BioTechnologia vol. 92(3) • pp. 265-280 • 2011
Journal of Biotechnology, Computational Biology and Bionanotechnology REVIEW PAPER

Deterministic models and stochastic simulations in multiple reaction models in systems biology

PAWEŁ LACHOR [1]*, KRZYSZTOF PUSZYŃSKI [2], ANDRZEJ POLAŃSKI [1]

[1] Institute of Informatics, Silesian University of Technology, Gliwice, Poland

[2] Institute of Automatic Control, Silesian University of Technology, Gliwice, Poland

* Corresponding author: pawel.lachor@polsl.pl

Abstract

In this paper, we have overviewed deterministic and stochastic approaches for the modeling of bio-molecular reactions in systems biology. We have described and compared different versions of stochastic simulation approaches towards the modeling of bio-molecular reaction systems, the direct approach and the family of the tau-leap methods. We also have illustrated differences between different approaches by providing numerical examples of computational analyses for selected models. Computational examples of application of stochastic simulation algorithms involved two systems of different interacting bio-molecular species and one model from the area of ecology, describing interactions between two animal populations. Apart from the direct approach and tau-leap family of methods, we have also overviewed the so called hybrid algorithm of stochastic simulations, proven to be very efficient when there are huge differences between reaction rates in the system.

Key words: stochastic simulation algorithms, direct algorithm, Tau-Leap approximated stochastic simulation algorithms, hybrid algorithm

Introduction

Models in systems biology, describing the dynamics of interactions between bio-molecules, help to explain and understand many aspects of biological processes occurring at the molecular level in cells of living organisms. There are many classes of models, such as signaling pathways, gene regulation networks, transcription networks, metabolic networks, protein interaction networks, etc., which describe dynamics of evolution and interactions of populations of bio-molecular species. Numerous examples for models of bio-molecular systems have previously been discussed in the literature (e.g. Hat et al., 2009).

Dynamics in models of evolution of bio-molecule populations are often described by using the deterministic – limit approach based on systems of coupled first-order ordinary differential equations (ODEs) (Hat et al., 2009; Krishna et al., 2006). Deterministic approach is based on the hypothesis that population sizes of interacting bio-molecules are large and as a consequence state vectors of bio-molecular models can be defined by average population size of bio-molecular species (Butcher, 2003). There are very efficient methods for numerically solving the systems of differential equations (Butcher, 2003). Deterministic models are therefore very easy to use. They also provide unique insights into bio-molecular interactions and help explain many mechanisms of evolution in systems biology models.

However, it is well known that there are situations where limitation of the deterministic approach can hamper capturing important properties of the system under study. Most important limitations of deterministic approach are as follows.

1) Deterministic approach although efficient in terms of computational load, is not accurate for systems that contain low-rate reactions. These are most often related to species occurring in small molecular quantities. If a system contains bio-molecular species of low cellular concentration (quantity) then representing this species in molecular reactions by its population mean can introduce either significant errors in model predictions or can even change the model's qualitative properties.

2) When the behavior of a bio-molecular system is studied for parameter ranges close to bifurcation points (which correspond to qualitative changes of systems dynamics) then, as mentioned above, limiting to average values

can cause overlooking important features of the system dynamics. Also for systems whose evolution significantly depends on (distribution) of initial conditions, such as bi stable or multi stable systems, deterministic modeling may not adequately describe the distributions of system responses.

3) Deterministic approach ignores stochastic variation of time responses of the system under study. Investigating stochastic variation of system trajectories is on one hand interesting by itself. On the other hand it is well known that stochasticity in systems biology models has very important biological/evolutionary meaning (Geva-Zatorsky et al., 2006) which provides another motivation for its study.

Stochastic simulation approach is applied in order to study effects of random fluctuations in numbers of molecular species in systems biology models. Stochastic simulations algorithm (SSA) for interactions of bio-molecules was first proposed by Gillespie (Gillespie, 1976) for simple models of chemical reactions. Gillespie's idea of stochastic simulation is to record changes in bio-molecular species sizes vector over time, following from subsequent occurrences of different reactions. In each step of the simulations, random draw of the reaction to occur next is done on the basis of the value of reaction propensities. Gillespie's paper led to the generation of a number of subsequent literatures (e.g. Chatterjee et al., 2005; Cao et al., 2006). The original idea was developed by improving its computational efficiency and by fitting the structure of the algorithm to the specific properties of the analyzed system.

