Open Systems Pharmacology
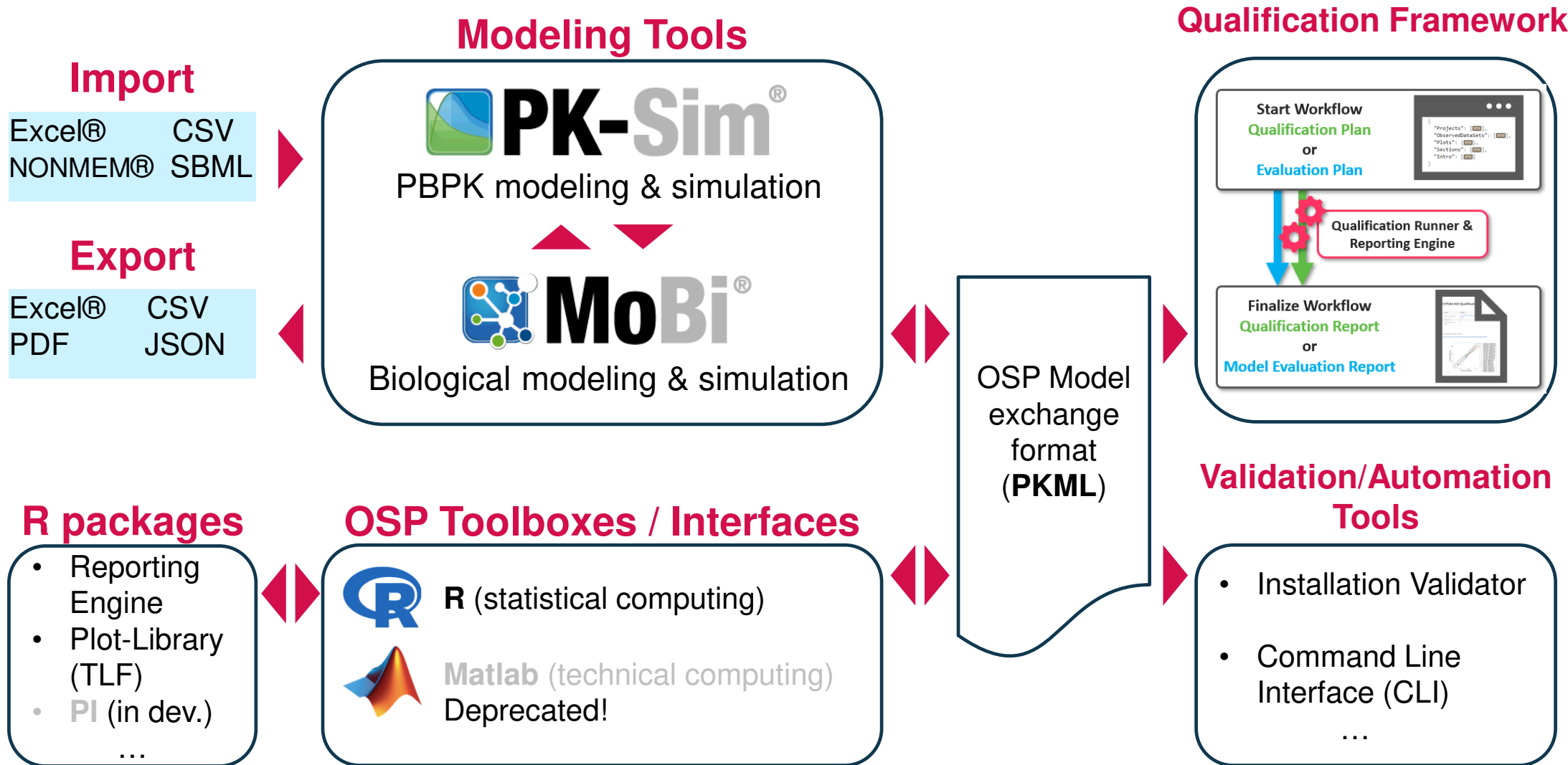
MoBi and R-Toolbox

