BIOINFORMATICS ORIGINAL PAPER Vol. 22 no. 24 2006, pages 3067–3074
doi:10.1093/bioinformatics/btl485

Systems biology

COPASI—a COmplex PAthway SImulator

Stefan Hoops[1,†], Sven Sahle[2,†], Ralph Gauges[2], Christine Lee[1], Jürgen Pahle[2], Natalia Simus[2], Mudita Singhal[1], Liang Xu[1], Pedro Mendes[1,*] and Ursula Kummer[2]

[1]Virginia Bioinformatics Institute, Virginia Tech, Washington St. 0477, Blacksburg, VA 24061, USA and [2]Bioinformatics and Computational Biochemistry, EML Research, Schloss-Wolfsbrunnenweg 33, D-69118 Heidelberg, Germany

Received on June 29, 2006; revised on August 29, 2006; accepted on September 14, 2006
Advance Access publication October 10, 2006
Associate Editor: Jonathan Wren

ABSTRACT

Motivation: Simulation and modeling is becoming a standard approach to understand complex biochemical processes. Therefore, there is a big need for software tools that allow access to diverse simulation and modeling methods as well as support for the usage of these methods.

Results: Here, we present COPASI, a platform-independent and user-friendly biochemical simulator that offers several unique features. We discuss numerical issues with these features; in particular, the criteria to switch between stochastic and deterministic simulation methods, hybrid deterministic–stochastic methods, and the importance of random number generator numerical resolution in stochastic simulation.

Availability: The complete software is available in binary (executable) for MS Windows, OS X, Linux (Intel) and Sun Solaris (SPARC), as well as the full source code under an open source license from http://www.copasi.org.

Contact: mendes@vbi.vt.edu

INTRODUCTION

Simulation and modeling is becoming one of the standard approaches to understand complex biochemical processes. Therefore, there is a growing need for software tools that allow access to diverse simulation and modeling methods as well as support for the usage of these methods. These software tools should be compatible, e.g. via file standards, platform independent and user friendly to avoid time-consuming conversions and learning procedures. In addition, the software should be actively maintained and updated by its authors.

Standard methodology used in the field comprises, e.g. the deterministic [integration of ordinary differential equations (ODEs)] and stochastic [e.g. using Gillespie's algorithm; Gillespie (1976)] simulation of reaction networks, the computation of steady states and their stability, stoichiometric network analysis, e.g. computing elementary modes (Schuster *et al*., 1999), sensitivity analysis [metabolic control analysis; Fell (1996); Heinrich and Shuster, 1997], optimization and parameter estimation.

In order to meet this need for software in the field, several tools have been developed and released recently (see http://www.sbml.org). Most tools offer specific functionalities, e.g. stochastic simulations of reaction networks (Le Novére and Shimizu, 2001) and flux analysis (Klamt *et al*., 2003). However, some tools contain whole suites of functionalities, e.g. simulation, flux and control analysis (Tomita *et al*., 1999; Sauro *et al*., 2003; Meng *et al*., 2004).

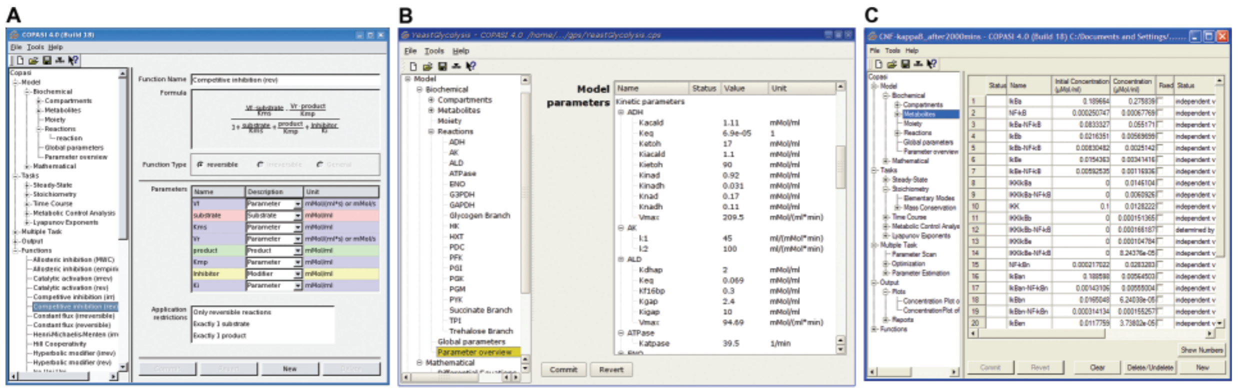
In order to improve the compatibility of these tools, markup languages such as SBML (Hucka *et al*., 2003) and CellML (Lloyd *et al*., 2004) were created to allow model exchange. Many tools are now able to read and write models in these file formats.

Here we present a new program—COPASI (COmplex PAthway SImulator)—which combines all of the above standards and some unique methods for the simulation and analysis of biochemical reaction networks. COPASI is the successor to Gepasi (Mendes, 1993, 1997) and is available for all major operating systems (Linux, Mac OS X, Windows, Solaris). As described below, COPASI supports non-expert users by, for example, automatically converting reaction equations to the appropriate mathematical formalism (ODEs or reaction propensities). The general features of COPASI are described briefly and those that are unique are discussed in more detail.

GENERAL FEATURES

COPASI is a stand-alone program that can be used through two different executable versions: a graphical user interface (CopasiUI) and a command line version (CopasiSE) that only contains the calculation engine. CopasiSE is intended for situations in which the user is not expected to interact with the software. The following use cases are examples of situations in which it would be used: (i) when third-party programs manipulate COPASI files, call CopasiSE to produce results, and then inspect and continue generating other COPASI files depending on results; (ii) to run simulations 'in the background', which is useful when the run takes a long time; (iii) as a simulation engine for specialized front-ends that may be created by others. Essentially, CopasiSE allows much flexibility of execution and control, with the penalty that this version can only run numerical procedures, not edit models. CopasiUI, on the other hand, is the complete version of the program and is the one that we expect users to run most often. CopasiUI provides a full graphical user interface (GUI), including functions for creating and editing models and plotting results. In terms of execution of the numerical procedures (simulation, optimization, etc.) the two versions are essentially equal, except that CopasiUI may be slightly slower when producing graphical output. In practice the two executables share the same source code and are expected to produce exactly the same results.

