal covariance matrix
  - Etc.

- **IQMcreateNLMEproject** has many optional argument, allowing to control most NLME features of interest

```
>> help IQMcreateNLMEproject
```

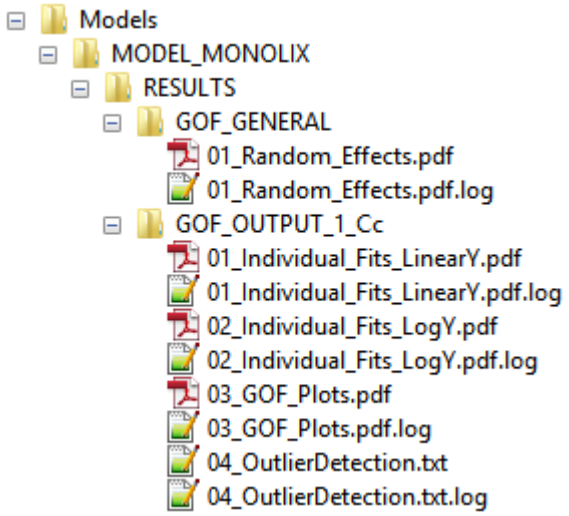
Running NONMEM and MONOLIX Models

- Change into **Example Files/SimplePKmodel/Scripts** folder
- An NLME model can be run by the following command

```
>> help IQMrunNLMEproject   % Get help

>> IQMrunNLMEproject('../Models/MODEL_MONOLIX')   % Run the MONOLIX model
>> IQMrunNLMEproject('../Models/MODEL_NONMEM')    % Run the NONMEM model
```

- Change into **Example Files/SimplePKmodel/Models** folder
- Have a look at the content in the RESULTS folders for both models
  - The subfolders in the RESULTS folders contain a wide range of goodness of fit assessments for the models – no need to postprocess the output of NONMEM and MONOLIX manually



Results on NONMEM and MONOLIX runs

- Have a look at the **project_results.txt** file in the **MODEL_NONMEM/RESULTS** folder
- Compared to the typical NONMEM log file this one is very readable and informative

```
================================================================================
    Summary results
    Project: MODEL_NONMEM
================================================================================

--------------------------------------------------------------------------------
Termination information (Method(s): SAEM)
--------------------------------------------------------------------------------
    STOCHASTIC PORTION WAS NOT TESTED FOR CONVERGENCE
    REDUCED STOCHASTIC PORTION WAS COMPLETED

--------------------------------------------------------------------------------
Name                              Value          stderr          RSE (%)
--------------------------------------------------------------------------------
log(ka)                          0.4746          0.3247           68.42
log(CL)                           1.242          0.4073           32.79
log(Vc)                           3.699          0.2386           6.449

ka                                1.607         0.5621*           34.97*
CL                                3.463           1.597*           46.13*
Vc                                40.41           10.08*           24.93*
                                    (*approximation by sampling)

omega(ka)                        0.6649          0.2282           34.32
omega(CL)                        0.2808          0.1652           58.81
omega(Vc)                        0.1522          0.1209           79.41

error_ADD1                       0.7301          0.0397           5.438
```

```
--------------------------------------------------------------------------------
Correlation of fixed effects and covariate coefficients
--------------------------------------------------------------------------------
ka                                1
CL                             0.24       1
Vc                            -0.33   -0.96       1

Eigenvalues (min, max, max/min): 0.04   2.11   55.20

--------------------------------------------------------------------------------
Correlation of random effects (variances) and error parameters
--------------------------------------------------------------------------------
omega2(ka)                        1
omega2(CL)                     0.38       1
omega2(Vc)                    -0.53   -0.71       1
error_ADD1                     0.31   -0.06    0.12       1

Eigenvalues (min, max, max/min): 0.24   2.09   8.71

--------------------------------------------------------------------------------
Objective function (SAEM)
The SAEM objective function should not be used for statistical testing.
Please consider the use of the IMPORTANCESAMPLING option!
--------------------------------------------------------------------------------
OFV:     -17.1658
AIC:     -3.16576
BIC:     29.2118
```

henning.schmidt@intiquan.com

IntiQuan

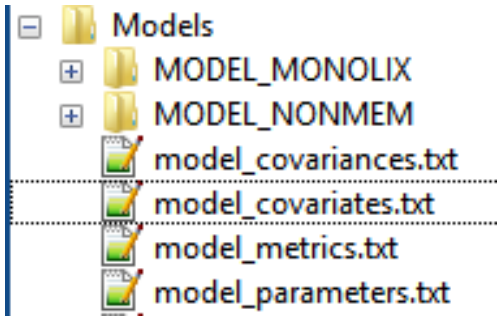
Running all NLME Models in a Folder

- If NLME models/projects are located within the same top folder (here: „**Models**“), they can be run in parallel by the following command:

```
>> help IQMrunNLMEprojectFolder   % Get help

>> IQMrunNLMEprojectFolder('../Models')   % Run all NLME models in folder
```