Juri Solodenko
April 2021

OPEN SYSTEMS
PHARMACOLOGY

- ➢ OSP software landscape

- ➢ PK-Sim vs. MoBi: when to use

- ➢ MoBi: Overview

- ➢ MoBi: modeling concepts

    - ➢ Building blocks

    - ➢ „Generic" modeling
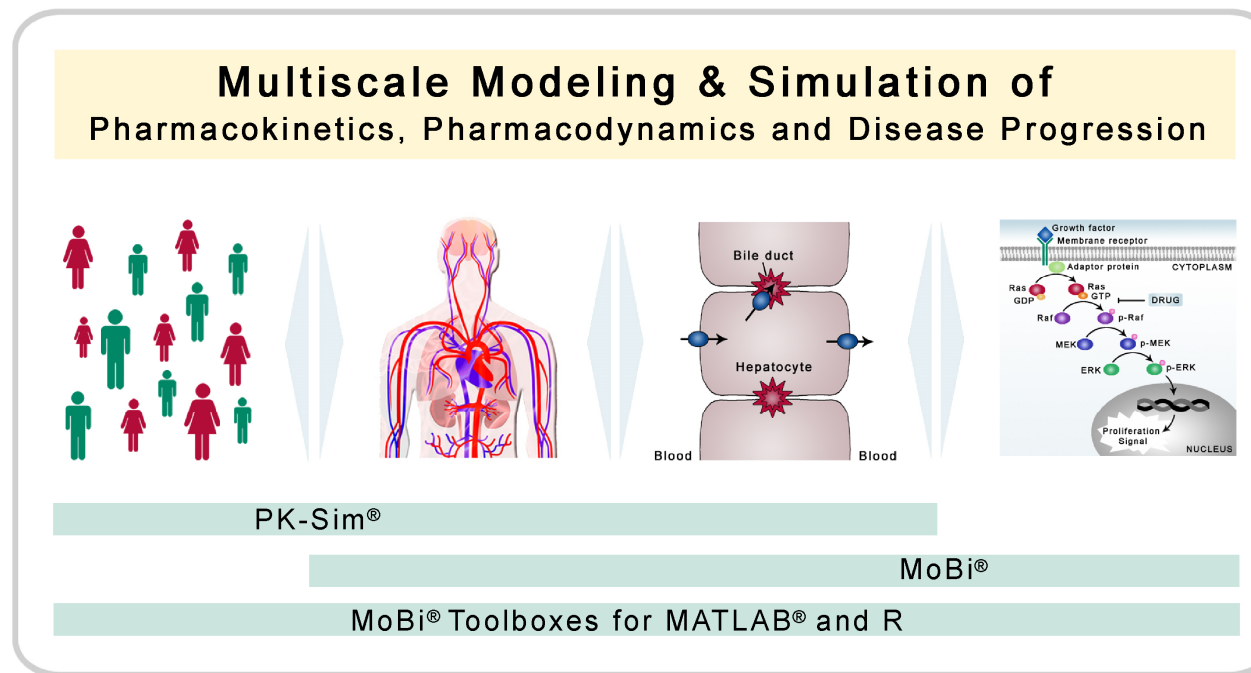
- ➢ R-Toolbox

OSP software landscape

Modeling Tools

Import

Excel®	CSV
NONMEM®	SBML

Export

Excel®	CSV
PDF	JSON

PK-Sim®
PBPK modeling & simulation

MoBi®
Biological modeling & simulation

OSP Model exchange format (**PKML**)

Qualification Framework

Start Workflow
Qualification Plan
or
Evaluation Plan

Qualification Runner & Reporting Engine

Finalize Workflow
Qualification Report
or
Model Evaluation Report

R packages

- Reporting Engine
- Plot-Library (TLF)
- PI (in dev.)
  …

OSP Toolboxes / Interfaces

R (statistical computing)

Matlab (technical computing)
Deprecated!

Validation/Automation Tools

- Installation Validator

- Command Line Interface (CLI)
  …

OSP Modeling Tools: PK-Sim and MoBi

- PK-Sim® is a comprehensive software tool for **whole-body physiologically based pharmacokinetic** (**PBPK**) **modeling**.

- Like most PBPK modeling tools, **PK-Sim® is designed for use by non-modeling experts and only allows for minor structural model modifications**.

- It enables rapid access to all relevant anatomical and physiological parameters for humans and common laboratory animals contained in the integrated database.

- Users can access different PBPK calculation methods, which enables fast and efficient model building and parameterization.

- PK-Sim® offers different model structures to choose from, e.g., to account for important differences between small and large molecules.

- More importantly, PK-Sim® is fully compatible with the expert modeling software tool MoBi®, thereby allowing full access to all model details including the option for extensive model modifications and extensions.

OSP Modeling Tools: PK-Sim and MoBi

- MoBi® is a systems biology software tool for multiscale physiological modeling and simulation.

  - Within the restrictions of ordinary differential equations, almost any kind of (biological) model can be imported or set up from scratch.

  - Examples include biochemical reaction networks, compartmental disease progression models, or PBPK/PD models.

  - However, de novo development of a PBPK model, for example, is very cumbersome such that the preferred procedure is to import them from PK-Sim®.

Multiscale Modeling & Simulation of
Pharmacokinetics, Pharmacodynamics and Disease Progression

PK-Sim®

MoBi®

MoBi® Toolboxes for MATLAB® and R

PK-Sim vs. MoBi: main features

	PK-Sim	MoBi
Integrated databases with: - anatomical and physiological properties of humans and laboratory animals - protein expressions - whole-body model structures for small molecules and biologics - prediction models for tissue partition coeff. / cellular and intestinal permeabilities[*]	X	[*]X
Template library for: - optional processes (DDI / metabolizing pathways / active transporter / protein binding) - food effects / administration protocols / formulations	X	
Creating of individuals and populations	X	
Building Block (BB) Concept: Simulation creation by simple combining of previously defined BBs	X	X
Visualization: plotting of all calculated time courses	X	X
Visualization: population plots (Box-Whisker, Range, Scatter, ..)	X	
Fully integrated Parameter Identification (PI) Toolbox	X	X
Sensitivity Analysis (SA) of PK-Parameters (AUC, Cmax, …) vs. simulation parameters.	X	X
Built-in working journal for manual annotation of models and simulations	X	X
„No limits" modeling: modify formulas, change structure beyond templates, …		X

When will we use MoBi® ?

➤ Changes in parameters that are not accessible in PK-Sim® .

➤ Structural physiological changes (e.g., adding an organ).

➤ PK-PD modeling (e.g., tumor growth and growth inhibition).

➤ Modeling of metabolic or signaling networks (non-PK models), in vivo and in vitro scenarios.

➤ … (and many more)

MoBi: Building Block Concept

- MoBi® uses **building blocks (BB)** that are grouped into:

  - Spatial Structures

  - Molecules

  - Reactions

  - Passive Transports

  - Observers

  - Events

  - Molecule Start Values

  - Parameter Start Values

  - Observed Data

- Building blocks out of the above-mentioned groups can be combined to generate models.

- The advantage of building blocks is that they can be reused.

- For example, a different set of parameter start values may define a new scenario, situation, or individual.