*To whom correspondence should be addressed.
†These authors wish it to be known that, in their opinion, the first two authors should be regarded as joint first authors.



Function Name: Competitive inhibition (rev)

Formula:

$$\frac{\frac{Vf \cdot substrate}{Kms} - \frac{Vr \cdot product}{Kmp}}{1 + \frac{substrate}{Kms} + \frac{product}{Kmp} + \frac{Inhibitor}{Ki}}$$

Function Type: [✓ reversible] irreversible General

Parameters:

Name	Description	Unit
Vf	Parameter	mMol/(ml*s) or mMol/s
substrate	Substrate	mMol/ml
Kms	Parameter	mMol/ml
Vr	Parameter	mMol/(ml*s) or mMol/s
product	Product	mMol/ml
Kmp	Parameter	mMol/ml
Inhibitor	Modifier	mMol/ml
Ki	Parameter	mMol/ml

Application restrictions:
Only reversible reactions
Exactly 1 substrate
Exactly 1 product

Model parameters

Name	Status	Value	Unit
Kinetic parameters			
ADH			
Kacald		1.11	mMol/ml
Keq		6.9e-05	1
Ketoh		17	mMol/ml
Kiacald		1.1	mMol/ml
Kietoh		90	mMol/ml
Kinad		0.92	mMol/ml
Kinadh		0.031	mMol/ml
Knad		0.17	mMol/ml
Knadh		0.11	mMol/ml
Vmax		209.5	mMol/(ml*min)
AK			
k1		45	ml/(mMol*min)
k2		100	ml/(mMol*min)
ALD			
Kdhap		2	mMol/ml
Keq		0.069	mMol/ml
Kf16bp		0.3	mMol/ml
Kgap		2.4	mMol/ml
Kigap		10	mMol/ml
Vmax		94.69	mMol/(ml*min)
ATPase			
Katpase		39.5	1/min

	Status	Name	Initial Concentration (μMol/ml)	Concentration (μMol/ml)	Fixed	Status
1		IkBa	0.189664	0.275839		independent v
2		NF-kB	0.000250747	0.00067769		independent v
3		IkBa-NF-kB	0.0833327	0.055171		independent v
4		IkBb	0.0216351	0.00569699		independent v
5		IkBb-NF-kB	0.00830482	0.0025142		independent v
6		IkBe	0.0154363	0.00341416		independent v
7		IkBe-NF-kB	0.00592535	0.00116936		independent v
8		IKKIkBa	0	0.0146104		independent v
9		IKKIkBa-NF-kB	0	0.0060926		independent v
10		IKK	0.1	0.0128222		independent v
11		IKKIkBb	0	0.000151365		independent v
12		IKKIkBb-NF-kB	0	0.000168187		determined by
13		IKKIkBe	0	0.000104784		independent v
14		IKKIkBe-NF-kB	0	8.24376e-05		independent v
15		NF-kBn	0.000217022	0.0283283		independent v
16		IkBan	0.188598	0.00564503		independent v
17		IkBan-NF-kBn	0.00143106	0.00555004		independent v
18		IkBbn	0.0165048	6.24038e-05		independent v
19		IkBbn-NF-kBn	0.000314134	0.000155257		independent v
20		IkBen	0.0117759	3.73802e-05		independent v

Fig. 1. COPASI's interface for model editing: the left panel contains all of the possible actions organized in a hierarchy; the right panel changes according to the selection on the tree control. (**A**) kinetic functions: the mathematical formula of the kinetic rate laws is displayed together with the specifications of the kinetic parameters including their dimensions. (**B**) Parameter overview; listing all model parameters in a single table. including initial conditions and kinetic parameters. (**C**) Metabolite overview: where all molecules of the model are listed and their properties edited.

COPASI's graphical interface is similar to Windows Explorer in operation, where there is a vertical window on the left with a set of functions organized in a hierarchical way; on the right there is a larger window that contains all of the controls to operate the function selected on the left (Fig. 1). The major group of functions in the program are as follows:

- *Model*, where the model can be edited and viewed according to a biochemical or mathematical perspective.
- *Tasks*, consisting of the major numerical operations on the model: steady state, time course, stoichiometry, metabolic control analysis and Lyapunov exponents. Below each task an entry with results will appear after the task has been run.
- *Multiple Tasks*, which are operations repeating elementary tasks: parameter scanning, optimization and parameter estimation.
- *Output* is where plots and reports are defined and listed.
- *Functions* containing the mathematical functions available, such as the rate laws.

Model editing is done through tables and specialized widgets (Fig. 1) and the program provides various ways of editing the model items. For example, the user can change the value of kinetic constants in the single reaction widget (which provides detailed information about a single reaction), or in the parameter overview widget (Fig. 1B). There are two major views of the model: one set of widgets provides a view from the biochemical perspective, where the model is composed of reactions, compartments, metabolites, etc.; while another provides a mathematical view, where the model is composed of variables and differential equations. In the current version of COPASI the mathematical representation is for viewing only. The advantage of having two alternative views is that we expect different users to have different backgrounds and be more comfortable with one view or the other. This also provides a common software tool that may act as a translator of concepts for collaborators from different backgrounds.

Developing a model usually means combining information, e.g. kinetic data, from different sources. One possible source of errors in this process is the conversion of units of concentrations, kinetic parameters, etc. Although COPASI cannot do automatic conversion of such units, it provides help to the user in the following way: When a kinetic function is entered (or chosen from the integrated kinetics library) COPASI determines the units of the kinetic parameters from an analysis of the rate law, where possible. This allows the user to easily determine if available kinetic parameters from one source match a kinetic function from another source. To our knowledge, COPASI is currently the only software tool that is able to do so.