  - Parallel runs only possible if the parallel toolbox for MATLAB available, otherwise these models will be run sequentially
  - Parallel runs also require correct setup of **IQMpro/SETUP_PATHS_TOOLS_IQMPRO.m**
- Using **`IQMrunNLMEprojectFolder`** will generate additional information in the folder in which the NLME projects are located – have a look at **“../Models**” – it now contains four additional files (and their 4 log files)
  - **model_metrics.txt, model_parameters.txt, model_covariates.txt, model_covariances.txt**
  - Have a look at these files – self explaining and we will see more in the tutorials



Creation of Model Result Tables

- Execute the following code in the script

```
%% Create a table with model results for both models
IQMfitsummaryTable({'../Models/MODEL_MONOLIX' '../Models/MODEL_NONMEM'},'../Output/Model_Result_Table')
```

- This produces the following table, which with the reporting package of IQM Tools can be directly included into Word, LaTeX and HTML reports

	MODEL_MONOLIX			MODEL_NONMEM		
Parameter	Value	(RSE%)	(trans)	Value	(RSE%)	(trans)
CL	3.5	(8.53%)	(log)	3.46	(46.2%*)	(log)
Vc	40.1	(5.06%)	(log)	40.4	(25%*)	(log)
ka	1.58	(19.5%)	(log)	1.61	(35.1%*)	(log)
* Standard errors for NONMEM models approximated by sampling due to MU-Referencing						
omega(CL)	0.268	(24.5%)		0.281	(58.8%)	
omega(Vc)	0.151	(26.6%)		0.152	(79.4%)	
omega(ka)	0.638	(22.5%)		0.665	(34.3%)	
error_ADD1	0.734	(7.68%)		0.73	(5.44%)	
OBJ	346.29			-17.1658		
AIC	360.29			-3.1658		
BIC	363.69			29.2118		

See the „../Output“ folder for the generated table text file and the associated log file

henning.schmidt@intiquan.com

Defining more Options for NLME Model Generation

- The function **IQMcreateNLMEproject** has many optional settings, which might require quite some text for their definition. It is still far easier than to write MONOLIX or NONMEM code, but it takes time
- IQM Tools contains a function that generates template code with almost all features, allowing the user to do quick modifications and be done

```
>> help IQMcreateNLMEmodelGENcode
```

- Preparation for tutorial example
  - Change into **Example Files/SimplePKmodel/Scripts** folder
  - Open **SCRIPT_02_PKmodel_moreOptions.m** in the MATLAB editor
  - Update line number 12 with the path of your IQM Tools installation
  - Execute lines 4-16 in the script (highlight them and press „F9“)

henning.schmidt@intiquan.com

Defining more Options for NLME Model Generation

- Execute the following code in the **SCRIPT_02_PKmodel_moreOptions.m**

```
%% Generate code to generate an NLME model with most of the options
% available to IQMcreateNLMEproject
modelPath   = '../Models/modelPK.txt';
dosingPath  = '../Models/dosingPK.dos';
dataPath    = '../Data/dataNLME1.csv';
projectPath = 'MODEL_GEN';
IQMcreateNLMEmodelGENcode(modelPath,dosingPath,dataPath,projectPath)
```

  - This command generates template text in a temporary file and opens it in the MATLAB editor
  - Copy the generated code into the **SCRIPT_02_PKmodel_moreOptions.m**, starting from line 27

  - Read through the generated code – it is very well documented in should be almost self explaining

henning.schmidt@intiquan.com

Defining more Options for NLME Model Generation

- Locate the following lines in the generated code

```
% "covNames" needs to contain the names of the CONTINUOUS covariates of
% interest in the dataset. Please exchange the names that are available with the
% names of covariates in your dataset.
covNames = {'WT0' 'AGE0' 'BMI0'};

% "catNames" needs to contain the names of the CATEGORICAL covariates of
% interest in the dataset. Please exchange the names that are available with the
% names of covariates in your dataset.
catNames = {'SEX','OBESE'};
```

- Edit to match the following (we need to define the candidate covariates that are present in the dataset and which we might want to consider in the model)

```
% "covNames" needs to contain the names of the CONTINUOUS covariates of
% interest in the dataset. Please exchange the names that are available with the
% names of covariates in your dataset.
covNames = {'WT0'};

% "catNames" needs to contain the names of the CATEGORICAL covariates of
% interest in the dataset. Please exchange the names that are available with the
% names of covariates in your dataset.
catNames = {'SEX'};
```

henning.schmidt@intiquan.com

Defining more Options for NLME Model Generation

- Execute the first cell, defining the algorithm settings
- Execute the second cell, defining the candidate covariates
- In the third cell, please locate the following code

```
% Definition of covariance and covariate models
options.covarianceModel         = '';
options.covariateModel          = '';
```

and update to match the following

```
% Definition of covariance and covariate models
options.covarianceModel         = '';
options.covariateModel          = '{CL,WT0,SEX},{Vc,WT0}';
```

This will estimate covariate coefficients for WT0 on CL and Vc, and SEX on CL.