Select one BB of each kind
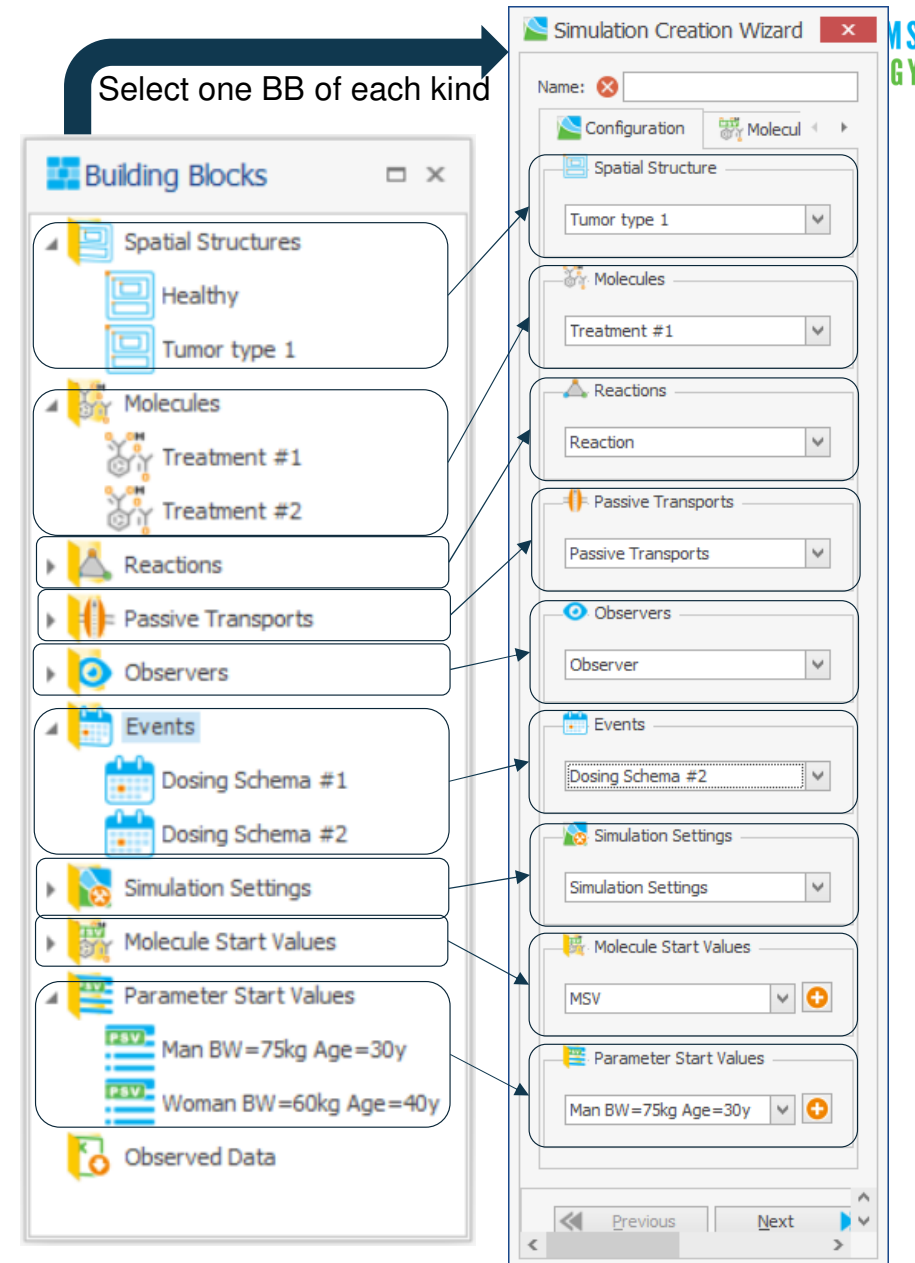
Export from PK-Sim® to MoBi®: Where to find what

OSP OPEN SYSTEMS PHARMACOLOGY

Building blocks in PK-Sim®

- Individuals
- Populations
- Compounds
- Formulations
- Administration Protocols
- Events
- Observed Data
- Observers

Building blocks in MoBi®

- Spatial structure
- Molecules
- Reactions
- Passive Transports
- Observers
- Events
- Simulation Settings
- Molecule Start Values
- Parameter Start Values
- Observed Data

MoBi: Workflow

- Import simulation(s) and observed data from PK-Sim **AND**/**OR**
- Import simulation(s) from SBML **OR**
- Start from scratch

→

- Modify imported building blocks
- Create new building blocks
- Import observed data

→

- Create simulations from building blocks

↓

- Run simulations
- Plot simulation results
- Compare simulation results to observed data
- Perform sensitivity analysis

←

- Change simulation parameters (manually or via parameter identification)
- Exchange building blocks

←

- Export results (Excel, CSV, …)
- Export simulations for processing in PK-Sim or other OSP tools (pkml)

MoBi: Quick Tour

MoBi: Modeling concepts

Spatial Structure: define and connect model compartments

- A spatial structure can be an organism consisting of organs, cells and other substructures. Alternatively, it can be a laboratory setup, like a test tube.

- Containers of a spatial structure can be defined as **Physical**, which may contain molecules in the simulation. They can also be defined as Logical, meaning they do not represent a real container with molecules but instead just representing a grouping of sub-containers.

- Any container may have parameters associated with it. They can describe physical or biological properties of the container that are required for processes like transports or reactions (e.g., *Volume*, *pH*, …)

- Containers are connected via **Neighborhoods**

- Containers and neighborhoods may be labelled with **tags**. These tags, together with the name given to a container or neighborhood, may be used for selectively enabling observers, active or passive transports, or events.
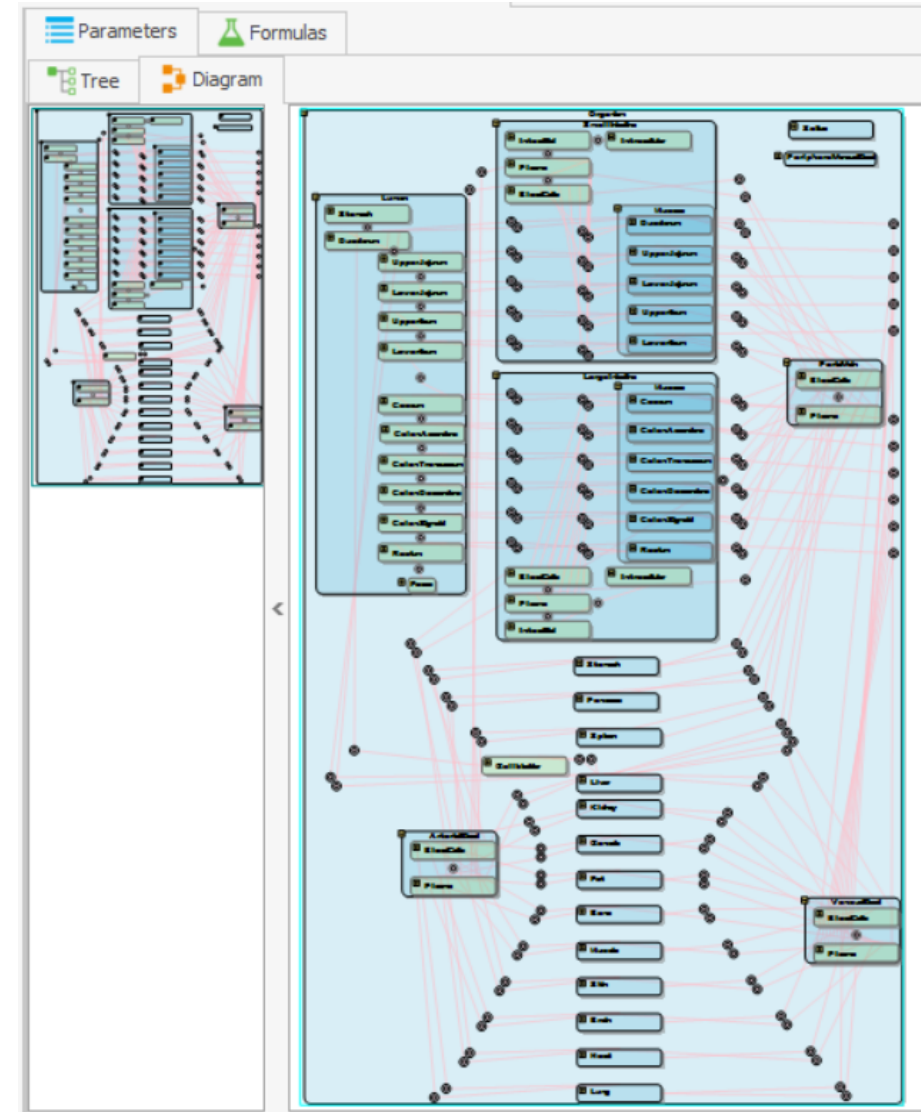
Molecules:

- Mostly, molecules will be chemical or biological compounds and can either be quantified by amount or by concentration.

- It is important to specify whether a molecule may move freely through all containers of a model, which is the default setting, or if it is stationary, i.e., it is immobilized in the current container. In the latter case, for example, the molecule may represent a membrane-bound receptor protein.

Molecule Start Values

- This building block defines the initial amounts or concentration of all molecules in the model.

- If you want to exclude a specific molecule from a specific container, de-select the "is present" checkbox.

Parameter Start Values

- This building block defines the start values of the parameters in the model.

- This allows for the setting of specific physiological parameters if different individuals, species or patient subgroups are to be considered.

„Generic" modeling: Formulas in building blocks

- Example 1: compartment volume of the intracellular space in a tissue organ is defined by the product of the **organ volume** and the **fraction of intracellular space** of the organ

- This could be defined by a separate formula in each tissue organ

- Disadvantages

  - N formulas need to be defined (one per tissue organ)

  - If a modification of intracellular volume definition is required: N formulas need to be adjusted

Brain

PLS   BC

ENDO

INT
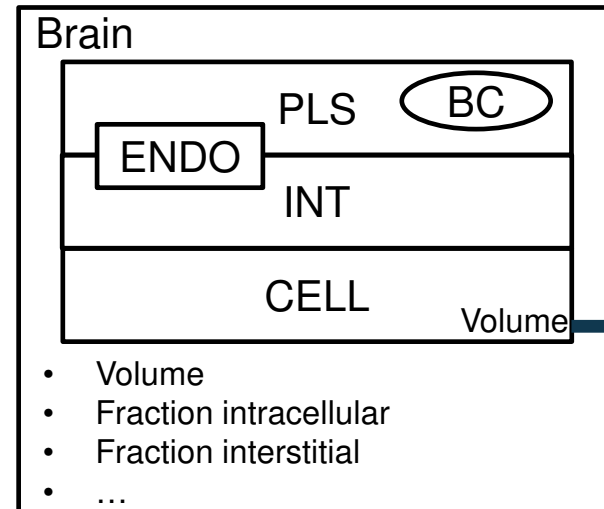
CELL   Volume → Volume_**Brain** * Fraction_intracellular_**Brain**

- Volume
- Fraction intracellular
- Fraction interstitial
- ...

Bone

PLS   BC

ENDO

INT
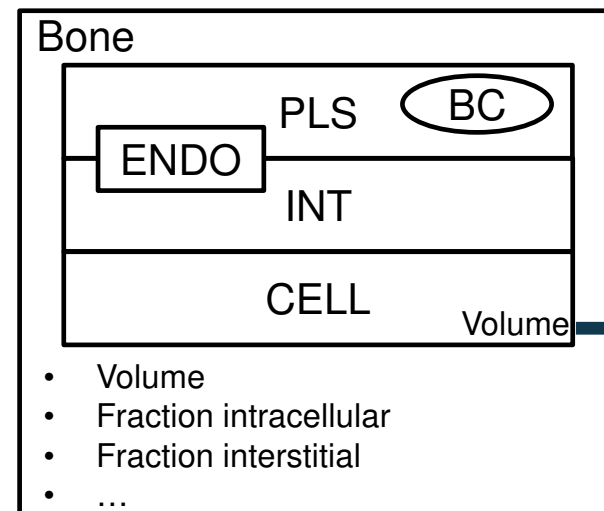
CELL   Volume → Volume_**Bone** * Fraction_intracellular_**Bone**

- Volume
- Fraction intracellular
- Fraction interstitial
- ...

...

„Generic" modeling: Formulas in building blocks

- Better solution: ONE formula shared between all tissue organs – possible due to the usage of **"generic" reference**s to the formula items
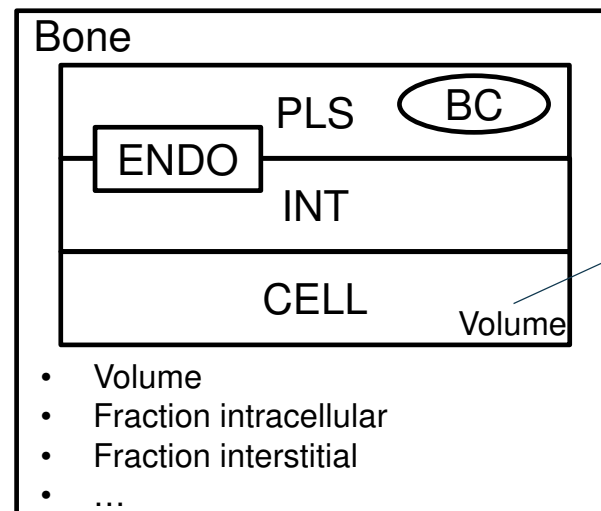

Brain

PLS   BC

ENDO

INT

CELL   Volume

- Volume
- Fraction intracellular
- Fraction interstitial
- ...