COPASI's native file format is based on XML and documentation of its schema is available so that other tools can write (or read) it. COPASI can also read Gepasi files, providing backwards compatibility with its predecessor. Finally and as explained below in more detail COPASI is able to import SBML either level 1 or level 2, and thus it can obtain models from many sources, such as other simulators, model databases, pathway databases and so on (see http://www.sbml.org). To store the complete model information, including task settings and output definitions, COPASI uses its own file format. Models can also be exported in SBML and the program can write the ordinary differential equations in plain C files (ready to be included in other C/C++ programs) and in Berkeley Madonna's format (http://www.berkeleymadonna.com), a popular program for nonlinear dynamics which does not import SBML.

COPASI can also output results of its various functions in two ways: report files and plots. The user can define report files containing any number of simulation results, parameters and other model items; this is done through an interface where all these items are organized in a hierarchy. In addition, the software has a number of predefined report formats that cover most of the common use cases. Results of operations are also presented directly in the user interface, in table format, which can be saved to tab-delimited files easily. Examples of such tables presented directly are time courses, the Jacobian matrix, matrices of control coefficients, etc. Tab-delimited files can also be saved directly from the plot window. Plotting support is built-in and plots, such as reports, can be defined in very flexible ways. COPASI supports *x*–*y* line plots and distribution histograms (a feature not commonly found in simulators, Fig. 2), scales can be linear or log-transformed, and the plot window allows zooming and panning.

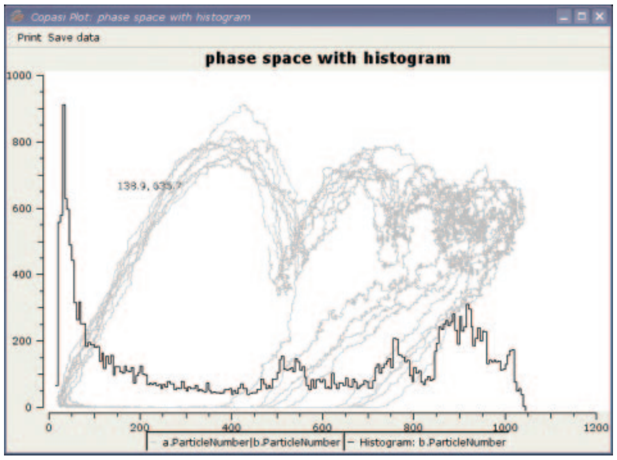

Copasi Plot: phase space with histogram
Print Save data
phase space with histogram
138.9, 635.7
a.ParticleNumber|b.ParticleNumber
Histogram: b.ParticleNumber

Fig. 2. Example of COPASI plotting capabilities, depicting a stochastic simulation of a model with oscillations. The light gray curve is the trajectory of the system in a 2D phase space projection. The black curve is a histogram of the distribution of the particle numbers of one of the species during the oscillation.

COPASI calculates time courses using a deterministic or a stochastic framework, depending on the user preference. For deterministic solutions, the LSODA integrator is used (Petzold, 1983), whereas for stochastic solutions the Gibson–Bruck version (Gibson and Bruck, 2000) of the Gillespie method (Gillespie, 1976) is applied (Fig. 2). In addition, a hybrid method that we developed is also available and described below. The user can easily switch between these methods by choosing from a drop-down list. COPASI can automatically convert chemical kinetic rate laws into their appropriate discrete stochastic equivalents, although this feature can be disabled when desired. Generally, for not too small particle numbers, the propensity of a reaction (the differential probability that a reaction event will happen in the next small time interval) is proportional to the reaction rate, which is given by the kinetic function of the reaction. This has been shown for mass action kinetics (Gillespie, 1976) and is at least approximately true under certain conditions for other kinetic types, such as Michaelis–Menten (see discussion below). For very small particle numbers, however, this relation does not necessarily hold. Consider, for example, a second-order mass action reaction ($2S \rightarrow P$). Its forward reaction rate is proportional to the square of the number of substrate particles. On the other hand the probability of a reaction event, calculated from the probability that two substrate particles meet (Gillespie, 1976), is proportional to $N_S*(N_S - 1)$, where N_S is the number of particles of the substrate. When presented with a model that was written for the deterministic simulation paradigm, COPASI can automatically do the conversion from N^2 to $N*(N-1)$, thus enabling a transparent switch from deterministic to stochastic simulation.

Another basic simulation function is the calculation of steady states, which is carried out by a combination of the damped Newton method and forward or backward integration (using LSODA). The steady state can also be characterized with linear stability analysis (Stucki, 1978) and metabolic control analysis (Fell, 1996).

COPASI determines structural (stoichiometric) properties of the biochemical network. Mass conservation is calculated using the algorithm described by Vallabhajosyula *et al*. (2006) that uses Householder reflections. (Vallabhajosyula *et al*. mentioned that COPASI used Gauss elimination for this purpose, which was true at that time; however, since then we have switched to using this more efficient algorithm.) Elementary flux modes, a unique set of the smallest possible sub-networks that still allow a steady state (Schuster *et al*., 1999) are calculated using our implementation of the METATOOL algorithm (Pfeiffer *et al*., 1999).

COPASI is equipped with a number of diverse optimization algorithms that can be used to minimize or maximize any variable of the model, following the scheme proposed by Mendes and Kell (1998). Two algorithms are based on estimating derivatives of the objective function, steepest descent and Levenberg–Marquardt (Levenberg, 1944; Marquardt, 1963; Goldfeld *et al*., 1966); a direct search algorithm, which is based on geometric concepts, the Hooke–Jeeves method (Hooke and Jeeves, 1961); four evolutionary algorithms, evolutionary programming (Fogel *et al*., 1992), genetic algorithm [a version with floating point encoding; Michalewicz (1994)], evolution strategy with stochastic ranking (SRES; Runarsson and Yao, 2000) and a genetic algorithm with stochastic ranking; finally, there is also a simple random search algorithm.