- In the third cell towards the bottom, please comment out either the NONMEM or the MONOLIX model generation

```
% For MONOLIX use:
IQMcreateNLMEproject('MONOLIX',model,dosing,dataFIT,projectPath,options)

% % For NONMEM use:
% IQMcreateNLMEproject('NONMEM',model,dosing,dataFIT,projectPath,options)
```

- Execute the third cell, generating the NLME model and running it

henning.schmidt@intiquan.com

Defining more Options for NLME Model Generation

- Have a look at the output for the generated model in **../Models/MODEL_GEN**
- If you want, repeat the exercise with NONMEM instead of MONOLIX or vice versa

- **Interesting:**
  - **The same NLME dataset is used for both tools**
  - **Switching tools is not more complicated than switching „NONMEM“ for „MONOLIX“ or the other way round**

A word about implementation of covariates

- Previous example showed the use of WT0 and SEX as covariates, but not how these covariates are mathematically implemented in the NONMEM or MONOLIX model
- IQM Tools handles covariates in the following way:
  - **MU-referencing** when exporting to NONMEM (In MONOLIX this is implicitly done)
  - Centers continuous covariates automatically by the median of the covariate in the dataset
  - Log-transforms the centered continuous
  - Numerically smallest category of a categorical covariate is used as references value
  - Additive aggregation of covariate terms and MU in the transformed parameter domain (according to the chosen distribution of the individual parameters)
  - Example continuous covariate (with assumed median of 70):

```
log(CL) = log(TCL) + beta*log(WT0/70)
=> CL      = TCL*(WT0/70)^beta
```

  - Example categorical covariate (SEX=1 male, =2 female)

```
log(CL_male)   = log(TCL)
log(CL_female) = log(TCL) + beta
```

henning.schmidt@intiquan.com

A word about implementation of covariates

- This means that the implementation of the covariates is slightly different from what Pharmacometricians are used to
- However, the following needs to be considered:
  - For the typical applications in the Pharma industry such an approach is enough to identify covariates of clinical relevance and thus viable and acceptable
  - For research applications where the goal is to assess many different possible covariate models and compare them to each other, IQM Tools still allows to code the mathematical expressions of the covariate models into the IQMmodels code.
    - In this case, the following needs to be considered:
      - Covariates need to be passed to the model as regression parameters
      - The SAEM algorithm should not be used. Please used gradient based methods from NONMEM in this case
- Any covariate expression is possible in IQM Tools, if defining models manually

```
CL = TCL*(AGE0-40)^beta1*(1+beta2*(SEX-1))
```

- For automatic tools, such as SCM, the standard approach is used

Robustness Analysis of NLME Parameter Estimations

- In order to assess the robutsness of parameter estimates it is possible to consider the standard errors of the estimated parameters
- However, this often does not give the full picture and often is overly optimistic

- IQM Tools allows to easily run the same model N times from randomized initial guesses
- This allows to assess the robustness of the parameter estimates and thus the robustness of the model in detail
  - Not robust models often are identified by a large difference in final objective function values and in correlations of parameter values with each other

- In the following we will perform such a robustness analysis

henning.schmidt@intiquan.com

Robustness Analysis of NLME Parameter Estimations

- Locate the following lines in the code

```
% Definition of covariance and covariate models
options.covarianceModel          = '';
options.covariateModel           = '{CL,WT0,SEX},{Vc,WT0}';
```

- Exchange against (since WT0 and SEX were not significant on CL)

```
% Definition of covariance and covariate models
options.covarianceModel          = '';
options.covariateModel           = '{Vc,WT0}';
```

- Locate the following lines in the code

```
% Robustness analysis. If Ntests>1 then Ntests models will be generated in the project folder,
% each starting from randomly selected initial guesses (based on std_noise_setting).
options.Ntests                   = 1;
options.std_noise_setting        = 0;
```

- Setup the robustness analysis by exchanging with

```
% Robustness analysis. If Ntests>1 then Ntests models will be generated in the project folder,
% each starting from randomly selected initial guesses (based on std_noise_setting).
options.Ntests                   = 10;
options.std_noise_setting        = 0.5;
```