Bone

PLS   BC

ENDO

INT

CELL   Volume

- Volume
- Fraction intracellular
- Fraction interstitial
- ...

...

Volume_**PARENT_ORGAN** *
Fraction_intracellular_**PARENT_ORGAN**

| Formula type: | Formula (an explicit formula) |
| Formula name: | VOLUME_cell_4Comp |

Alias	Path		Dimension
f_cell	..|..	Fraction intracellular	Dimensionless
V	..|..	Volume	Volume

f_cell*V

„Generic" modeling: Formulas

- Example 2: Blood flow transports from arterial blood plasma to (tissue) organs plasma:

```
Rate ~ Concentration_arterial *
       BloodFlow_organ
```

- N tissue organs

- M transported molecules

- But just one formula for the transport rate!

```
Blood_Flow_rate_TARGET_ORGAN  *
Concentration_SOURCE_MOLECULE *
...
```
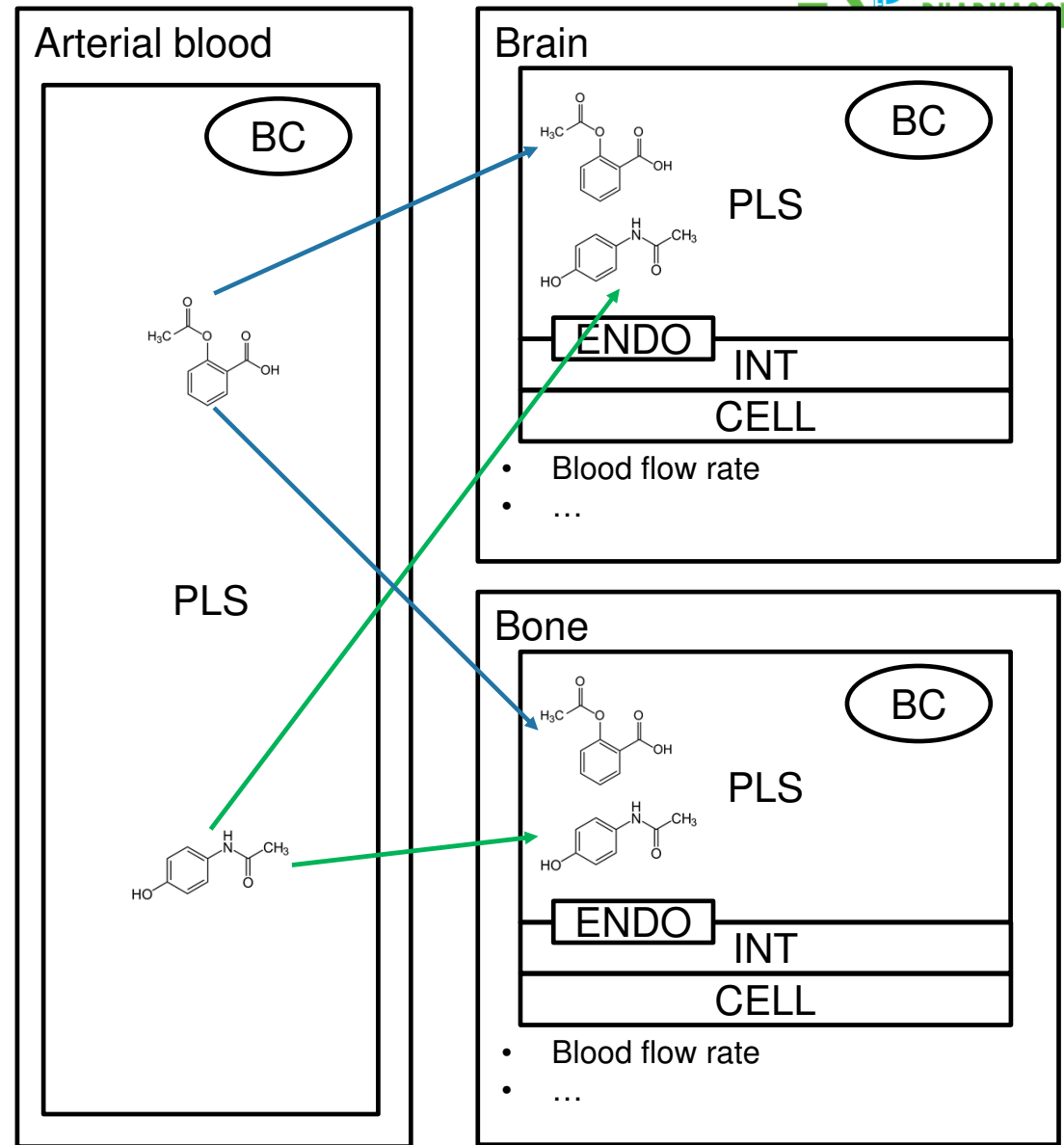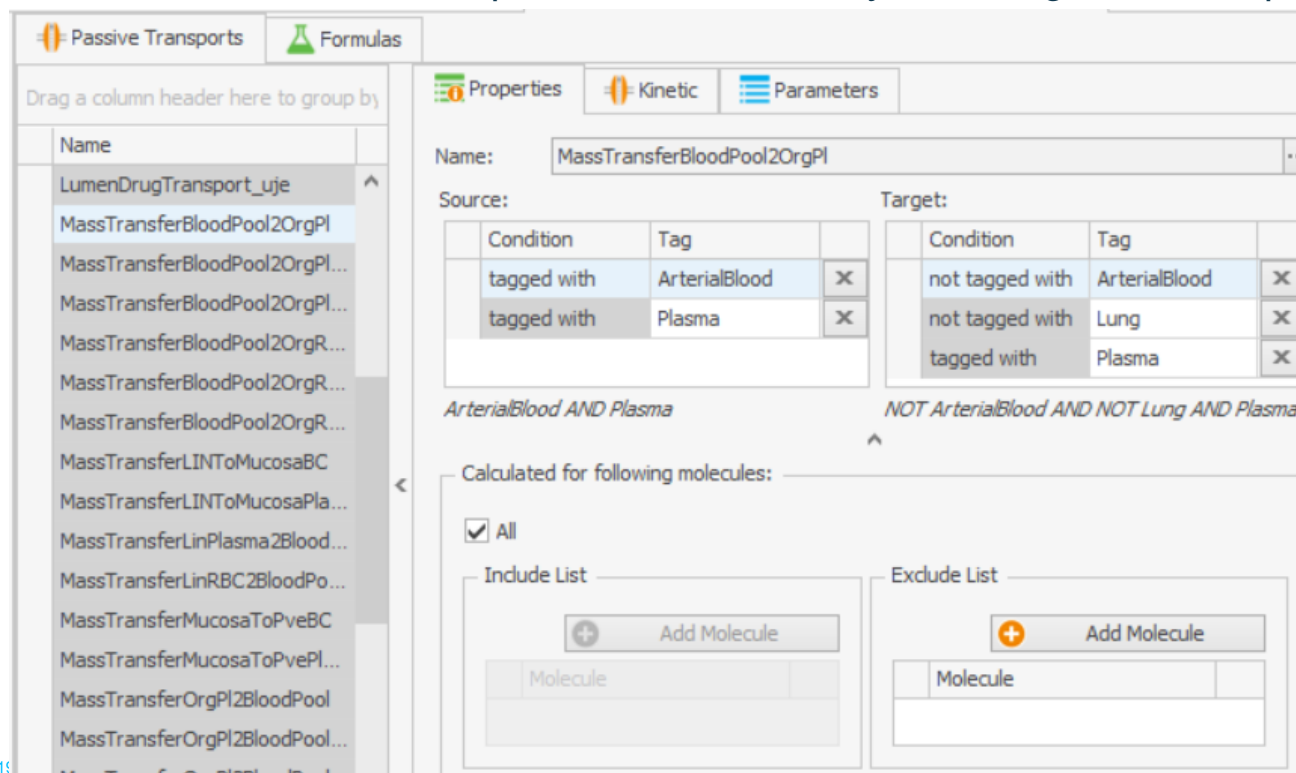
Formula type:	Formula (an explicit formula)	
Formula name:	MassTransferBloodPool2OrgPl	

