Published in final edited form as:
Computation (Basel). 2014 December ; 2(4): 246–257. doi:10.3390/computation2040246.

SBMLSimulator: A Java Tool for Model Simulation and Parameter Estimation in Systems Biology

Alexander Dörr[1,†,*], **Roland Keller**[1,†], **Andreas Zell**[1], **Andreas Dräger**[1,2]

[1]Center for Bioinformatics Tuebingen (ZBIT), University of Tuebingen, Sand 1, 72076 Tübingen, Germany

[2]Systems Biology Research Group, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093, USA

Abstract

The identification of suitable model parameters for biochemical reactions has been recognized as a quite difficult endeavor. Parameter values from literature or experiments can often not directly be combined in complex reaction systems. Nature-inspired optimization techniques can find appropriate sets of parameters that calibrate a model to experimentally obtained time series data. We present SBMLsimulator, a tool that combines the Systems Biology Simulation Core Library for dynamic simulation of biochemical models with the heuristic optimization framework EvA2. SBMLsimulator provides an intuitive graphical user interface with various options as well as a fully-featured command-line interface for large-scale and script-based model simulation and calibration. In a parameter estimation study based on a published model and artificial data we demonstrate the capability of SBMLsimulator to identify parameters. SBMLsimulator is useful for both, the interactive simulation and exploration of the parameter space and for the large-scale model calibration and estimation of uncertain parameter values.

Keywords

Systems Biology Markup Language (SBML); ordinary differential equation (ODE) modeling; simulation; parameter estimation

[*]Author to whom correspondence should be addressed; alexander.doerr@uni-tuebingen.de; Tel.: +49-7071-29-77174; Fax: +49-7071-29-5091.
[†]These authors contributed equally to this work.

Author Contributions
AlD and RK contributed equally, implemented the majority of the source code, and declare shared first authorship. AnD initialized and coordinated the project, drafted the manuscript, and supervised the work together with AZ. All authors contributed to the implementation, read and approved the final manuscript.

Conflicts of Interest
The authors declare no conflict of interest.

1. Introduction

The dynamic simulation of quantitative biological models belongs to the key aspects of research in systems biology [1]. Biological network models are typically encoded in Extensible Markup Language (XML)-based description formats [2], such as the Systems Biology Markup Language (SBML) [3]. In the most common scenario, such a model can be interpreted in terms of an ordinary differential equation (ODE) system with additional constructs, such as discrete events, delays, or algebraic rules [4,5]. A graphical display of the resulting curves can greatly facilitate the analysis of the system. The calculation of these dynamics requires the initial values of each model component to be known, including the concentration of reactive species as well as parameters, such as Michaelis constants. However, in many cases some of these values are missing or at least uncertain [6,7]. This makes their estimation or adjustment with respect to given experimental data necessary [8] (e.g., available in Gene Expression Omnibus [9] or KiMoSys databases [10]). To this end, heuristic optimization routines can be applied to fit the model to the experimental data. Nature-inspired optimization methods are known to handle even highly nonlinear optimization problems [11]. The basic idea of these methods is that many natural processes can be seen as optimization tasks, for instance the evolution of a population or the flight of a swarm of birds. In several applications, the Java™ framework EvA2 [12] has been shown to be promising for the optimization of biological systems [11,13–16].

Many systems biology simulation and optimization frameworks are available, e.g., AMIGO [17], SBToolbox2 [18], SBML-PET [19], COPASI [20], or Potters-Wheel [21]. The focus of SBMLsimulator is to provide the scientific community with an easily usable and flexible parameter estimation tool that understands and supports all aspects of the modeling format SBML through all of its levels and versions. Programs that do not completely cover the full standard, might have run-time advantages under certain conditions, but cannot guarantee that all models can be solved. Important features of SBMLsimulator include that it (a) is a free platform-independent open-source solution (b) does not depend on any commercial software (c) fully supports all specifications of SBML (d) comes with an easily usable program layout.