The optimization algorithms are also used for estimating parameter values that best fit a set of data provided by the user. To this end, COPASI reproduces the functionality of Gepasi (Mendes and Kell, 1998) and exceeds it by allowing mixtures of time course and steady-state data to be used simultaneously (Gepasi could only deal with one of these types of data at a time).

Finally, COPASI is able to compute the Lyapunov exponents and the divergence of a given system (as described below in more detail).

ARCHITECTURE

Our goals for COPASI are ease of use, availability of complex analysis methods and fast reliable simulation, which seem to be contradictory. In addition the software needs to be available for the majority of scientists, i.e. the main operating systems need to be supported. These requirements drove the architectural design.

Ease of use requires an interactive GUI. However, the size and complexity of some of the simulation tasks such as 'Parameter Estimation' requires speed and batch processing capabilities. To address both needs we chose C++ as the programming language and made a clear separation between the representation and the computing layer. This results in two separate versions of the program, one with a GUI and the other with a command line interface, that nevertheless share a common code base. The code itself is structured into smaller packages such as the model, the time course integration or the parameter estimation libraries to enhance maintainability.

The software is available for the following operating systems: Microsoft Windows (Windows 98 and above), Linux, Mac OS X (Power–PC, runs also on Intel) and Sun Microsystems Solaris (version 8 and above on SPARC). This cross platform portability has been achieved by adhering to the ANSI C++ standard and relying on toolkits and libraries that are available on the major operating systems.

The GUI was constructed based on the QT toolkit (Trolltech, Inc., Oslo), which is the essential portability layer (i.e. produces different OS versions from the same source code). This results in a program that has the expected look and feel of the underlying operating system. Since the users are provided with an interface following the same style and layout as other applications on their computer the usability is greatly enhanced. In addition support libraries for QT are available, which allow us to provide convenient plotting of results (Qwt project, http://qwt.sf.net) and rendering of mathematical expressions (QT Solutions, MML Widget).

For reliable and fast computation COPASI uses standard numerical libraries: LAPACK (Anderson *et al*., 1999) for linear algebra, BLAS (Lawson *et al*.,1979) for matrix and vector operations, and LSODA (Petzold, 1983) from ODEPACK (Hindmarsh, 1983) for integration of ODEs. There are optimized versions of LAPACK and BLAS for the major hardware and operating system that we have used to build the binary versions, providing COPASI with a little extra performance. For example, the Intel BLAS version uses special features of Pentium processors, whereas the Apple BLAS version uses the AltiVec unit of the PowerPC processors.

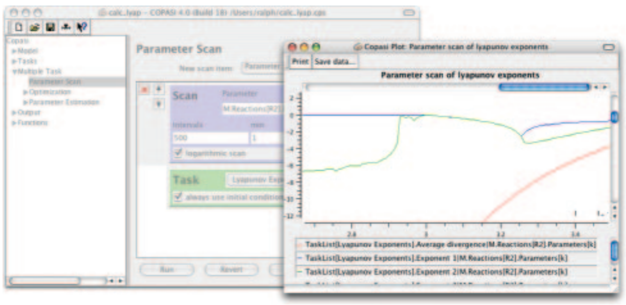

calc_lyap - COPASI 4.0 (Build 18) /Users/ralph/calc_lyap.cps
Copasi
Model
Tasks
Multiple Task
Parameter Scan
Optimization
Parameter Estimation
Output
Functions
Parameter Scan
New scan item:
Scan
Parameter
M.Reactions[R2]
Intervals
min
500
1
logarithmic scan
Task
Lyapunov Exp
always use initial conditions
Run
Revert
Copasi Plot: Parameter scan of lyapunov exponents
Print
Save data...
Parameter scan of lyapunov exponents
2
0
-2
-4
-6
-8
-10
-12
2.8
3
3.2
3.4
TaskList[Lyapunov Exponents].Average divergence|M.Reactions[R2].Parameters[k]
TaskList[Lyapunov Exponents].Exponent 1|M.Reactions[R2].Parameters[k]
TaskList[Lyapunov Exponents].Exponent 2|M.Reactions[R2].Parameters[k]

Fig. 3. Lyapunov exponents and average divergence as a function of a kinetic constant in an oscillating system. First and second Lyapunov exponents are the two upper curves, average divergence is the curve on the bottom right of the plot. This example was obtained through a parameter scan repeating the Lyapunov exponent calculation 500 times at different values of the kinetic constant.

SIMULATION ISSUES

Deciding on different simulation methods

Deciding which simulation methods are best suited for a specific reaction system is a tricky business. In principle, stochastic methods are more accurate because they take into account the intrinsic stochasticity of the system and do not rely on the assumption of continuous concentrations. However, stochastic methods are much slower than the deterministic ones based on solving ODEs. In addition, several stochastic trajectories, need to be calculated to determine a clear picture of the system's behavior; when stochastic effects are important, the concept of a single trajectory is replaced by a distribution of trajectories thus several single trajectory instances are needed to sample the distribution. The question is under which conditions it is valid to use an ODE representation. Hybrid algorithms that combine solution of ODEs with stochastic methods, as the one we describe below, solve this problem in special cases. These hybrid methods also require a decision algorithm to partition the system into components that can be simulated with ODEs and components that have to be simulated stochastically. So far, the stochastic–deterministic hybrid methods rely on some sort of heuristics (e.g. based on the number of particles or propensities) or more correctly on comparison of results obtained with the stochastic and the deterministic methods alone. Whereas the first method is error-prone, the second one does not provide any advantage over computing the complete system with the slower stochastic method alone. Thus there is a considerable need for good decision algorithms, particularly if they are based on sound theoretical arguments. So far, such an algorithm is not existing. However, as we have shown recently (Kummer *et al*., 2005), the computation of the divergence of the system (sum of Lyapunov exponents) is a factor that can support the user in making a decision. COPASI is capable of computing the divergence (Fig. 3). Since this is only one factor for the decision making which simulation method is suitable and it cannot be used automatically yet, it is not used in our hybrid algorithm as described below.