Alias	Path	Dimension
Q	TARGET|..|Blood flow rate	Flow
C_pls	SOURCE|MOLECULE|Concentration	Concentration (molar)
HCT	Organism|Hematocrit	Dimensionless

dN/dt = Q*C_pls*(1-HCT)

Arterial blood

BC

PLS

Brain

BC

PLS

ENDO INT

CELL

- Blood flow rate
- ...

Bone

BC

PLS

ENDO INT

CELL

- Blood flow rate
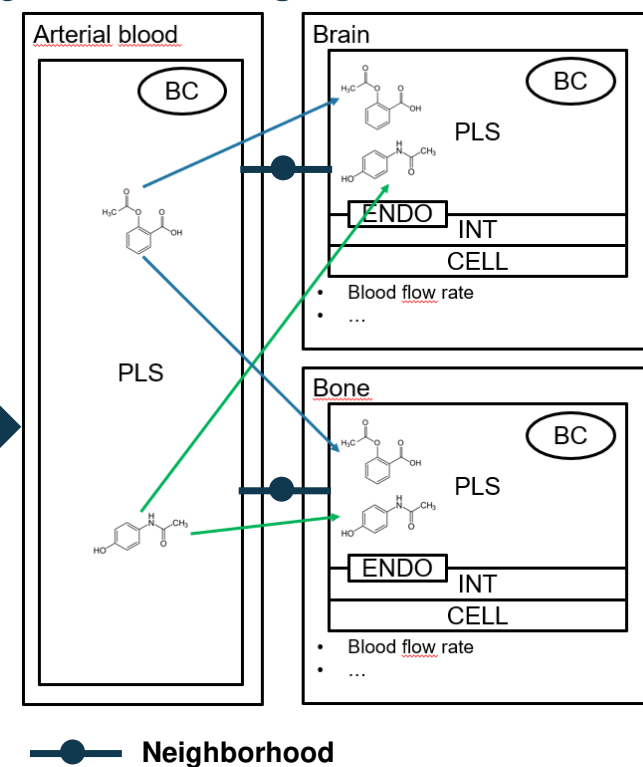- ...

Passive Transports

- Passive transports will affect non-stationary molecules. Examples are passive diffusion, the flow of body fluids like blood, or perfusion processes.

- A passive transport is defined by source and target, while the transport rate is defined by a kinetic formula.

- Often, it is desired to define transport processes by a generic type of equation, e.g., in all organs from blood to interstitial space. This is done by selecting the corresponding **container tag conditions**.
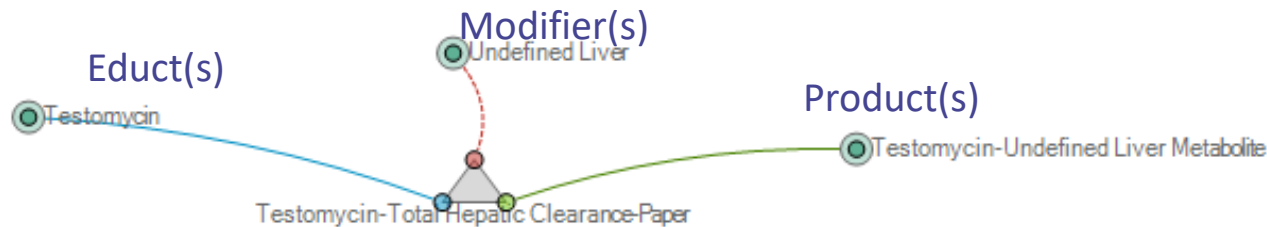
Reactions and reaction networks

- In a Reactions building block, all (bio-)chemical reactions which are of interest for the current project are defined.

- A reaction is defined by lists of reaction partners (educts, products and modifiers), stoichiometry and reaction rate

- Reactions are defined **independent of the location** and take place wherever all reaction partners are present in non-zero amounts.
  - Can be restricted by **container criteria**

Further Building Blocks

- **Observers**: An observer which can be displayed in a chart is an output derived from one or several molecules or parameters by a defined formula.