The SBMLsimulator project unifies two powerful libraries in one tool with a common user interface. Consequently, SBMLsimulator benefits from all the optimization algorithms provided by EvA2 [12] and as well as from all the modeling languages and ODE solvers that the Systems Biology Simulation Core Library (SBSCL) makes available. This modular program design opens the door to independently extend and exchange both libraries. SBMLsimulator hence evolves with the improvements of either one of its underlying libraries. With little effort, SBMLsimulator can then be modified to take advantage on all enhancements. The SBSCL already supports all elements of SBML and comprises several ODE solvers including a solver for stiff ODE systems (for a comparison of SBSCL to related simulation frameworks see [5] and the up-to-date comparisons of the SBML standard compliance [22]). EvA2 provides a plethora of nature-inspired heuristic optimization algorithms, such as evolution strategies [23], genetic algorithm [24], differential evolution [25], and particle swarm optimization [26], and many more. Teams of core developers maintain both open-source libraries SBSCL and EvA2 that can also be easily extended by the scientific community.

2. Implementation

The simulation of a model is done with the SBSCL [5] and JSBML [27] is used as internal data structure for model representation. The SBSCL outputs the simulation results in a specific format, which enables SBMLsimulator to plot the time course of the model's species.

The basis of all those routines is a target function for model calibration, the so-called *fitness* function. Given a set of parameter values and experimental data, this function returns a value to EvA2 that reflects the quality of the current model configuration. That is, how precise the given set of parameters can reproduce the experimental data. For model calibration, the fitness is a distance function between simulation output and experimental data. Both, simulation and distance, are computed by the SBSCL. EvA2 optimizes the parameters of the model with respect to the fitness function. Hence, EvA2 tries to find a parameter set that accounts for the smallest possible deviation of simulation results and the given experimental data set. Before a parameter estimation can be performed, optimization targets and search intervals need to be specified by the user (see description in Results section).

The estimation with EvA2 can take several hours or days, as for each fitness evaluation a simulation has to be conducted. Hence, the time of one simulation run greatly influences the time needed for parameter estimation. Such parameter estimations will often be conducted in the command-line mode, which is an alternative to the graphical user interface (GUI) that SBMLsimulator launches by default. However, in the GUI mode the simulation results of the current parameter combination producing the best fitness are always plotted in SBMLsimulator. This enables the user to investigate intermediate results.

3. Results and Discussion

SBMLsimulator is straightforward to use and runs on every platform where a Java Virtual Machine is available. Its graphical user interface provides an intuitive way for displaying the curves of model entities chosen by the user. In addition, it comprises a graphical and command-line user interface that both provide a connection to EvA2. The program estimates all uncertain quantities with respect to given time-series of metabolite or gene expression values.

3.1. Graphical User Interface

The graphical user interface (see Figure 1) comprises several separated sub-windows for the presentation of simulation results as well as for settings that can be modified by the user: at the lower part of the GUI, SBMLsimulator enables the user to choose the numerical solver and to set the start and end point as well as the step size of the simulation. For some solvers an error tolerance can also be specified under Edit/Preferences. The user also has the choice between different quality functions for calculating the distance between simulated and experimental data. The upper left part of the window allows the user to select, which model quantities should be plotted. Furthermore, the compartment, parameter and initial values of metabolites can be modified in the middle left part. Finally, the simulation of the model with the chosen settings can be conducted by one of the GUI's control elements.

A dedicated dialog assists with the import of experimental data. SBMLsimulator suggests the most intuitive mapping of columns in the experimental data to model quantities. The user can then confirm this suggestion or adapt the mapping accordingly. The given experimental data are plotted together with the simulation results (as shown in Figure 1) in the user interface in order to facilitate the comparison with the simulation results of the model. Furthermore, after model simulation, the distance of the simulated data to the loaded experimental data is computed and displayed at the bottom of the SBMLsimulator window.

Besides the main window (Simulation), which shows the simulation results and enables the modification of simulation settings, there are also windows for displaying the simulation data (Computed data) and the imported experimental data (Experimental data) in table format. Furthermore, the user has the possibility to investigate the structure of the model in another window (Model).

Besides a comparison with simulated values, the import of experimental data facilitates a parameter estimation with respect to that data is possible. The user needs to specify the parameters as well as their ranges in a dedicated window. An alternative provided to set the parameters in this window, is to import a text file listing the parameters to estimate and their ranges. After the user has chosen the parameters and their limits, the user can select the type of evolutionary algorithm and change specific settings of the desired routine in EvA2. Eva2 lets the user choose between a large number of different optimization routines including hill climbing [29], simulated annealing [30], evolution strategies [23,31] with support for covariance matrix adaptation [32], genetic algorithm [24], differential evolution [25,33], and particle swarm optimization [26,34,35] as well as multi-modal approaches [15].