- Execute the 3rd cell to generate the 10 models

Robustness Analysis of NLME Parameter Estimations

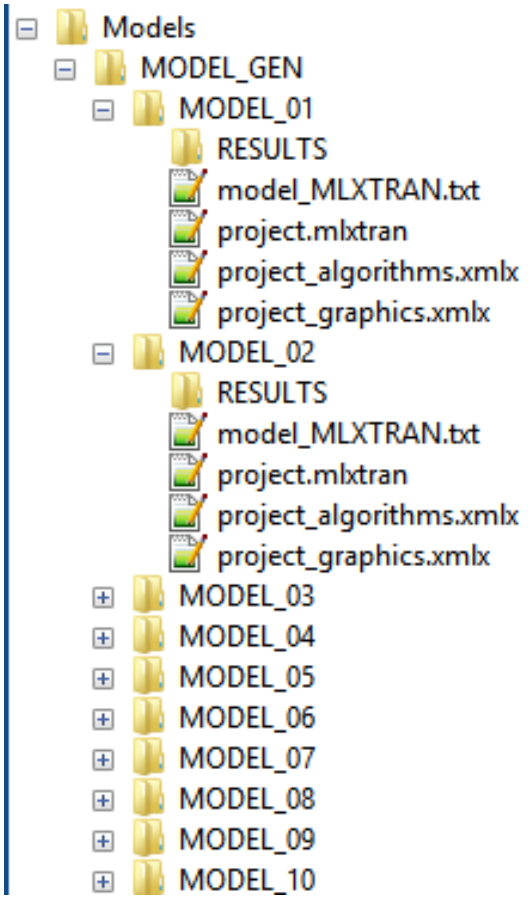
- Have a look into Models/MODEL_GEN
  - 10 Models have been generated
  - Only difference: randomly sampled initial guesses for the fixed effect parameters that are estimated
- Execution of the third cell lead to an error when attempting to run the models with the command

```
% Run the model:
IQMrunNLMEproject(projectPath)
```

- Here there are several models generated in „projectPath“ and thus the above command needs to be exchanged to

```
% Run the model:
IQMrunNLMEprojectFolder(projectPath)
```

- Re-run the third cell to generate and run the 10 models



henning.schmidt@intiquan.com

IntiQuan

Robustness Analysis of NLME Parameter Estimations

- Please have a look at the folder Models/MODEL_GEN and the contained text files
- These give a good overview of the robustness of this model

MODEL	round(BIC)	CL	Vc	ka	error_ADD1	error_PROP1
MODEL_06	358	3.5	40.9	1.54	0.415	0.059
MODEL_07	358	3.51	40.6	1.52	0.423	0.0578
MODEL_01	358	3.5	40.6	1.53	0.432	0.0559
MODEL_03	358	3.5	40.6	1.53	0.429	0.0573
MODEL_05	358	3.49	40.7	1.54	0.445	0.0536
MODEL_08	358	3.5	40.7	1.53	0.444	0.0541
MODEL_10	358	3.5	40.8	1.55	0.441	0.0539
MODEL_04	359	3.51	40.6	1.52	0.434	0.057
MODEL_02	359	3.51	40.5	1.53	0.45	0.0532
MODEL_09	359	3.51	40.6	1.53	0.428	0.0584

- Note: the purpose of this robustness functionality is not to try to find some initial guesses that might make a numerically unstable NLME Tool of choice to converge to a solution, as PsN Execute does. The goal of this function is to generate valuable information about a models robustness to allow the modeler to take decisions with respect to which parameters (fixed and/or random) to estimate and which not.

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- **General representation of clinical data**
  - Generalized data format for clinical data used in IQM Tools
  - Checking of data consistency
  - Conversion to task specific dataset format
  - Graphical exploration
  - Conversion to dataset for nonlinear mixed effect parameter estimation
- Examples

henning.schmidt@intiquan.com

General standard dataset

- Capturing clinically relevant information
- Reuse of the data for many purposes (across line functions – DMPK, PMX, STATS)
- Collection of data in databases
- Efficient data preparation
- Efficient data checking => higher quality data
- Efficient standard analyses
- Easy integration in workflow approaches

General standard dataset – is it feasible?

Datasets for pharmacometric analyses: internal review and standardization efforts

Andrijana Radivojevic[1)], Henning Schmidt[2)]

Pharmacometrics Modeling & Simulation, [1)]Novartis Pharmaceuticals Corporation, East Hanover, USA, [2)]Novartis Pharma AG, Basel, Switzerland

NOVARTIS

Background & Objective

Analysis datasets for the conduct of pharmacometric (PMX) analyses are usually prepared by the modeler him/herself or requested from a supporting programming group. The content and the structure of such datasets are defined in a dataset specification. There are currently no standards in this regards, and therefore the dataset structure typically depends on the modeling activity, the modeling tool, the models to be assessed, and even on the modeler. This kind of *freedom* is directly translated into lower efficiency in dataset preparation process, multiple iterations between modelers and programmers, and compromised quality of the of delivered analysis dataset.