- **Events** (and Applications):

  - An **event** is used to change an entity, like the amount of molecules or a reaction rate, when a given condition is met. Examples: food intake, sports activities, …

  - An application will be (in almost all cases) will be created within PK-Sim® and then transferred to MoBi®

- **Simulation Settings** include some additional setting like

  - *output time intervals* for which results should be generated

  - Numerical settings of the (ODE) solver

  - output selections (quantities which will be shown in charts)

  - (Default) chart templates

- **Observed data**: experimental data can be imported and used in VPC-plots, for (automated) parameter identification etc.

OSP R-Toolbox

R-Toolbox: main features

- R-Toolbox provides methods to read, manipulate, and simulate (and write) pkml models

  - Change *values* of *parameters* and *start values* of *molecules*

  - Get simulated results as vector of values

  - But: You CANNOT change the structure or equations of the model!

- Creation of individuals and populations

- Individual and Population simulations

- PK-analysis (**user-defined** PK-parameters can be easily created)

- Sensitivity analysis

- Supports Windows and Linux (runs **standalone** - OSP Suite Installation not required)

R-Toolbox: motivation

- Why use R with OSPS?

  - _Flexibility_ and _reproducibility_ – create figures that are fit for your objective

  - _Automation_ – create your own workflows

  - _Advanced parameter identification_ – use advanced algorithms, objective functions…

  - _Save time!_ - parallel simulations (incl. ready to use **Docker container**, allowing simple plug&play into HPC/Cloud environments (e.g., Amazon Web Services (**AWS**)) via the **Docker** technology)