3.2. Parameter Estimation Study

In order to assess the capabilities of SBMLsimulator, we estimated the parameters of a model by Bucher *et al.* [28] explaining the biotransformation of atorvastatin (model BIOMD0000000328 of BioModels Database [36]). We first obtained an artificial data set by simulating the model with the respective start and end time described in the publication. We saved the simulated data and extracted time points similar to those used in the publication. This data set was then read in by SBMLsimulator, the parameters were deleted from the model, and optimization with EvA2 was launched with respect to the artificial data set using differential evolution [25]. The same parameters as in the publication were estimated with the intervals given in Table 1.

Here initial minimum/maximum stands for the interval, in which the parameters are randomly initiated, and minimum/maximum for the interval during the estimation procedure. We chose the same intervals for initiation and parameter estimation. The numerical integration was conducted with a Rosenbrock solver [37] (absolute error tolerance: 10^{-12}, relative error tolerance: 10^{-6}), as this routine is suitable for solving stiff differential equation systems. As fitness function we used the relative squared error, which has been suggested by, e.g., Dräger *et al.* [11,38].

During an optimization, SBMLsimulator plots the best simulation curves together with the respective data each time it has finished processing a new generation of parameter combinations. Figure 1 shows such an intermediate result for the artificial data set. Hence, the user can nicely see how the distance between the simulated time course and data set drastically decreases over time. For such large-scale parameter estimations it is most suitable to run SBMLsimulator in command-line mode, because it can take a long time (see Section 5) and this task can then be performed on a computer cluster. Running parameter estimations on a computer cluster is also recommended, because estimations are usually conducted multiple times. This is done because an optimization run can get stuck in a local optimum of the fitness function. Another reason for repeating a parameter estimation multiple times is that this is a straightforward way to assess whether the estimated quantities are identifiable [39]. In order to test the identifiability of the parameters with SBMLsimulator, we estimated the model quantities of the atorvastatin biotransformation model 100 times on a computer cluster. Afterwards those 50% of the parameter estimates with the best fits were extracted and the distributions of the estimated quantity values were plotted (see Figure 2). It can be seen that the estimated parameter values had a low standard deviation, which means that SBMLsimulator could reliably identify these parameters. A very high standard deviation of a parameter would in contrast to that have suggested that this parameter could not be identified.

The results of this experiment show that our linkage of parameter estimation and simulation works well.

4. Conclusions

With SBMLsimulator we provide a freely usable platform-independent tool for simulation and parameter estimation of biochemical models, which fully supports all SBML elements. It combines two powerful toolboxes under one graphical interface: SBSCL for simulation and the nature-inspired optimization framework EvA2 for estimation of uncertain parameter values. SBMLsimulator is useful for both, the interactive simulation and exploration of the parameter space and for the large-scale model calibration. With a parameter estimation study based on a published model we have demonstrated its capability to identify model parameters with respect to experimental data. SBMLsimulator allows users to easily run models and to calibrate them to their experiments, to learn about simulation, or to gain new insight into the model's behavior by modifying individual parameters or model components. All this is facilitated by the platform independence of SBMLsimulator. Since the SBSCL is also an open-source library, the user can even modify how different elements of a model are interpreted and learn from deviating simulation results. For large-scale simulation experiments, the use of native libraries can offer an advantage with respect to calculation speed. However, with a comprehensive command-line interface SBMLsimulator can also easily be deployed on a computer cluster to perform several simulations in parallel. A comprehensive users' guide is available on the project's homepage, which describes all program features and their use in detail.

In future versions of SBMLsimulator we are planning to include the possibility that the parameter search is conducted in log-scale. As the parameters are often of different magnitudes, this log-scale can be of advantage [16]. In order to satisfy the growing interest in other modeling languages like the Cell Markup Language (CellML) [41], SBMLsimulator should also be able to simulate models given in a different format. To this end, the SBSCL needs to be extended and after some minor modifications SBMLsimulator will gain the capability to simulate models in different formats, such as the simulation of models given in CellML similar to CellMLSimulator [42].

5. Availability and Requirements

The current version of SBMLsimulator is available at the project's homepage as a runnable Java™ archive file (JAR) together with a documentation of the program.

Project name: SBMLsimulator

Project homepage: http://www.cogsys.cs.uni-tuebingen.de/software/SBMLsimulator/

Contact: sbmlsimulator@googlegroups.com

Operating systems: Platform independent, i.e., for all systems for which a Java™ Virtual Machine (JVM) is available.