This paper is aimed as a survey devoted to (i) description of deterministic and stochastic approach to modeling of bio-molecular systems, (ii) comparison between deterministic and stochastic approach, (iii) comparison and applicability study of different versions of stochastic simulations approach to modeling of bio-molecular systems. Survey papers devoted to variants of Gillespie algorithm have already appeared in the literature (Meng et al., 2004; Pineda-Krch, 2008). When compared to the existing literature surveys, our paper is more general in the following sense: (i) we illustrate differences between different approaches by providing numerical examples of computational analyses for selected models; (ii) we include wider scope of variants of the Gillespie algorithm, namely apart from direct approach and tau-leap family of methods, we also overview and present computational example for the so called hybrid algorithm of stochastic simulations (Haseltine and Rawlings, 2002), proven to be very efficient in situations where propensities of reactions in the studied system are of different orders of magnitude.

The paper is structured as follows. In the section *Models of interactions of bio-molecular species* we present basic principles of bio-molecular interactions modeling. In the section *Deterministic and stochastic simulation algorithms*, overview of deterministic and stochastic methods in systems biology is presented. In the section *Computational complexity of stochastic simulation algorithms* we discuss problem of computational complexity of stochastic simulation algorithms. In the section *Computational examples* we present exemplary results of applications of these methods, which provide a base for quantitative comparisons of different approaches. Finally, in the Conclusion section, conclusions following from the whole study are summarized.

Models of interactions of bio-molecular species

Before we start a discussion on different approaches in simulating techniques, we introduce the notations to be used in subsequent sections. We denote the reaction that can occur between biomolecules by R_j where j is the reaction number. The total number of reactions in the model is denoted by M. Reactions between biomolecules can lead to the disappearance of some types of biomolecules and creation of others.

The history of occurrences of all reactions is summarized by actual numbers of all biomolecules in the system or their molar concentrations, called population state vector. Population state vector is denoted by $x(t) = (x_1(t), ..., x_N(t))$ where N is total number of bio-molecular species.

Each reaction is characterized by its propensity coefficient, which defines the probability of occurrence of the reaction. Propensity for each reaction R_j is denoted by $a_j(x)$ and the sum of all propensities by $a_0(x)$. Each reaction R_j is also characterized by a vector $v_j = (v_{1j}, ..., v_{Nj})$, called state change vector, whose component v_{ij} denotes the change in the number molecules of the species introduced by reaction R_j.

Reaction rate, Hill model, inhibition, stimulation

The law of mass action (Waage and Guldberg, 1864) derived by Gulberg and Waage is a basic mathematical

model of reactions in systems biology. According to this law, the probability of each reaction event is proportional to the product of the concentration of participating reactants. In the deterministic limit, rate of each elementary reaction is proportional to the product of concentrations of the participating reactants. From the law of mass action it follows that the increase in concentration of the reagents will result in an increase of the reaction rate, while the decrease must result in the slowdown. If a reagent is "drained out", the reaction does not occur apt all. Let us show an example of a reaction of two reagents, A and B, which form a complex AB

$$A + B \rightarrow AB \tag{1}$$

In accordance to the law of mass action, the rate of this reaction should be expressed as a product of concentrations of reactants, denoted by $[A]$ and $[B]$, and the propensity coefficient p,

$$r = p \times [A] \times [B] \tag{2}$$

Rates of all reactions contribute to balances of all masses, which in the simple example stated above has the form

$$\frac{d[AB]}{dt} = r = p \times [A] \times [B] \tag{3}$$

The product structure given by the above-mentioned law of mass action is additionally modified by assuming the possible character of the influence of a bio-molecule on the reaction rate, stimulation or inhibition. Different types of influence can be modeled by Hill equation with appropriate parameters, given below,

$$r(A) = \frac{[A]}{dt} \frac{A^n}{k^n + A^n} \tag{4}$$

Hill equation model contains two parameters. The parameter k is interpreted as a dissociation constant or "half occupation" constant. Depending on the value of the parameter n in the exponent, called the Hill coefficient, Hill equation can be a model of positive cooperativity (stimulation effect) of the reactant A in the reaction, when n takes positive values, $n > 0$, or negative cooperativity (inhibition effect), in the opposite case, where $n < 0$.