Calculation of the Lyapunov exponents is performed using the algorithm proposed by Wolf *et al*. (1985). One problem of this algorithm is that it requires a correct time interval for the orthonormalization of the linearized system. If the interval is too large the estimated value of the negative exponents will be much too high. If the interval is too small the calculation will be inefficient. COPASI provides the user with a criterion to judge the appropriateness of this parameter for the given model. A well-known property of Lyapunov exponents is that the sum of all exponents equals the average divergence of the system. On the other hand the divergence is defined as the trace of the Jacobian. Since COPASI can calculate both the Lyapunov exponents and the divergence of the system from the Jacobian, the user can compare these two values. We have found that a match of these two values constitutes a reliable indicator that the orthonormalization interval is not too large.

Although the absolute value of the divergence at one point in parameter space is not sufficient to decide for or against a simulation method, the computation of several values and the comparison of a stochastic and deterministic simulations at a point where the divergence of the specific system is at a maximum offers the following insight, valid for many systems: if both simulations coincide, the deterministic simulation should be reliable for all points in parameter space with lower divergence. If the solutions do not coincide, the computation at this point has to be carried out using stochastic or hybrid methods. Another reference point with lower divergence can then be taken for a subsequent analysis. We plan to provide a more automated analysis including more factors than just the divergence and a potential interpretation of the absolute values in future COPASI versions.

Hybrid algorithms

A number of hybrid methods have been proposed that combine different mathematical methodologies to simulate a biochemical system (Haseltine and Rawlings, 2002; Puchalka and Kierzek, 2004; Kiehl *et al*., 2004; Salis and Kaznessis, 2005; Salis *et al*., 2006; Alfonsi *et al*., 2005). They partition the reaction network into subnetworks and use appropriate stochastic, approximate stochastic or deterministic simulation methods on each of those subnets. In most cases the rationale behind this is to avoid the use of time-consuming stochastic methods on certain parts of the network, where they are not needed. Instead faster simulation methods are used on those subnets to accelerate the whole simulation.

Our hybrid method combines the stochastic simulation algorithm by Gibson and Bruck (Next Reaction Method; Gibson and Bruck,

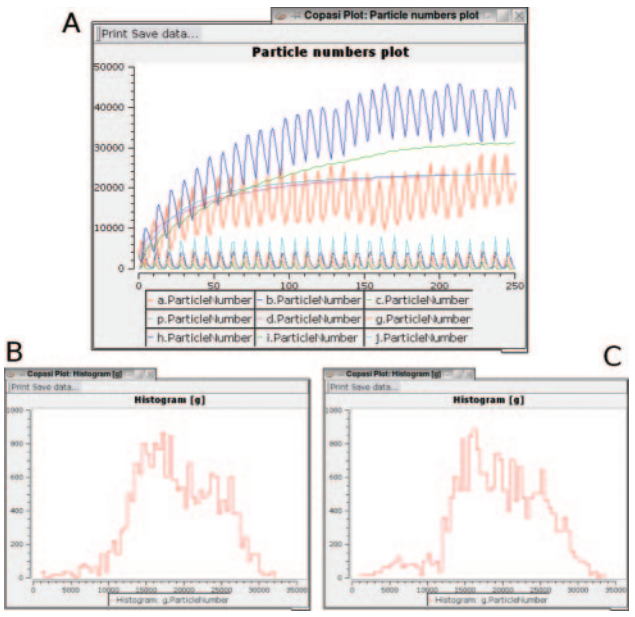


Fig. 4. Hybrid simulation of the calcium oscillation system in Kummer *et al.* (2000), comprising species a, b and c, coupled to a linear pathway of reactions (species d, g, h, i, j) via a calcium protein buffer complex p, which activates the reaction from g to h. All other steps in the linear pathway have Henri–Michaelis–Menten kinetics. Shown are the particle numbers over time (**A**) and a comparison of the hybrid (**B**) (lower and upper particle number limits 9 900 and 10 100, respectively) and pure stochastic (**C**) simulations in terms of the particle number histograms of species g.

2000) with different algorithms for the numerical integration of ODEs. The biochemical network is dynamically partitioned into a deterministic and a stochastic subnet depending on the current particle numbers in the system. The user can define limits for when a particle number should be considered low or high. The stochastic subnet contains reactions involving low numbered species as substrate or product. All the other reactions form the deterministic subnet. The two subnets are then simulated in parallel using the stochastic and deterministic solver, respectively (Fig. 4). The reaction probabilities in the stochastic subnet are approximated as constant between two stochastic reaction events.

Our hybrid method, which was developed and described in a diploma thesis of one of the authors (Pahle, 2002), is similar to the one by Haseltine and Rawlings (2002), though in fact they both were implemented independently during the same time. The main difference is the dynamic partitioning with user-defined limits for the particle numbers and the hysteresis-like repartitioning scheme.

This hybrid algorithm is able to simulate models faster than pure stochastic methods, while still taking into account the random effects in the stochastic subnetwork. If the limits for the particle numbers are set to zero, the whole network will be simulated deterministically. With increasing limits the calculation eventually converges to an exact stochastic simulation of the system. We tested our implementation in this limit successfully on the Discrete Stochastic Model Test Suite (http://www.calibayes.ncl.ac.uk/Resources/dsmts). If the particle number limits are in an intermediate range, between two repartitionings of the system the simulation proceeds similar to the method described and validated in Haseltine and Rawlings (2002). In addition, we compared our hybrid solver to an exact stochastic solver with respect to distributions of species particle numbers in different test systems. One example is shown in Fig. 4. However, we want to stress that due to the still heuristic partitioning criterion it is possible for the user to choose the limits such that the result of the hybrid solver deviates from the exact stochastic result as possible with other hybrid methods.

The dynamical partitioning is vital, e.g. for oscillating systems, but the speedup is very model-dependent. Because of the computational overhead for partitioning the system the hybrid method can occasionally take longer than pure stochastic methods. In addition, low-numbered species that take part in fast reactions slow the simulation down by forcing the fast reactions to be simulated stochastically. By using two distinct user-defined limits for the particle numbers and a hysteresis-like updating scheme for the partitioning we avoid unnecessary and time-consuming reaction swaps if particle numbers are fluctuating in a medium range.