Programming language: Java™

Other requirements: Java™ Runtime Environment (JRE) 1.6 or above

License: GNU Lesser General Public License (LGPL) version 3

Acknowledgments

The authors thank Stephanie Hoffmann, Benjamin Kandel, Marcus Klein, Maria Thomas, Ulrich M. Zanger, and Bernhard Ø. Palsson for fruitful discussions, Marcel Kronfeld who helped implementing the connection to EvA2, and the students Max Zwießele and Philipp Stevens for implementing parts of the plot function.

This work was funded by the Federal Ministry of Education and Research (BMBF, Germany) in the project Virtual Liver Network (BMBF, Germany, grant number 0315756), the European Commission as part of a Marie Curie International Outgoing Fellowship within the EU 7th Framework Program for Research and Technological Development (EU project AMBiCon, 332020), and a National Institute of Health grant for the continued development of essential SBML software support (NIH, USA, award number GM070923).

Abbreviations/Nomenclature

AMIGO	Advanced Model Identification in systems biology using Global Optimization
COPASI	COmplex PAthway SImulator
CellML	Cell Markup Language
GUI	graphical user interface
JAR	Java™ archive file

JRE	Java™ Runtime Environment
JSBML	Java™ SBML
JVM	Java™ Virtual Machine
LGPL	GNU Lesser General Public License
ODE	ordinary differential equation
SBML	Systems Biology Markup Language
SBML-PET	SBML Parameter Estimation Tool
SBSCL	Systems Biology Simulation Core Library
XML	Extensible Markup Language