Differential equation models and lists of reactions models, SBML

Dynamics in systems biology models can be described by using different methods. The most common approach is based on writing systems of coupled first order differential equations describing concentrations for each single species involved in the model by a separate equation. Such a method allows the researcher to present even the considerably complex systems in a brief form. Constructing model with ODEs is usually associated with a deterministic solution, although there are complex tools allowing the calculation of stochastic trajectories in such models.

Across the literature, we may find many examples of models of bio-molecular systems written in the form of sets of differential equations. For illustration, we will focus on the simple model described in (Hat et al., 2009). The simple "Toy" model introduced in this paper describes $p53 | Mdm2$ regulatory core of interactions of two proteins, $p53$ and $Mdm2$. This model was also used in further chapters for computational illustration of the problem of efficiency of different stochastic approaches for simulation in systems biology models. The "Toy" model consists of only three components (species), total $p53$, cytoplasmic $Mdm2$ ($Mdm2c$) and nuclear $Mdm2$ ($Mdm2n$). The model (Hat et al., 2009) was described using ODEs. Three appropriate differential equations were derived, each describing concentration over time of one of species involved in the model ($p53$, $Mdm2c$ and $Mdm2n$). These equations are presented below,

$$\frac{d}{dt} p53 = m \cdot s_1 - k_{d1} \cdot p53 \cdot (Mdm2n)^2 \tag{5}$$

$$\frac{d}{dt} Mdm2c = n \cdot \left(s_2 + s_3 \cdot \frac{(p53)^3}{s_4 + (p53)^3} \right) - k_1 \cdot \frac{k_2}{k_2 + p53} \cdot Mdm \tag{6}$$

$$\frac{d}{dt} Mdm2n = k_1 \frac{k_2}{k_2 + p53} \cdot Mdm2c - k_{d2} \cdot Mdm2n \tag{7}$$

The above-mentioned model describes balances of three interacting proteins and includes reaction rate laws provided by the law of mass action and Hill models with different types of interactions.

Another method to describe models in system biology is based on listing all reactions and for each reaction defining appropriate rate laws. Listing all reactions in the system is basically equivalent to writing system's equations for balances of volumes of bio-molecular species. For the case of the "Toy" model of $p53$ and $Mdm2$ interactions (Hat et al., 2009), the "list of reactions" model will assume the following form,

$$\varnothing \rightarrow p53, \qquad r = m \cdot s_1 \tag{8}$$

$$p53 \rightarrow \varnothing, \qquad r = k_{d1} \cdot [p53] \cdot [Mdm2n]^2 \tag{9}$$

$$\varnothing \rightarrow Mdm2c, \qquad r = n \cdot s_2 \tag{10}$$

$$\varnothing \leftarrow Mdm2c, \qquad r = n \cdot s_3 \cdot \frac{[p53]^3}{s_4 + [p53]^3} \tag{11}$$

$$Mdm2c \rightarrow Mdm2n, \qquad r = k_1 \cdot \frac{k_2}{k_2 + [p53]} \cdot [Mdm2c] \tag{12}$$

$$Mdm2n \rightarrow \varnothing, \qquad r = k_{d2} \cdot [Mdm2n] \tag{13}$$

In the context of the "list of reactions" models, it is worthwhile to mention the System Biology Markup Language (SBML) (http://sbml.org), a standard designed to ease biological data exchange over different systems and tools. SBML standard is based on representing list of reactions of the type (8)-(13) as .xml files and allows for fast and efficient exchange of biological data and models between a variety of software environments. Formulating a model, like reactions (8)-(13), in the SBML format does not require any advanced knowledge of the XML schema of the systems uses. There are interactive tools that help creating SBML models (Swainston and Mendes, 2009).

Graphical representation

The above-described methods, while good when preparing simulation experiments, may not be comprehensive enough to present/understand the mechanisms of the system, for a researcher. A valuable help in understanding system's functionalities is the use of graphical representation. Below, in Figure 1, a diagram representing interactions in the $p53 \mid Mdm2$ regulatory mechanism, modeled by equations (5)-(7), or alternatively by the list of reactions (8)-(13), is presented. This diagram illustrates conventions commonly applied in graphical representations of system biology models.