We settled on the simple partitioning criterion using particle numbers for three reasons. First, the amplitude of relative fluctuations of particle numbers are high in low-numbered species. Single reaction events can have a significant impact here. Reactions involving those species should therefore be handled stochastically. Second, most of the computational effort of stochastic simulation algorithms is spent on fast reactions. In order to speed up the simulation, fast reactions should be taken out of the stochastic subsystem and simulated deterministically. The higher the number of substrate particles, the faster the reaction will be. This is true for mass action kinetics and for some parts of the phase space of enzyme kinetics. Third, if only reactions involving high-numbered species are simulated deterministically, the relative changes in particle numbers are minimal. For this reason, the change in reaction probabilities in the stochastic subnet caused by the fast subnet during one step can be neglected.

ODEs describing biochemical networks are often stiff. In our hybrid method we therefore use the LSODA algorithm (Petzold, 1983), which is adequate for the numerical integration of the deterministic subnetwork in the presence of stiffness. We also implemented a hybrid solver that uses a fourth-order Runge–Kutta method for cases when one is certain that the system is never stiff. Because the hybrid calculation requires many separate ODE integrations in small time intervals, the use of a simple one-step solver like Runge–Kutta can be faster since it lacks the computational overhead of predictor corrector methods.

There exist several mathematically equivalent algorithms for the stochastic simulation of biochemical networks [the Direct Method and First Reaction Method, Gillespie (1976); the Next Reaction Method, Gibson and Bruck (2000); and the method by Cao *et al.* (2004)]. In our hybrid solver we chose the method by Gibson and Bruck for the simulation of the stochastic part of the network, as it was the most convenient to integrate into our hybrid calculation scheme.

Handling enzyme kinetics in stochastic simulations

When stochastically simulating a reaction network, which has been described by a set of ODEs, all reaction rates have to be transferred to a corresponding reaction probability. This is rather simple and straightforward in the case of mass action kinetics (Gillespie, 1976). However, enzyme kinetic rate laws represent a lumping of terms each corresponding to an elementary mass action reaction; an important question is whether it is justifiable to use such a rate expression within stochastic simulations. Several authors (Rao and Arkin, 2003; Cao *et al*., 2005) have shown that as long as the initial assumptions for the assumed kinetics hold (e.g. excess substrate, fast reversible enzyme–substrate complex formation, etc.), it is indeed justifiable to assume the enzymatic reaction to constitute one single step with the respective rate law.

Basically, the rate law consists of a mass action part and a kinetic part (Hofmeyr, 1995). The kinetic part depends on reactant amounts and other factors, so it is not constant, but it could be assumed to freeze and become constant for the single reaction event that is computed in each step of the algorithm. This rate then has to be computed anew for the next step of the stochastic simulation.

Handling reversible reactions in stochastic simulations

In order to perform a stochastic simulation, reversible reactions have to be handled as a separate forward and backward irreversible reactions. In deterministic simulations forward and backward reaction rates can cancel each other out; in stochastic simulations each single reaction event has to be considered separately. COPASI provides a feature that, at the modeler's request, converts all reversible reactions to the corresponding individual forward and backward reactions in order to allow for a stochastic simulation of the model. The tool adjusts the reaction scheme and model description automatically. However, due to the difficulty in dissecting an arbitrary reversible kinetics into two irreversible kinetic functions, fully automatic conversion only works for mass action kinetics. For more complex kinetics the user will have to adjust the kinetics after the conversion.

Numerical considerations in stochastic simulations

The selection of a pseudo random number generator (PRNG) is commonly based only on its ability to create 'random' numbers. The major factors being its period and its capability to create unrelated numbers in higher dimensions. Additionally, the speed is also considered important as a large set of random numbers have to be chosen. COPASI uses the Mersenne Twister (Matsumoto and Nishimura, 1998) for all of the above reasons. A factor that is commonly overlooked in this context is that for large biochemical networks the numerical resolution or density of the numbers a PRNG can create is also an extremely important factor. It is trivial that one should use a PRNG implementation which is able to generate double versus float values in the interval (0,1) for probabilities. However, commonly this number is computed by creating an integer in $[0, (2^{32}-1)]$ and dividing it by $(2^{32}-1)$, i.e. we have a density of generated numbers of $1/(2^{32})$. That this density does not suffice to correctly simulate one slow reaction among a large number of fast ones can easily be seen. Let us consider the direct variant of the Gillespie algorithm and a system with 1000 fast reactions and one slow where the factor of slow to fast is 10^{-7}. This leads to a probability 10^{-10} for the slow reaction which is less than $1/(2^{32}-1)$, the maximal resolution for a random number generator that creates unsigned 32 bit integers; i.e. the slow reaction would never fire. The Mersenne Twister used by COPASI optionally provides pseudo random numbers with a density of $1/(2^{53}-1) < 1.2 \times 10^{-16}$ by drawing two random numbers. The latter limit is due to the floating point representation of a double and cannot be easily improved on today's architectures without major performance losses.

FLEXIBLE OUTPUT: HISTOGRAMS

COPASI contains a simple but flexible plotting tool. It can generate plots of all the numerical values that are generated in calculations. For example, one can plot time series of model variables (Fig. 4), phase space plots (Fig. 2), plots of model values versus model parameters or simulation results versus parameters of the numerical algorithms. In addition histograms of arbitrary values can be displayed. This is especially useful in the context of stochastic simulations where distributions of results are often of interest. With this feature it is easy to generate visualizations of stochastic steady-state distributions (Fig. 2) or the results of repeated simulations. All plots (including histograms) are updated during the calculations so the progress of the simulation can be monitored as it happens. Plot definition is very powerful providing the many model and simulation items through a detailed hierarchy; whereas, this provides much flexibility there are also predefined plots covering the most common diagrams, which can be generated through selection from a menu.