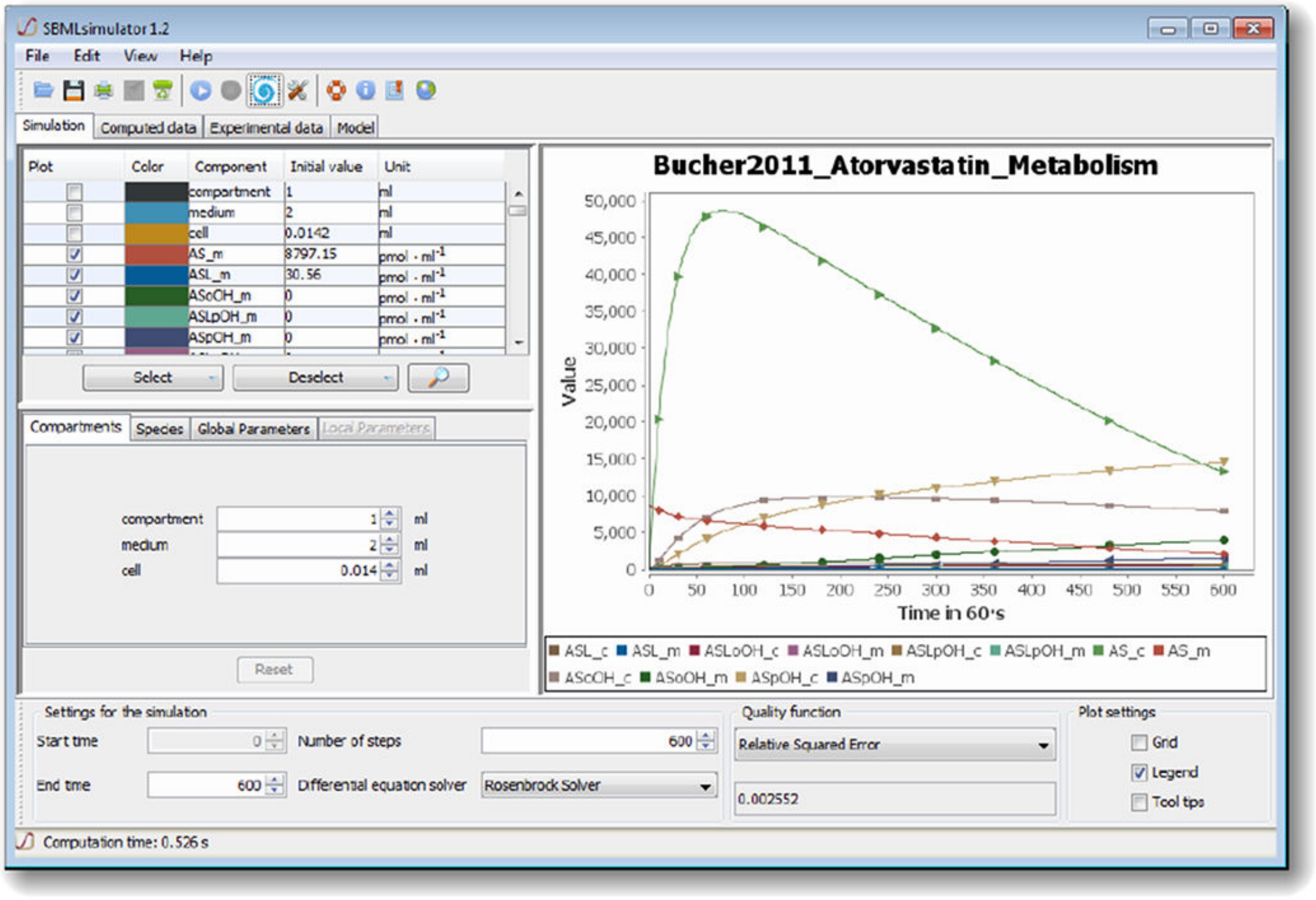
References

1. Jamshidi N; Wiback SJ; Palsson BØ *In Silico* Model-Driven Assessment of the Effects of Single Nucleotide Polymorphisms (SNPs) on Human Red Blood Cell Metabolism. Genome Res. 2002,12, 1687–1692. [PubMed: 12421755]
2. Dräger A; Palsson BØ Improving collaboration by standardization efforts in systems biology. Front. Bioeng 2014, 2, doi: 10.3389/fbioe.2014.00061.
3. Hucka M; Finney A; Bornstein BJ; Keating SM; Shapiro BE; Matthews J; Kovitz BL; Schilstra MJ; Funahashi A; Doyle JC; et al. Evolving a *lingua franca* and associated software infrastructure for computational systems biology: The Systems Biology Markup Language (SBML) project. Syst. Biol. IEE 2004,1, 41–53.
4. Dräger A; Planatscher H Encyclopedia of Systems Biology; Springer-Verlag: New York, NY, USA, 2013; pp. 1249–1251.
5. Keller R; Dörr A; Tabira A; Funahashi A; Ziller MJ; Adams R; Rodriguez N; Le Novère N; Hiroi N; Planatscher H; et al. The systems biology simulation core algorithm. BMC Syst. Biol 2013, 7, 55. [PubMed: 23826941]
6. Costa R; Machado D; Rocha I; Ferreira E Critical perspective on the consequences of the limited availability of kinetic data in metabolic dynamic modelling. IET Syst. Biol 2011, 5, 157–163. [PubMed: 21639589]
7. Chen N; Del VI; Kyriakopoulos S; Polizzi K; Kontoravdi C Metabolic network reconstruction: Advances in *in silico* interpretation of analytical information. Curr. Opin. Biotechnol 2012, 23, 77–82. [PubMed: 22119273]
8. Dräger A; Planatscher H Encyclopedia of Systems Biology; Springer-Verlag: New York, NY, USA, 2013; pp. 1627–1631.
9. Barrett T; Wilhite SE; Ledoux P; Evangelista C; Kim IF; Tomashevsky M; Marshall KA; Phillippy KH; Sherman PM; Holko M; et al. NCBI GEO: Archive for functional genomics data sets-update. Nucl. Acids Res 2013, 41, 991–995.
10. Costa RS; Veríssimo A; Vinga S KiMoSys: A web-based repository of experimental data for KInetic MOdels of biological SYStems. BMC Syst. Biol 2014, 8, 85. [PubMed: 25115331]
11. Dräger A; Kronfeld M; Ziller MJ; Supper J; Planatscher H; Magnus JB; Oldiges M; Kohlbacher O; Zell A Modeling metabolic networks in *C. glutamicum*: A comparison of rate laws in combination with various parameter optimization strategies. BMC Syst. Biol 2009, 3, 5. [PubMed: 19144170]
12. Kronfeld M; Planatscher H; Zell A The EvA2 Optimization Framework In Learning and Intelligent Optimization; Blum C, Battiti R, Eds.; Springer Verlag: Venice, Italy, 2010; pp. 247–250.
13. Dräger A; Supper J; Planatscher H; Magnus JB; Oldiges M; Zell A Comparing Various Evolutionary Algorithms on the Parameter Optimization of the Valine and Leucine Biosynthesis in

Corynebacterium glutamicum. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 620–627.