General Workflow
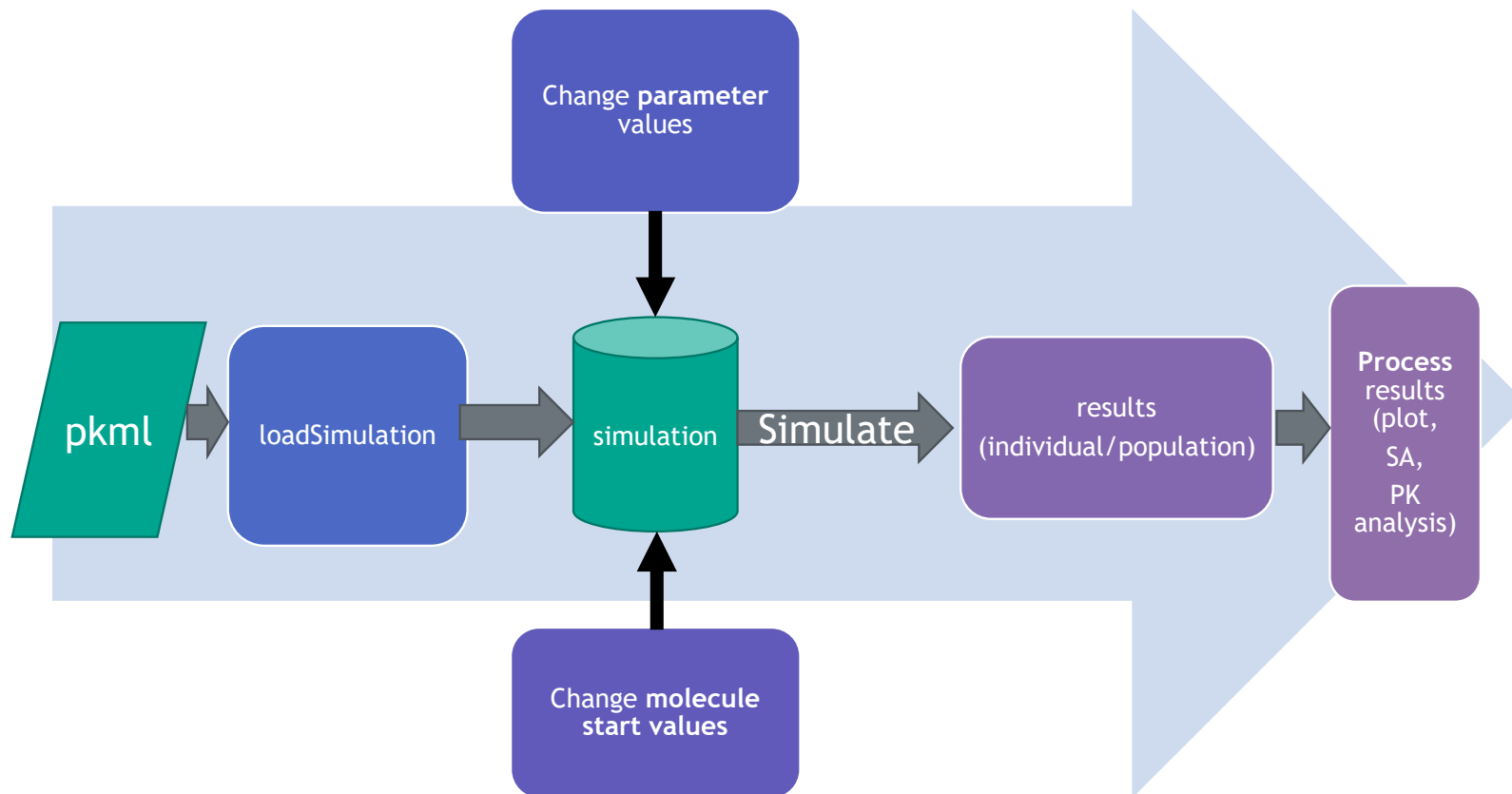
Example 1: run individual simulation, change parameters

```r
library(ospsuite)

# Load simulation
sim <- loadSimulation("Aciclovir.pkml")

# Change parameter value (liver volume)
liverVolume <- getParameter("Organism|Liver|Volume", sim)
print(liverVolume)
#> Parameter:
#>    Path: Organism|Liver|Volume
#>    Value: 2.1675 [l]
#>    isDistributed: TRUE
#>    isStateVariable: FALSE
setParameterValues(liverVolume, 1)
print(liverVolume)
#> Parameter:
#>    Path: Organism|Liver|Volume
#>    Value: 1.00 [l]
#>    isDistributed: TRUE
#>    isStateVariable: FALSE

# Run simulation
simulationResults <- runSimulation(simulation = sim)

# Get the first simulated output and plot it
resultsPath <- simulationResults$allQuantityPaths[[1]]
print(resultsPath)
#> [1] "Organism|PeripheralVenousBlood|Aciclovir|Plasma (Peripheral Venous Blood)"
resultsData <- getOutputValues(simulationResults, quantitiesOrPaths = resultsPath)
resultsTime <- resultsData$data$Time
resultsValues <- resultsData$data$`Organism|PeripheralVenousBlood|Aciclovir|Plasma (Peripheral Venous Blood)`
plot(resultsTime, resultsValues, type = "l")
```
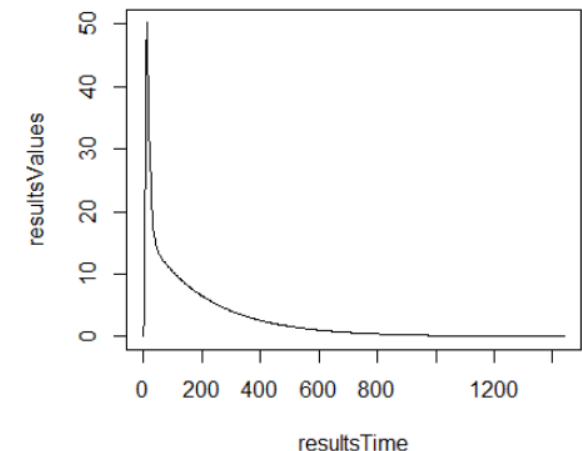
Example 2: create population, run population simulation

```r
library(ospsuite)

# Create population from population characteristics
populationCharacteristics <- createPopulationCharacteristics(
                            species = Species$Human,
                            population = HumanPopulation$Asian_Tanaka_1996,
                            numberOfIndividuals = 50,
                            proportionOfFemales = 50,
                            weightMin = 30, weightMax = 98, weightUnit = "kg",
                            heightMin = NULL, heightMax = NULL,
                            ageMin = 0, ageMax = 80, ageUnit = "year(s)")

myPopulation <- createPopulation(populationCharacteristics = populationCharacteristics)

# Load simulation
sim <- loadSimulation("Aciclovir.pkml")

# Run population simulation
populationResults <- runSimulation(simulation = sim, population = myPopulation)
print(populationResults)
#> SimulationResults:
#>    Number of individuals: 50

# Get the first simulated output and plot it
resultsPath <- populationResults$allQuantityPaths[[1]]
print(resultsPath)
#> [1] "Organism|PeripheralVenousBlood|Aciclovir|Plasma (Peripheral Venous Blood)"
resultsData <- getOutputValues(populationResults, quantitiesOrPaths = resultsPath)
resultsTime <- resultsData$data$Time
resultsValues <- resultsData$data$`Organism|PeripheralVenousBlood|Aciclovir|Plasma (Peripheral Venous Blood)`
plot(resultsTime, resultsValues, type = "l")
```
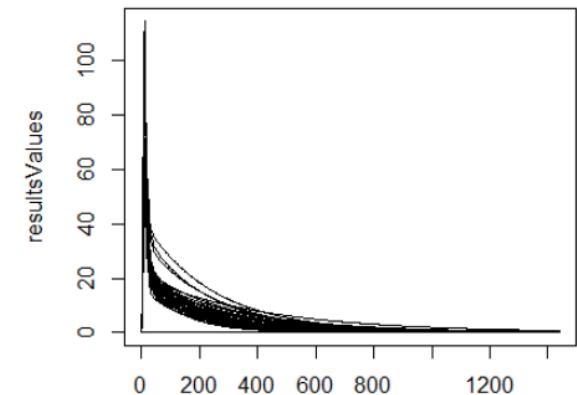
Example 3: Calculate pk values in a population simulation

```r
library(ospsuite)

# Create population from population characteristics
populationCharacteristics <- createPopulationCharacteristics(
  species = Species$Human,
  population = HumanPopulation$Asian_Tanaka_1996,
  numberOfIndividuals = 50,
  proportionOfFemales = 50,
  weightMin = 30, weightMax = 98, weightUnit = "kg",
  heightMin = NULL, heightMax = NULL,
  ageMin = 0, ageMax = 80, ageUnit = "year(s)")

myPopulation <- createPopulation(populationCharacteristics = populationCharacteristics)

# Load simulation
sim <- loadSimulation("Aciclovir.pkml")

# Run population simulation
populationResults <- runSimulation(simulation = sim, population = myPopulation)

# Calculate PK-analyses
pkAnalysis <- calculatePKAnalyses(results = populationResults)

# print available PK parameter names
pkAnalysis$allPKParameterNames
#>    [1] "C_max"       "C_max_norm"       "t_max"     "C_tEnd"     "AUC_tEnd"          "AUC_tEnd_norm"
#>    [7] "AUC_inf"     "AUC_inf_norm"     "MRT"       "Thalf"      "FractionAucLastToInf"  "CL"
#>    [13] "Vss"        "Vd"

# Get C_max parameter
c_max <- pkAnalysis$pKParameterFor(quantityPath = resultsPath, pkParameter = "C_max")
print(c_max)
#>    Organism|PeripheralVenousBlood|Aciclovir|Plasma (Peripheral Venous Blood): C_max
#>    Dimension: Concentration (molar)
#>    Unit: µmol/l
c_max$values
#>    [1]   56.58231  53.85415  55.56487  53.30254  54.23650  59.40544  96.93449  54.66903  59.37316  60.1951
#>    [15]  52.09306 106.74448  80.44911  66.90536 102.34718  58.16575  58.75869  66.63985  83.09749  53.540
#>    [29]  66.23164  65.73529  72.27832  63.65108  60.89072  69.34486  61.48062  74.38184  71.07183  63.598
#>    [43]  66.29802  61.69744  57.31629  60.77491  78.32831  61.15554  78.40282  81.85518
```

Documentation

https://docs.open-systems-pharmacology.org/working-with-r/r-introduction

Questions?