Standards, however, could considerably improve the process of dataset specification, preparation, and allow for automated dataset quality checks. The goal of this work is to analyze the typical content of PMX datasets, reasons for their variability, and to assess if and how standardization in this context might be feasible.

Methods

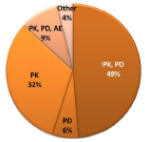
Extensive review of historical and existing data requests within the Novartis PMX Modeling & Simulation (M&S) department was performed. This included 384 different data requests, spanning the period 1999–2015. Those data requests were directly linked with corresponding PMX tasks related to support of drug development programs across various Novartis business units and disease areas, e.g. immunology, dermatology, primary care, critical care, neuroscience, ophthalmology, biosimilars and oncology.

The exercise provided solid statistics on dataset structure and data content of analysis datasets for PMX deliverables. Inter-modeler variability considering styles, preferences, analysis platforms, modeling methods, disease area specific requirements and more, was captured as well.

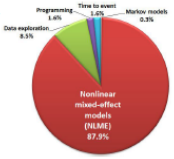
Results

Clinical Questions, Data Content and Analysis Types

- Characterization of dose-exposure-response relationships has been, and still is, the main *leitmotif* in all PMX activities. It is then not surprising that the majority of requested datasets are tailored to enable those kinds of analyses and answer related clinical questions.



- Consequently, pharmacokinetic (PK), pharmacodynamic (PD) and adverse event (AD) information are captured in datasets, with the main focus on describing and/or predicting drug efficacy and safety.
- Nonlinear mixed-effects (NLME) modeling remains the predominant methodology used to analyze this data, with only rare occurrence of alternative approaches.
- Given that data exploration is, and should be, a subset task of any modeling activity, there is no need to differentiate it in a separate analysis category.



Heterogeneity in data specification

- Despite the aim of answering similar questions, capturing similar information and analyzing data using similar methodology, the content and structure of analysis datasets still largely depend on the modeling activity, the modeling tool, the model to be assessed, and the modeler.
- Our survey gives some insight on major sources of variability:
  - Non-unified nomenclature: same variables are named differently, e.g., REGIMEN, ARM, TRT, COHORT.
  - Non-mnemonic nomenclature: variable names are not possible to be associated with the entity they represent e.g., FLAG1, FLAG2, ..., FLAG30.
  - Different programming abilities among modelers: basic data manipulations are part of data specifications, thus increasing the dataset preparation time.
  - Extent of requested information: directly influencing the size of analysis dataset, e.g., from < 1MB – 800MB (when including time points for simulation).
  - Modeling platform: NONMEM vs. MONOLIX, R vs. MATLAB.
- All these differences significantly compromise the efficiency and quality of PMX deliverables and directly impact:
  - Resources for data preparation: each data request is a new data request, with different structure and different content.
  - Project takeover: datasets almost never contain names of observations or units, but only compartment numbers which are linked with their meaning in other documents – making it more complicated for a naïve modeler to onboard.
  - Fluent usage of various software tools: datasets are adapted to one software tool (mostly NONMEM) only.
  - Reusability and Reproducibility: datasets are adopted to the individual code of one modeler performing the analysis. If another modeler has his/her own set of scripts, then another matching dataset needs to be created, containing the same information but in a different structure.

Discussion

- As presented in [1], the concept of a generalized dataset and its automatic transformation into an analysis specific dataset offers a robust solution that overcomes the aforementioned issues.

- The notion of generalized dataset resembles the idea of a master dataset for clinical projects. It consists of a broad collection of clinical information in a well-defined structure that is independent of the project (typically standard within a therapeutic area), of the modeling activity to be performed, and of the modeling tool to be used.
- The standardized structure allows for more efficient and less error-prone preparation of datasets, automatic data consistency checks, pooling across studies, easier understanding and re-use of the data, automatic generation of common graphical exploration plots, automatic conversion to modeling activity and tool dependent modeling datasets.
- This approach directly contributes to improved efficiency and quality of PMX deliverables, and a proof-of-concept was demonstrated by implementation in [2].

- Recognizing the advantages of data standardization, Novartis PMX M&S is on its way towards the successful organization-wide implementation of the concepts discussed here:
  - The existing generalized dataset described in [1] was updated to cover ≥ 95% of the identified needs in a diverse PMX group.
  - Proof-of-principle has been demonstrated in multiple clinical projects.
  - It is a collaborative effort between modelers and programmers, who work in parallel on improving related processes and background infrastructure, and developing tools.