14. Dräger A; Kronfeld M; Supper J; Planatscher H; Magnus JB; Oldiges M; Zell A Benchmarking Evolutionary Algorithms on Convenience Kinetics Models of the Valine and Leucine Biosynthesis in *C. glutamicum*. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 896–903.
15. Kronfeld M; Dräger A; Aschoff M; Zell A On the Benefits of Multimodal Optimization for Metablic Network Modeling. In Proceedings of the German Conference on Bioinformatics, Halle, Germany, 30 September 2009; pp. 191–200.
16. Raue A; Schilling M; Bachmann J; Matteson A; Schelke M; Kaschek D; Hug S; Kreutz C; Harms BD; Theis FJ; Klingmüller U; Timmer J Lessons Learned from Quantitative Dynamical Modeling in Systems Biology. PLoS ONE 2013, 8, e74335. [PubMed: 24098642]
17. Balsa-Canto E; Banga JR AMIGO, a toolbox for advanced model identification in systems biology using global optimization. Bioinformatics 2011, 27, 2311–2313. [PubMed: 21685047]
18. Schmidt H; Jirstrand M Systems Biology Toolbox for MATLAB: A computational platform for research in systems biology. Bioinformatics 2006, 22, 514–515. [PubMed: 16317076]
19. Zi Z; Klipp E SBML-PET: A Systems Biology Markup Language-based parameter estimation tool. Bioinformatics 2006, 22, 2704–2705. [PubMed: 16926221]
20. Hoops S; Sahle S; Gauges R; Lee C; Pahle J; Simus N; Singhal M; Xu L; Mendes P; Kummer U COPASI—A COmplex PAthway SImulator. Bioinformatics 2006,22, 3067–3074. [PubMed: 17032683]
21. Maiwald T; Timmer J Dynamical modeling and multi-experiment fitting with PottersWheel. Bioinformatics 2008, 24, 2037–2043. [PubMed: 18614583]
22. Bergmann FT; Hucka M; Smith L; Keating SM SBML Test Suite Database. Available online: http://sbml.org/Facilities/Database/ (accessed on 17 December 2014).
23. Rechenberg I Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution; Frommann-Holzboog: Stuttgart, Germany, 1973 (In German)
24. Holland J Adaptation in Natural and Artificial Systems; University of Michigan Press: Ann Arbor, MI, USA, 1975.
25. Storn R; Price K Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Opt 1997,11, 341–359.
26. Kennedy J; Eberhart R Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November-1 December 1995; Volume 4, pp. 1942–1948.
27. Dräger A; Rodriguez N; Dumousseau M; Dörr A; Wrzodek C; le Novère N; Zell A; Hucka M JSBML: A flexible Java library for working with SBML. Bioinformatics 2011, 27, 2167–2168. [PubMed: 21697129]
28. Bucher J; Riedmaier S; Schnabel A; Marcus K; Vacun G; Weiss TS; Thasler WE; Nüssler AK; Zanger UM; Reuss M A systems biology approach to dynamic modeling and inter-subject variability of statin pharmacokinetics in human hepatocytes. BMC Syst. Biol 2011, 5, 66. [PubMed: 21548957]
29. Tovey CA Hill climbing with multiple local optima. SIAM J. Algebr. Discret. Methods 1985, 6, 384–393.
30. Kirkpatrick S; Gelatt CD; Vecchi MP Optimization by Simulated Annealing. Science 1983, 220, 671–680. [PubMed: 17813860]
31. Schwefel HP Evolutionsstrategie und Numerische Optimierung. Ph.D. Thesis, Department of Process Engineering, Technical University of Berlin, Berlin, Germany, 1975 (In German)
32. Hansen N; Ostermeier A Completely Derandomized Self-Adaptation in Evolution Strategies. Evolut. Comput 2001, 9, 159–195.
33. Storn R. On the Usage of Differential Evolution for Function Optimization; Proceedings of the 1996 Biennial Conference of the North American Fuzzy Information Processing Society; Berkeley, CA, USA. June 1996; 519–523.
34. Clerc M; Kennedy J The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Trans. Evolut. Comput 2002, 6, 58–73.