PARAMETER ESTIMATION: SIMULTANEOUS FIT TO TIME COURSE AND STEADY STATE

A special case of optimization is parameter estimation. The objective function in this case is given implicitly by a function that measures the distance between the model and the experimental data, such as a sum of squares of residuals. The parameters of the model (e.g. initial concentrations and kinetic constants) are then adjusted to minimize the objective function. COPASI uses the following weighted sum of squares as objective function which is minimized for a parameter set P.

$$S(P) = \sum_{i=1}^{n} \omega_i \left(x_i - y_i(P)\right)^2, \qquad (1)$$

where the $y_i(P)$ are the simulated data corresponding to the experimental data x_i. The sum is taken over all provided data points and ω_i is a weight to make all trajectories of each variable have similar importance in the fit. Currently COPASI calculates weights as $\omega_i = 1/\sigma_i^2$, with σ_i^2 is the variance of the trajectory containing x_i (calculated independently for each item being fit). In future versions of the software there will be further choices, such as $\omega_i = 1/\left|\langle x_i \rangle\right|$ and $\omega_i = 1/\sqrt{\langle x_i^2 \rangle}$, where $\langle x_i \rangle$ is the mean value of the points in a trajectory (also calculated independently for each item being fit). Users will also be able to override the value of any weight manually.

The software is able to simultaneously fit the model to data from steady-state and time course experiments. This is achieved by enabling the user to provide multiple data files with multiple experiments. Each experiment must provide independent input data and dependent data. The sum in Equation (1) is taken over all dependent data points x_i and the corresponding y_i are calculated with respect to the given independent data.

The minimization of Equation (1) can be achieved with generic optimization algorithms, but some specific algorithms, such as Levenberg–Marquardt (Levenberg, 1944; Marquardt, 1963), make use of the special form of the objective function $S(P)$. COPASI distinguishes those special algorithms and provides them with the vector of residuals rather than just the weighted sum of squares, as is the case for the generic algorithms. In particular, COPASI includes two versions of the Levenberg–Marquardt algorithm, the original one from Marquardt (1963) that specifically uses the vector of residuals, and a version that uses only the sum of squares (Goldfeld *et al*., 1966). The selection of one or the other version of this algorithm is done automatically and the fact that there are two versions is transparent to the user.

PARAMETER SCANS FEATURING REPEAT AND RANDOM GRIDS

COPASI can perform complex calculations that combine several simulation runs. A graphical interface is provided that allows easy access to the following features: parameter scans where a simulation is run several times, each one with a different value of the scanned parameters; repeated simulations, where a simulation is repeated without any change in parameters, useful for delineating distributions of trajectories in stochastic simulation; and random parameter sampling, where parameter values are drawn from a defined distribution, performing Monte Carlo parameter scans. These features can also be nested arbitrarily, for example, it is possible to define a calculation where for each of 10 values of a kinetic parameter a stochastic simulation is repeated 100 times. An example of parameter scans is depicted in Figure 3.

COMMUNITY INTEGRATION

In order to be compatible with other systems biology software, COPASI reads and writes SBML files through libsbml (http://www.sbml.org/software/libsbml/). All valid SBML files can be read, although the user is warned when the model contains features of SBML that are not yet supported (though only a few SBML features are not handled: rules and events). To assure conformance with the standard, our software is regularly tested with the SBML semantic test-suite (http://www.sbml.org/wiki/Semantic_Test_Suite) which consists of a battery of SBML files and expected results to test each of the individual SBML features. COPASI passes all tests except those for the features currently not supported.

In addition to SBML support, COPASI is also able to export the kinetic model as simple C code that can be incorporated in C/C++ programs, as well as in tools such as Mathematica, etc. The user has also the possibility of exporting the model as a Berkeley Madonna file (http://www.berkeleymadonna.com), which is a popular program to simulate biochemical systems, but has no means to import SBML.

CONCLUSION

We have presented COPASI, a software tool that integrates diverse numerical methods used in computational systems biology. One of its important novel features is the way in which it facilitates a modeler to easily switch between different simulation approaches. We discussed in more detail the specific problems that such feature implies and a number of numerical procedures that have been implemented to address them. Additional unique features of COPASI that are helpful in studying biochemical networks were described, such as flexible parameter scans, optimization of arbitrary expressions and parameter estimation using time course and steady-state data simultaneously. In conclusion, COPASI is a powerful yet user-friendly tool that provides common as well as unique features for systems biology research.

ACKNOWLEDGEMENTS

The following people contributed significantly to COPASI by programming, testing and useful discussions (in alphabetical order): Jens Christian Brasen, Diogo Camacho, Carel van Gend, Ankur Gupta, Yonqun He, Wenjun Hu, Katrin Hübner, Adaoha Ihekwaba, Aejaaz Kamal, Mrinmyee Kulkarni, Ann Zahle Larsen, Sarah Lilienthal, Rohan Luktuke, Bharat Mehrotra, Femke Mensonides, Saroj Mohapatra, Ursula Rost, Gaurav Singh, Anurag Srivastava, Iulian Stoleriu, Wei Sun, Irina Surovtsova, Sameer Tupe, Ralph Voigt, Andreas Weidemann, Katja Wegner, Jürgen Zobeley. We thank the BMBF, the Klaus Tschira Foundation and the Virginia Bioinformatics Institute for generous funding of this project.

Conflict of Interest: While COPASI is free for non-commercial use, a portion of the revenue from possible COPASI commercial licences may be awarded to the authors by their institutions.

REFERENCES

Alfonsi,A. *et al*. (2005) Adaptive simulation of hybrid stochastic and deterministic models for biochemical systems. *ESAIM Proc*., **14**, 1–13.

Anderson,E., Bai,Z., Bischof,C., Blackford,S., Demmel,J., Dongarra,J., Du Croz,J., Greenbaum,A., Hammarling,S., McKenney,A. and Sorensen,D. (1999) *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Cao,Y. *et al*. (2004) Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *J. Chem. Phys*., **121**, 4059–4067.