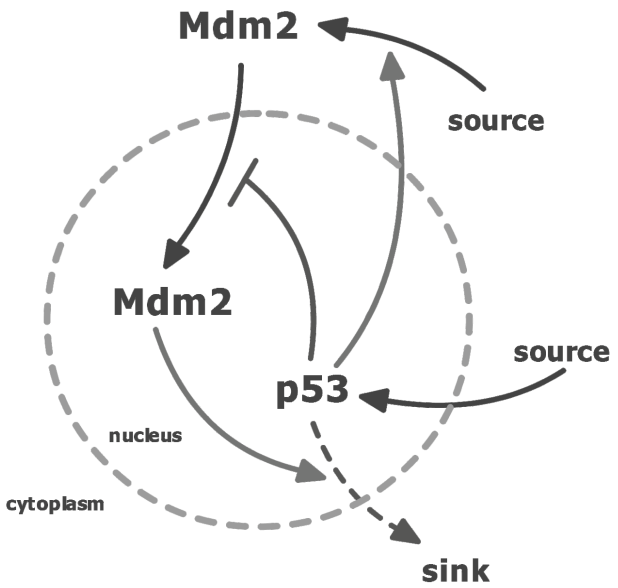

Fig. 1. Diagram of $p53 \mid Mdm2$ regulatory core

From Figure 1, one can observe that in the studied system we can find two interlinked feedback loops, positive and negative. Feedback structure includes $p53$, in the nucleus, blocking (inhibiting) nuclear import of $Mdm2$ which in turn enhances (stimulates) $p53$ degradation in the nucleus. Additionally, $p53$ stimulates production of $Mdm2$ in the cytoplasm.

Deterministic and stochastic simulation algorithms

In this section, algorithms for solving/simulating models of interacting bio-molecules are presented. As aforementioned, these algorithms can be generally divided into two categories, deterministic and stochastic.

Deterministic models

Deterministic methods for simulating the behavior of a systems biology model involve obtaining numerical solution to ODEs. Well-developed methods for integrating ODEs, which are based on recursive procedures with adaptively adjusted step size, e.g., Runge-Kutta procedures, are fast, reliable and applicable even to systems of high order.

When a system is modeled by a set of ODEs, insights to its properties can be gained by a study based on the system theory computational tools including computing

equilibrium points, verifying their stabilities, verifying existence of possible cycles (limit cycles), describing types of transient processes in the system, checking the dependence of the behavior of solutions on parameters, bifurcation points and their types.

Using the above described analytical as semi-analytical tools is, however, substantially limited or impossible in the context of stochastic modeling. The limitations follow from the fact that in the stochastic case deterministic trajectories are replaced by families of probability distributions. Researching these probability distributions by stochastic sampling techniques was presented in the next sections.

Stochastic models

In the deterministic modeling case, there is not much room for options regarding the quality of simulations, since in the widely used software packages these options are often set automatically. In contrast, in the stochastic case, there are several further decisions for a researcher to make, regarding the structure of the algorithm and the related accuracy of simulations. These decisions highly depend on the size of the researched system. Important problem to solve is often the compromise between the time load of the computational project and the accuracy of the expected results. Some of the problems related to stochastic modeling as well as applicable approaches were also described in the subsections below.

In 1976, Gillespie published his first paper (Gillespie, 1976) that described an algorithm for simulating chemical (biochemical) reaction. Purpose of the stochastic simulation algorithm proposed by him was to record changes in the population vector over time. During each step of the Gillespie simulation procedure, we need to decide when and which reaction will occur as the soonest. Over the time many variants of original algorithm were proposed in an effort to improve computational efficiency while trying to preserve the exactness.

Exact stochastic simulation algorithms

Exact stochastic methods are valid for small numbers of bio-molecules in the system, as they account for each reaction event in the system separately. Originally, Gillespie (1976) proposed two methods for exact simulation, First Reaction Method and the Direct Method. Additionally, we will describe the Next Reaction Method proposed by Gibson and Bruck (2000) as an improved version of the First Reaction Method. When considering the exact solution, we need to take into account that it may slow down the calculation process significantly, especially for large systems.

First Reaction Method

Initialization. In the first reaction method, we start by initializing the number of reactions R_j, $j = 1, ..., M$, (where M is the total number of reactions) and concentration of species in population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at initial time.

In **step 1** of the first reaction method, for each reaction R_j we draw random numbers r_j from the uniform distribution and calculate the propensity functions $a_j(x)$.