35. Clerc M Particle Swarm Optimization; ISTE Ltd: London, UK, 2005; pp. 139–149.
36. Chelliah V; Laibe C; le Novère N BioModels Database: A Repository of Mathematical Models of Biological Processes In In Silico Systems Biology; Schneider MV, Ed.; Springer: New York, NY, USA, 2013; Volume 1021, pp. 189–199.
37. Press WH; Teukolsky SA; Vetterling WT; Flannery BP Numerical Recipes in FORTRAN; The Art of Scientific Computing; Cambridge University Press: New York, NY, USA, 1993.
38. Dräger A Computational Modeling of Biochemical Networks. Ph.D. Thesis, University of Tuebingen, Tübingen, Germany, 31 3 2011.
39. Schilling M; Maiwald T; Hengl S; Winter D; Kreutz C; Kolch W; Lehmann WD; Timmer J; Klingmüller U Theoretical and experimental analysis links isoform-specific ERK signalling to cell fate decisions. Mol. Syst. Biol 2009, 5.
40. R Development Core Team. R: A Language and Environment for Statistical Computing; R Foundation for Statistical Computing: Vienna, Austria, 2008.
41. Cooling MT A Primer on Modular Mass-Action Modelling with CellML In Systems Biology for Signaling Networks; Choi S, Ed.; Springer: New York, NY, USA, 2010; Volume 1, pp. 721–750.
42. Nickerson DP; Corrias A; Buist ML Reference descriptions of cellular electrophysiology models. Bioinformatics 2008,24, 1112–1114. [PubMed: 18310619]



Plot	Color	Component	Initial value	Unit
☐	⬛	compartment	1	ml
☐	🟦	medium	2	ml
☐	🟧	cell	0.0142	ml
✓	🟥	AS_m	8797.15	pmol · ml^{-1}
✓	🟦	ASL_m	30.56	pmol · ml^{-1}
✓	🟩	ASoOH_m	0	pmol · ml^{-1}
✓	🟩	ASLpOH_m	0	pmol · ml^{-1}
✓	🟦	ASpOH_m	0	pmol · ml^{-1}

compartment 1 ml
medium 2 ml
cell 0.014 ml

Settings for the simulation
Start time 0
End time 600
Number of steps 600
Differential equation solver Rosenbrock Solver

Quality function
Relative Squared Error
0.002552

Plot settings
☐ Grid
✓ Legend
☐ Tool tips

Computation time: 0.526 s

Figure 1.

The graphical user interface (GUI) of The Systems Biology Markup Language (SBML)simulator. The figure shows the main window of SBMLsimulator after importing the model by Bucher *et al.* [28]. SBMLsimulator enables the user to modify initial quantities (middle left part of window) and to choose the quantities for plotting (upper left). Furthermore, at the bottom of the window the user can specify settings for simulation, such as the integration routine, the simulation start and end time, the simulation step size, and the quality function for comparing the simulated data to experimental data. The simulation can be started by clicking on the simulation button. The right part shows an intermediate solution, whereby the original values are depicted by shapes and the simulated values dependent on the current set of parameters are shown as curves. In the given state, the parameter optimization already found a set of parameters that fit the predefined values with a small error.