Conclusion

Underestimation of the time needed for preparation of high quality modeling datasets is a common oversight, and lack of data standards and supporting tools can have direct implications on reproducibility and auditing of modeling work [3]. Therefore, a standardized dataset format within PMX organizations is a pivotal step toward efficient implementation of model-based drug development in a corporate setting. All this should be taken within a broader context of streamlined workflows and industrialization efforts. Global alignment on this topic would further advance the field of Pharmacometrics.

References:
[1] Schmidt, Radivojevic (2014) Enhancing population pharmacokinetic modeling efficiency and quality using an integrated workflow. J Pharmacokinet Pharmacodynam. DOI 10.1007/s10928-014-9370-4
[2] Schmidt (2013) SBPOP Package: Efficient support for model based drug development – from mechanistic models to complex trial simulation. PAGE www.sbtoolbox2.org
[3] Jönsson, Jonsson (2007) Timing and efficiency in population pharmacokinetic/pharmacodynamic data analysis projects. Pharmacometrics: The Science of Quantitative Pharmacology. Edited by E. Ette and P. J. Williams, John Wiley & Sons, Inc.

This study was supported by Novartis Pharma AG, Basel, Switzerland.

Poster presented at ACoP, October 4-7, 2015, Crystal City, VA, USA

Acknowledgments:
Didier Renard and Mick Looby for reviewing this poster.

Corresponding author: andrijana.radivojevic@novartis.com

It is not only feasible – it is absolutely necessary!

henning.schmidt@intiquan.com

IntiQuan

Documentation of General Standard Dataset used in IQM Tools

Column Name	Type	Description
IXGDF	numeric	Column containing numeric indices for each record (1,2,3,4,5, ...) to allow for matching records in case of post-processing the general dataset format
IGNORE	string	Reason/comment related to exclusion of the sample/observation from the analysis
USUBJID	string	Unique subject identifier
COMPOUND	string	Name of the investigational compound
STUDY	string	Study number
STUDYDES	string	Study description
PART	string	Part of study as defined per protocol (1 if only one part)
EXTENS	numeric	Extension of the core study (0 if not extension, 1 if extension)
CENTER	numeric	Center number
SUBJECT	string	Subject identifier (within a center - typically not unique across whole dataset)
INDNAME	string	Indication name
TRTNAME	string	Name of actual treatment given
TRTNAMER	string	Name of treatment to which individual was randomized
VISIT	numeric	Visit number
VISNAME	string	Visit name
BASE	numeric	Flag indicating assessments at baseline (=0 for non-baseline, =1 for first baseline, =2 for second baseline, etc.)
SCREEN	numeric	Flag indicating assessments at screening (=0 for non-screening, =1 for first screening, =2 for second screening, etc.)

henning.schmidt@intiquan.com

Documentation of General Standard Dataset used in IQM Tools

Column Name	Type	Description
DATEDAY	String	Start date of event ('01-JUL-2015')
DATETIME	string	Start time of event ('09:34')
DURATION	numeric	Duration of event in same time units as TIMEUNIT
NT	numeric	Planned time of event. Based on protocol, in the time unit defined in TIMEUNIT column
TIME	numeric	Actual time of event relative to first administration, in the time unit defined in TIMEUNIT column
TIMEUNIT	string	Unit of all numerical time definitions in the dataset ('hours','days','weeks','minutes')
TYPENAME	string	Unique type of event
NAME	string	Unique name for the event
VALUE	numeric	Value of the event, defined by NAME. E.g., the given dose, the observed PK concentration, or the value of other readouts. The values need to be in the units, defined in the UNIT column. Specific cases: - For concomitant medications the dose will be given - For BLOQ values, 0 will be used - Severity levels for adverse events - Should not be populated if VALUE_TEXT is populated
VALUETXT	string	Text version of value (if available and useful). Character value as given in the CRF. Should not be populated if VALUE is populated

henning.schmidt@intiquan.com

Documentation of General Standard Dataset used in IQM Tools

Column Name	Type	Description
UNIT	string	Unit of the value reported in the VALUE column. For same event the same unit has to be used across the dataset.
ULOQ	numeric	Upper limit of quantification of event defined by NAME
LLOQ	numeric	Lower limit of quantification of event defined by NAME
ROUTE	string	Route of administration (iv, subcut, intramuscular, intraarticular, oral, inhaled, topical, rectal)
INTERVAL	numeric	Interval of dosing, if single row should define multiple dosings
NRDOSES	numeric	Number of doses given with the specified interval, if single row should define multiple dosings
COMMENT	string	Additional information for the observation/event

The general dataset format in IQM Tools covers >95% of the typical pharmacometric analyses. It is certainly not complete and perfect, but it is a start in the right direction. An industry wide agreement would be of tremendous interest.