Step 2. Based on the randomly drawn numbers r_j, we compute the "putative" reaction time t_j for each reaction R_j, using the following formula:

$$t_j = -\frac{1}{a_j(x)} \cdot \ln(r_j) \qquad j = 1, \ldots, M \tag{14}$$

A reaction which eventually occurs in the simulated system is the reaction R_μ, whose index is defined by

$$t_\mu = \min(t_1, \ldots, t_m) \tag{15}$$

Step 3. For the given R_μ we proceed to recording the time of this reaction by adding the calculated time step to the current time counter.

$$t = t + t_\mu \tag{16}$$

Next, we proceed to implementing the changes to the population state vector $x(t)$ introduced by occurrence of the selected reaction.

$$x = x + v_\mu \tag{17}$$

If the current time t of experiment does not exceed the total time of simulation experiment, we proceed to the step 1 and follow the subsequent steps of the algorithm, otherwise we terminate the procedure.

The First Reaction Method is an exact method for computing stochastic realizations related to system's evolution. However, we need to take into consideration the fact that in each iteration, we must draw M random numbers. In addition, we calculate M times the logarithm to compute the putative reaction times. Therefore, for larger systems consisting of various reactions and species, decision of using the First Reaction Method can

result in a significant slowdown of the computational time.

Direct Method

In the Direct Method, proposed by Gillespie, the number of necessary draws of random numbers is reduced. This algorithm can be divided into four major steps.

Initialization. We initialize reactions R_j, $j = 1, ..., M$ (where M is the total number of reactions) and concentrations of species in population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at initial time.

Step 1. We draw two random numbers r_1 and r_2 from the uniform distribution. Then we calculate the propensity functions $a_j(x)$ for each reaction R_j. After that we sum all propensities and we denote the sum by $a_0(x)$.

$$a_0(x) = \sum a_j(x) \tag{18}$$

Step 2. In this step, we determine which R_j will occur next and the time instant of its occurrence. Using the sum of propensities $a_0(x)$ we calculate the time step T.

$$T = -\frac{1}{a_0(x)} \cdot \ln(r_1) \tag{19}$$

The smallest integer j satisfying

$$\left(\sum_{i=1}^{j} a_i(x) \right) > r_2 a_0(x)$$

is the index of the next reaction to be executed.

Step 3. In last step, we compute the time of the next reaction occurrence, $t = t + T$ and execute the selected reaction R_j by modifying the population state vector $x(t)$ by adding the vector of state change v_j for the reaction R_j.

$$x = x + v_j \tag{20}$$

If the current time of experiment does not exceed the total time, we go to the step 1 and follow the subsequent steps of the algorithm, otherwise the algorithm terminates.

The drawback of the Direct Method is that when we consider a system consisting of (very) rare and frequent reactions at the same time, the Direct Method, due to finite precision of random number generators, may lead to "overlooking" of all (or most) of rare events. This problem does not appear in the First Reaction Method which is, however, computationally less efficient as already mentioned.

Next Reaction Method

Gibson and Bruck (Gibson and Bruck, 2000) introduced an adaptation of the First Reaction Method improved in terms of its computational complexity. Improvements were achieved through the re-use of already calculated propensities. The number of draws of random numbers per step was reduced to one. The idea is based on introduction of the priority queue containing the sorted putative reaction times. Instead of using the relative putative time, in this case we use their absolute values.

Initialization. Analogous to the previously discussed methods, we initialize reactions R_j, $j = 1, ..., M$ (where M is the total number of reactions) and concentrations of species in the population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at the initial time. Next, the priority queue is created. In our case, we assume that the queue has the form of the ascending sorted list. The list includes the pop method which simultaneously removes and returns its first element. Next, for each reaction R_j we calculate the propensity functions $a_j(x)$ using the random number r_j drawn from the uniform distribution, we determine the reaction time t_j by using the formula,

$$t_j = -\frac{1}{a_j(x)} \cdot \ln(r_j) + t_0 \quad j = 1, \ldots, M \tag{21}$$

Finally, we add it to the priority queue. After this step, the initialization is over and we can proceed to Step 1.