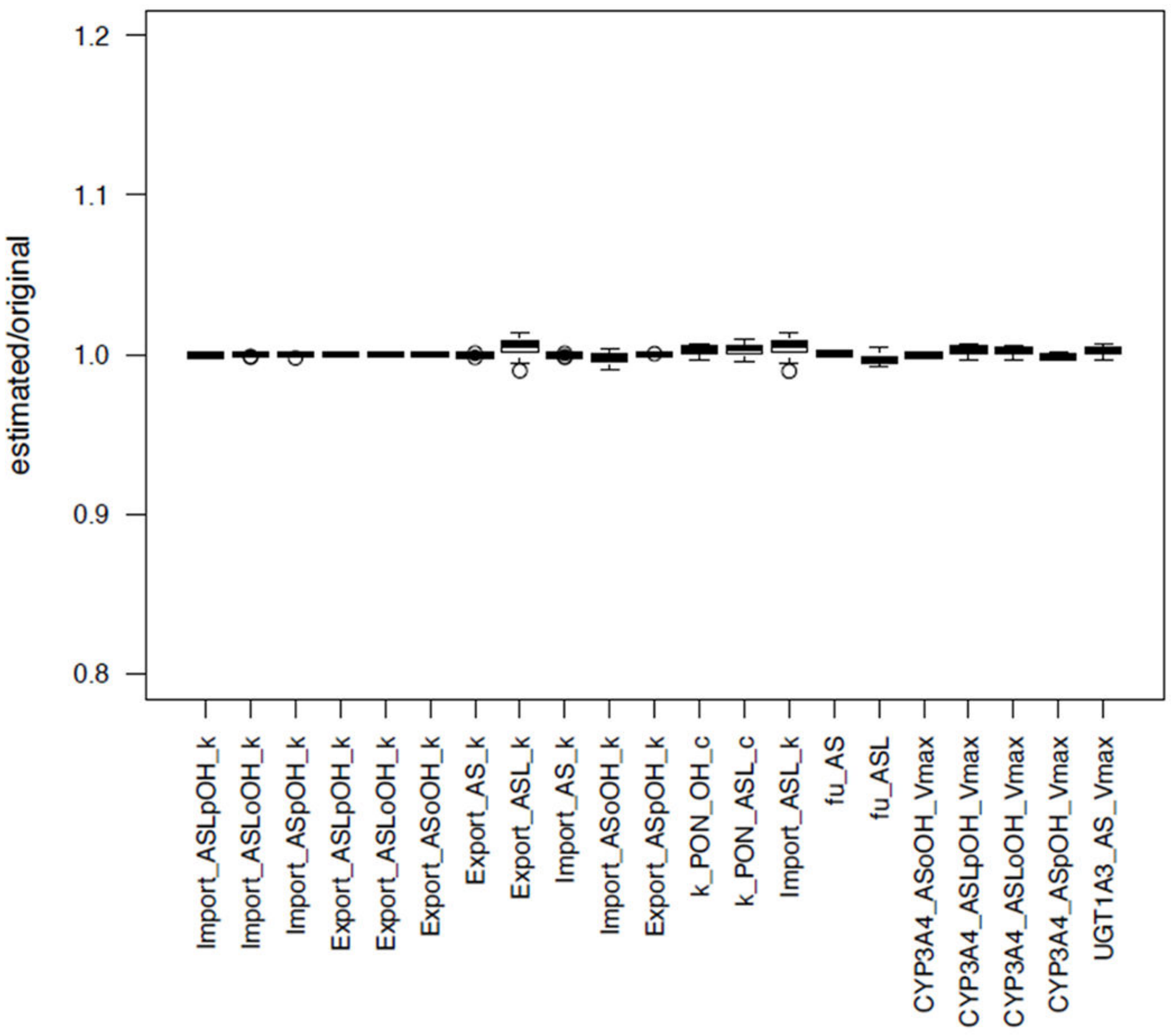


Figure 2.

Distribution of parameters estimated with SBMLsimulator. One hundred parameter estimations with SBMLsimulator for the model by Bucher *et al.* [28] were run on a computer cluster. The distribution of the 50 estimations with the best fitness values is shown here. For each parameter the estimated values were divided by the original parameter value prior to plotting. It is obvious from the plot that all parameters were estimated closely around their original values. The figure has been created with R software package [40].

Table 1.

Estimated parameters with units, their initial intervals and their intervals throughout parameter estimation for the model by Bucher *et al.* [28].

Parameter	Unit	Minimum Initial Value	Maximum Initial Value	Minimum Value	Maximum Value
Import_ASLpOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Import_ASLoOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Import_ASpOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Export_ASLpOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Export_ASLoOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Export_ASoOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Export_AS_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Export_ASL_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Import_AS_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Import_ASoOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Export_ASpOH_k	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
k_PON_OH_c	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
k_PON_ASL_c	$mL \cdot min^{-1}$	10^{-6}	0.1	10^{-6}	0.1
Import_ASL_k	$mL \cdot min^{-1}$	10^{-6}	1	10^{-6}	1
fu_AS	dimensionless	10^{-6}	1	10^{-6}	1
fu_ASL	dimensionless	10^{-6}	1	10^{-6}	1
CYP3A4_ASoOH_Vmax	$pmol \cdot min^{-1}$	10^{-6}	100	10^{-6}	100
CYP3A4_ASLpOH_Vmax	$pmol \cdot min^{-1}$	10^{-6}	100	10^{-6}	100
CYP3A4_ASLoOH_Vmax	$pmol \cdot min^{-1}$	10^{-6}	100	10^{-6}	100
CYP3A4_ASpOH_Vmax	$pmol \cdot min^{-1}$	10^{-6}	100	10^{-6}	100
UGT1A3_AS_Vmax	$pmol \cdot min^{-1}$	10^{-6}	100	10^{-6}	100