henning.schmidt@intiquan.com

Documentation of General Standard Dataset used in IQM Tools

- Optional columns in the IQM Tools General Standard Dataset format
  - The general data format might also contain the following columns, which are numeric equivalents to some string columns. If not provided, then these are generated automatically in the post processing steps of the general dataset in IQM Tools

Column Name	Type	Description
IND	numeric	Numeric indication flag (unique for each entry in INDNAME)
STUDYN	numeric	Numeric study flag (unique for each entry in STUDY)
TRT	numeric	Numeric actual treatment flag (unique for each entry in TRTNAME)
TRTR	numeric	Numeric randomized treatment flag (unique for each entry in TRTNAMER)

henning.schmidt@intiquan.com

Example for a General Standard Dataset

- Please change into the **Example Files/SimpleSysPharmExample/Data** folder
- Open the **data_general.csv** file
- Have a look

IGNORE	USUBJID	COMPOUI	STUDY	STUDY_DE	PART	EXTENSIO	CENTER	SUBJECT	INDICATIC	TRT	TRT_NAM	TRT_NAM	VISIT	VISIT_NAI	BASE	SCREEN	DATE_DA	DATE_TIM	DURATIO	NOMINAL	TIME	TIME_UNI	TYPE_NAI	NAME	VALU
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	1	0	Unknown	Unknown	0	0	0	Minutes	Body	BMI	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	1	0	Unknown	Unknown	0	0	0	Minutes	Body	Weight	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	1	0	Unknown	Unknown	0	0	0	Minutes	Demograp	Age	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	1	0	Unknown	Unknown	0	0	0	Minutes	Demograp	Gender	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	0	0	Unknown	Unknown	0	0	0	Minutes	Doses	Glucose d	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	1	0	Unknown	Unknown	0	0	0	Minutes	PD	Insulin	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	1	0	Unknown	Unknown	0	0	0	Minutes	PK	Glucose	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	0	0	Unknown	Unknown	0	15	15	Minutes	PD	Insulin	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	0	0	Unknown	Unknown	0	15	15	Minutes	PK	Glucose	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	0	0	Unknown	Unknown	0	30	30	Minutes	PD	Insulin	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	0	0	Unknown	Unknown	0	30	30	Minutes	PK	Glucose	
	Subject1	Glucose	OGTT123	Oral Gluc	1	0	1	2	Insulin re	75	75g Gluc	75g Gluc	NaN	Unknown	0	0	Unknown	Unknown	0	45	45	Minutes	PD	Insulin	

henning.schmidt@intiquan.com

IntiQuan

General standard dataset Processing and Exploration

- Please change into the **Example Files/SimpleSysPharmExample/Scripts** folder
- Open the **SCRIPT_01_prepare_explore_data.m** file in the MATLAB editor
  - Update line 14 with the path to your installation of IQM Tools

- It will walk you through the following steps:
  - Checking of data consistency
  - Conversion to task specific dataset format
  - Graphical exploration
  - Conversion to dataset for nonlinear mixed effect parameter estimation

- IQM Tools contains a large number of analysis function. In this example a wrapper function has been called that performs a large range of analyses. The user can do own analyses as well or use smaller sub functions in IQM Tools.
- Functions in IQM Tools are documented here:

```
>> doc IQMlite
>> doc IQMpro
```

henning.schmidt@intiquan.com

Tutorial Outline

- General introduction to the IQM Tools Suite
- Compliance mode
- Models: dosing, output and parameter definitions
- Dosing description and simulation
- Import and export of SBML models
- Important things to consider for systems pharmacology models
- Interface to NLME Tools
- General representation of clinical data
- **Examples**

henning.schmidt@intiquan.com

Simple Glucose-Insulin Interaction Example

- The data that has been explored in the previous section is the basis for NLME parameter estimation
- Please change into the folder **Example Files/SimpleSysPharmExample/Scripts**
- Open the file **SCRIPT_02_model.m**
  - Update line 14 with the path to your installation of IQM Tools

- Please work through the example script by executing the different parts and having a look at the output of these parts in the folder „Models“

- In a classroom setting we will go through all of this in more detail

Simple Glucose-Insulin Interaction Example
Simplistic Model

- Ensuring that the model is in steady-state for no external input

```
********** MODEL STATES

% Glucose absorption - first order
% Glucose input in gramm => convert to mmol/l
% Gd then in mmol/l
d/dt(Gd) = -kaG*Gd + 1000/VG/180.182*INPUT1

% Glucose
% G in mmol/l
% - At baseline values of Glucose and Insulin no change is happening (without absorption from extern)
% - Insulin above baseline is reducing glucose
d/dt(G)  = +kaG*Gd - p1*(G-Gb) - p2*(I-Ib) - kG1*G + kG2*G2
d/dt(G2) =                                   kG1*G - kG2*G2

% Insulin
% I in uU/ml
% - At baseline values of Glucose and Insulin no change is happening
% - Glucose above baseline is stimulating Insulin production
d/dt(I)  = -p3*(I-Ib) + p4*(G-Gb)

% Initial conditions
Gd(0)    = 0
G(0)     = Gb
G2(0)    = kG1/kG2*Gb
I(0)     = Ib
```