Step 1. The first element, t_μ, from the priority queue indicates the time at which the first reaction, R_μ, occurs. We pop this element from the priority queue, proceed to time $t = t_\mu$ and we implement changes to the population state vector introduced by this reaction.

$$x = x + v_\mu \tag{22}$$

Step 2. In this step, we focus our attention on the reactions whose propensities were affected due to the execution of the Step 1. For each reaction, R_i, from the affected pool we:

- compute the new propensity and store it in a temporary container a_{temp},
- re-calculate the reaction time and permit update on the priority queue,
- store the new value of propensity $a_i(x) = a_{temp}$

$$t_i = \left(\frac{a_i(x)}{a_{temp}} \right) \cdot \left(t_i - t \right) + t \tag{23}$$

Once all affected reactions have been updated, we move to Step 3 and proceed by calculating a new reaction time for the reaction R_μ. To do so, similar to Initialization step, we draw a random number r_μ from the uniform distribution and compute new occurrence time of the reaction R_μ as the exponentially distributed random number with mean $a_\mu(x)$,

$$t_\mu = \frac{1}{a_\mu(x)} \cdot \ln(r_\mu) + t \tag{24}$$

We store the new reaction time t_μ by pushing it to the priority queue.

If the current time of experiment does not exceed the total time, we proceed to the step 1 and follow the subsequent steps of the algorithm, otherwise we terminate.

The above-stated algorithm combines advantages of the two previous ones. On one hand, it does not overlook rare reactions, and on the other, by using the queue mechanism, computational efficiency is improved. This method performs very well, especially in systems consisting of many reacting species and reactions.

Approximate stochastic simulation algorithms

Algorithms described in previous subsections are called exact stochastic simulation algorithms, due to the fact that for each reaction, its exact time instant is separately drawn by using appropriate pseudo random generator. However, execution of these algorithms may be quite time consuming. In order to achieve better computational efficiency, several approximated algorithms have been proposed. These approximated methods, also called accelerated methods, were also presented in this subsection.

In order to achieve the acceleration, these methods sacrifice the accuracy of the solution for its better time efficiency. In the majority of approaches, several reactions occur simultaneously in the coarse-grained time increments. In all accelerated methods, the following Leap Condition must be satisfied.

$$a_j\left(x(t)\right) \approx \text{const} \quad \text{for} [t, t+\tau] \tag{25}$$

In other words, each time step (leap) τ must be small enough, so the changes in the propensities have to be insignificant.

Basic τ -Leap Method

This method was originally proposed by Gillespie as an approximate, accelerated procedure for simulating occurrences of reactions in stochastic models (Gillespie, 2001). The idea is to divide the time scale into steps (τ-leaps) and to allow for several firings of each reaction at each time step. In order to successfully apply τ-leaping, we need to use the largest possible value of τ satisfying the Leap Condition. If τ is set to too small values, we may slow down the algorithm execution, to 1 or even 0 firings per step, which will result in the loss of numerical efficiency.

One way of checking for the best τ is the post leap check of differences in propensities for each reaction

$$\left| a_j(x+\lambda) - a_j(x) \right| \qquad j = 1, \ldots, M \tag{26}$$

and then adjusting the leap according to the value of this difference by increasing or decreasing τ. The post leap check gives a reliable condition for τ, but clearly this method might be too time consuming due to the necessity of repeated checking of the post-leap condition.

In his paper Gillespie (Gillespie, 2001) describes also a pre leap check for best fit of τ. To compute time step, by using the pre-leap check, we define auxiliary variables, $b_{ij}(x)$ and $\xi(x)$, as follows.

$$b_{ji}(x) = \frac{\partial a_j(x)}{\partial x_i} \quad j = 1, \ldots, M; \; i = 1, \ldots, N \tag{27}$$

$$\xi(x) = \sum_{j=1}^{M} a_j(x) v_j \tag{28}$$

To predict best τ we assume some specified fraction ε ($0 < \varepsilon < 1$) as a boundary and we choose minimum value of τ satisfying the equation below,

$$\tau = \min_{j \in [1, M]} \left[\frac{\varepsilon a_0(x)}{\left| \sum_{i=1}^{N} \xi(x) b_{ji}(x) \right|} \right] \tag{29}$$

In our further review of this algorithm, we can assume that the appropriate value of τ has been estimated. For the description of the further part of the algorithm, it does not make a difference, as to which method of τ estimation was applied. With this assumption, we present the structure (stated below) of the τ-leap algorithm.

Initialization. We proceed by initializing reactions R_j, $j = 1, ..., M$ (where M is the total number of reactions) and concentrations of species in the population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at the initial time.