Cao,Y. *et al*. (2005) Multiscale stochastic simulation algorithm with stochastic partial equlibrium assumption for chemically reacting systems. *J. Comp. Phys*., **206**, 395–411.

Fell,D. (1996) *Understanding the Control of Metabolism*. Portland Press, London.

Fogel,D.B. *et al*. (1992) Meta-evolutionary programming. In Chen,R.R. (eds.), *25th Asilomar Conference on Signals, Systems & Computers*. IEEE Computer Society, Asilomar, pp. 540–545.

Gibson,M.A. and Bruck,J. (2000) Efficient exact stochastic simulation of chemical systems with many species and many channels. *J. Phys. Chem*., **A104**, 1876–1889.

Gillespie,D.T. (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comp. Phys*., **22**, 402–434.

Goldfeld,S.M. *et al*. (1966) Maximisation by quadratic hill-climbing. *Econometrica*, **34**, 541–555.

Haseltine,E.L. and Rawlings,J.B. (2002) Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. *J. Chem. Phys*., **117**, 6959–6969.

Heinrich,R. and Schuster,S. (1996) *The Regulation of Cellular Systems*. Champman & Hall. International Thomson Publishing.

Hindmarsh,A.C. (1983) ODEPACK, a systematized collection of ODE solvers. In Stepleman,R.S. (ed.), *Scientific Computing*. North-Holland, Amsterdam, pp. 55–64.

Hofmeyr,J.H. (1995) Metabolic regulation: a control analytic perspective. *J. Bioenerg. Biomembr*., **27**, 479–490.

Hooke,R. and Jeeves,T.A. (1961) Direct search solution of numerical and statistical problems. *J. ACM*, **8**, 212–229.

Hucka,M. *et al*. (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, **19**, 524–531.

Kiehl,T.R. *et al*. (2004) Hybrid simulation of cellular behavior. *Bioinformatics*, **20**, 316–322.

Klamt,S. *et al*. (2003) FluxAnalyzer: exploring structure, pathways, and flux distributions in metabolic networks on interactive flux maps. *Bioinformatics*, **19**, 261–269.

Kummer,U. *et al*. (2000) Switching from simple to complex oscillations in calcium signaling. *Biophys. J*., **79**, 1188–1195.

Kummer,U. *et al*. (2005) Transition from stochastic to deterministic behavior in calcium oscillations. *Biophys. J*., **89**, 1603–1611.

Lawson,L. *et al*. (1979) Basic linear algebra subprograms for FORTRAN usage. *ACM Trans. Math. Soft*., **5**, 308–323.

Le Novère,N. and Shimizu,T.S. (2001) StochSim: modelling of stochastic biomolecular processes. *Bioinformatics*, **17**, 575–576.

Levenberg,K. (1944) A method for the solution of certain nonlinear problems in least squares. *Quart. Appl. Math*., **2**, 164–168.

Lloyd,C.M. *et al*. (2004) CellML: its future, present and past. *Prog. Biophys. Mol. Biol*., **85**, 433–450.

Marquardt,D. (1963) An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math*., **11**, 431–441.

Matsumoto,M. and Nishimura,T. (1998) Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Model. Comput. Simul*., **8**, 3–30.

Mendes,P. (1993) GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems. *Comput. Appl. Biosci*., **9**, 563–571.

Mendes,P. (1997) Biochemistry by numbers: simulation of biochemical pathways with Gepasi 3. *Trends Biochem. Sci*., **22**, 361–363.

Mendes,P. and Kell,D.B. (1998) Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. *Bioinformatics*, **14**, 869–883.

Meng,T.C. *et al*. (2004) Cellware: the grid-enabled tool for cell modeling and simulation. *Bioinformatics*, **20**, 1319–1321.

Michalewicz,Z. (1994) *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin.

Pahle,J. (2002) Eine Hybridmethode zur Simulation biochemischer Prozesse (in German). *Diploma thesis Universität Karlsruhe (TH), Germany*.

Petzold,L. (1983) Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. *SIAM J. Sci. Stat. Comput*., **4**, 136–148.

Pfeiffer,T. *et al*. (1999) METATOOL: for studying metabolic networks. *Bioinformatics*, **15**, 251–257.

Puchalka,J. and Kierzek,A.M. (2004) Bridging the gap between stochastic and deterministic regimes in the kinetic simulations of the biochemical reaction networks. *Biophys. J*., **86**, 1357–1372.

Rao,C.V. and Arkin,A.P. (2003) Stochastic chemical kinetics and the quasi-steady-state assumption: application to the gillespie algorithm. *J. Chem. Phys*., **118**, 4999–5010.

Runarsson,T. and Yao,X. (2000) Stochastic ranking for constrained evolutionary optimization. *IEEE Trans. Evol. Comput*., **4**, 284–294.

Salis,H. and Kaznessis,Y. (2005) Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions. *J. Chem. Phys*., **122**, 054103.

Salis,H. *et al*. (2006) Multiscale Hy3S: hybrid stochastic simulation for supercomputers. *BMC Bioinformatics*, **7**, 93.

Sauro,H.M. *et al*. (2003) Next generation simulation tools: the Systems Biology Workbench and BioSPICE integration. *OMICS*, **7**, 355–372.

Schuster,S. *et al*. (1999) Detection of elementary flux modes in biochemical networks: a promising tool for pathway analysis and metabolic engineering. *Trends Biotechnol*., **17**, 53–60.

Stucki,J.W. (1978) Stability analysis of biochemical systems: a practical guide. *Progr. Biophys. Mol. Biol*., **33**, 99–187.

Tomita,M. *et al*. (1999) E-Cell: software environment for whole cell simulation. *Bioinformatics*, **15**, 72–84.

Vallabhajosyula,R.R. *et al*. (2006) Conservation analysis of large biochemical networks. *Bioinformatics*, **22**, 346–353.

Wolf,A. *et al*. (1985) Determining Lypunov exponents from a time series. *Physica*, **16D**, 285–317.