- This model will not be usable for NONMEM, since „G“ is used as a state name

Simple Glucose-Insulin Interaction Example
Simplistic Model

- Units and input/outputs are documented

```
********** MODEL NOTES
Very simple Glucose/Insulin interaction model ...

Units:
- Time in minutes
- Glucose state (G) in mmol/l
- Insulin state (I) in uU/ml
- INPUT1: glucose in gramm => requires conversion
- OUTPUT1: glucose in mmol/l
- OUTPUT2: glucose in uU/ml
```

- Parameters are documents

```
% Baseline parameters for glucose and insulin
Gb       = 5            % (1/min)                <estimate>
Ib       = 7            % (1/min)                <estimate>

% Glucose volume parameter for scaling the input from amount to concentration
VG       = 10           % (l)                    <estimate>

% Glucose absorption parameter
kaG       = 0.05         % (1/min)                <estimate>

% Glucose distribution parameter - second compartment
kG1      = 0.2          % (1/min)                <estimate>
kG2      = 0.2          % (1/min)                <estimate>

% Glucose-Insulin interaction parameter
p1       = 0.04         % (1/min)                <estimate>
p2       = 0.02         % (mmol/l/(uU/ml)/min)   <estimate>
p3       = 0.06         % (1/min)                <estimate>
p4       = 0.25         % (uU/ml/(mmol/l)/min)   <estimate>
```

henning.schmidt@intiquan.com

Simple Glucose-Insulin Interaction Example
Simplistic Model

- Outputs of the model are defined

```
********** MODEL VARIABLES

% Definition of outputs for linking to NLME parameter estimation tools
OUTPUT1   = G
OUTPUT2   = I
```

  - OUTPUT1 will be linked to the readout in the NLME dataset with YTYPE = 1
  - OUTPUT2 will be linked to the readout in the NLME dataset with YTYPE = 2

- If your model has more outputs that have matches with the observed data ... There is no limit ...

Simple Glucose-Insulin Interaction Example
Simplistic Model

- The model contains a single input definition (INPUT1), which defines a glucose bolus, taken orally. The unit of the administered glucose is gramm
- The dataset contains glucose doses (ADM=1, linking to INPUT1)

- Drug action could be added into the model by adding a PK model part and concentration dependent modulation of model properties of interest
  - This PK model part would get an INPUT2
  - For parameter estimation with drug action, the dataset would also need to contain drug dosing information with ADM=2, linking to INPUT2

- **For now, the model is simplistic, but nothing prevents you to make the model as complex as you wish – as long you implement drug and other external inputs in the right way**

- Currently it is an extreme lack of time that leads to this simple and not a more elaborate example ...

henning.schmidt@intiquan.com

Selection of parameters to be estimated

- Assuming you have 100s of parameters in your model and you want to know which ones to estimate ... Good luck ;-)

- Still, something can be done
  1. If you have a larger model with many parameters you most likely have already parameterized it as good as you were able to (literature data, manual tuning to reference profiles, ...)
  2. The model most likely already describes semi-quantitatively the behavior of interest and you just want to get more out of the model (e.g. covariates, drug action, etc.)
  3. In this case you could think about using local sensitivity analysis
     - Perform a parametric sensitivity analysis around a trajectory or input/dosing setting of interest and check how much impact the different parameters have on your readouts of interest (which are also the ones you expect to have in the dataset for parameter estimation)
     - Look at correlations of sensitivity trajectories, etc., etc.
     - Optimal design methods (just the evaluation step), calculating the Fisher Information Matrix are of interest as well ... And still on the todo list for implementation into IQM Tools

Then select a first set of parameters for estimation and give it a try ...

Tutorial Goal: „You should now be able to“

- Understand and use the compliance mode – or switch it off
- Define IQMmodels with dosing inputs and simulate dosing scenarios
- Import models from SBML to IQMmodels, add inputs and outputs for simulation and NLME approaches
- Understand the requirements on model definition for use of NONMEM or MONOLIX as parameter estimation tools
- Convert models to NONMEM and MONOLIX and estimate parameters
- Understand and handle the general dataset specification

THE END

Thank you for your participation and interest!

henning.schmidt@intiquan.com