Step 1. We continue by calculating propensities for each reaction. Using $\lambda = a_j(x)\,\tau$ as the mean value for reaction R_j we compute the number of its firings k_j by a random draw from the Poisson distribution with intensity λ,

$$k_j = \frac{\lambda^n e^{-\lambda}}{n!} \quad n = 0,1,2,\ldots \tag{30}$$

Step 2. We increment the time, $t = t + \tau$ and apply changes introduced by each R_j reaction to the population state vector $x(t)$, additionally taking into account the number of firings of R_j,

$$x = x + \sum_{j=1}^{M} k_j v_j \tag{31}$$

If the current time of the experiment does not exceed the total time, we go to Step 1 and follow the subsequent steps of the algorithm, otherwise we terminate.

The advantage of the above-described algorithm is its simplicity. However, this algorithm lacks the coordination of reacting species during single step, which can lead to a situation where population size of a species can become negative. Therefore, further improvements of this method have been proposed, which we describe below.

Chatterjee τ-Leap Method

The previously described basic approach was burdened with the risk of reaching negative population size, due to the lack of coordination between reacting species. The lack of coordination means that a situation can happen, where the number of firings of a reaction is greater than the number of available reactants. Chatterjee and coworkers in their work (Chatterjee et al., 2005) describe a method which allows for the avoidance of this case. To coordinate reactions and ensure mass conservation of the entire reactions network, two mechanisms are introduced. The first mechanism involves $k_j^{\bullet}$ – a maximum number of firings of a given reaction R_j, updated constantly between each subsequent firing during the τ interval. The second mechanism is the additional vector $\tilde{x}(t)$ introduced to track the currently available reacting population size during the τ time step. Before the execution of any firing, the currently available reacting population size is set to $\tilde{x}(t) = x(t)$. Each change in the population size introduced by subsequent firings of reaction R_j is tracked and applied to vector $\tilde{x}(t)$. The structure of the Chatterjee τ-Leap method is presented below.

Initialization. We proceed by initializing reactions R_j, $j = 1, ..., M$ (where M is the total number of reactions) and concentrations of species in population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at initial time.

Step 1. We continue with calculating propensities for each reaction and set the initial value of the population size change vector.

$$\tilde{x}(t) = x(t) \tag{32}$$

Then we compute the actual time step τ as the reciprocal of $a_0(x)$ (previously defined as the sum of all propensities) multiplied by a coarse-graining factor $f > 1$,

$$\tau = \frac{f}{a_0(x)} \tag{33}$$

After calculating the time leap, we proceed to Step 2 (for each reaction R_j).

Step 2.

- we compute the value of $k_j^{\bullet}$, a maximal number of possible firings for reaction R_j

$$k_j^{\bullet} = \min_{i=1,\ldots,N}(v_{ij} < 0)\left(\left\lfloor \frac{\tilde{x}_i}{|v_{ij}|} \right\rfloor\right) \tag{34}$$

- using random number generator based on binomial distribution, $B(k_j^{\bullet}, p)$ ($(k_j^{\bullet})$ is the number of experiments and p is the probability of the success) we draw the number of firings k_j,

$$k_j = B(k_j^{\bullet}, p), \text{ where } p = \frac{a_j(x)\tau}{k_j^{\bullet}} \tag{35}$$

- we apply changes to the $\tilde{x}(t)$ vector for all $v_{ij} < 0$,

$$\tilde{x}_i = \tilde{x}_i + v_{ij} k_j \tag{36}$$

Step 3. We update the time of simulation ($t = t + \tau$) and apply the changes to the population state vector $x(t)$ for each reaction R_j,

$$x = x + \sum_{j=1}^{M} k_j v_j \tag{37}$$

If the current time of experiment does not exceed the total time, we proceed to the Step 1 and follow the subsequent steps of the algorithm, otherwise we terminate the algorithm.

Cao τ-Leap Method

Cao and coworkers in their work (Cao et al., 2006) propose an optimized approach for tau-leap methods. They introduce a partition of reactions into two sets, the set of critical reactions J_c and the set of noncritical reactions J_{nc}. As the critical reaction, they define each reaction that within a specified number of firings n_c is depleting the number of any of the reactants. The purpose for such partitioning is the fact that the negative population typically arises from multiple firings of reactions that are close to consume all its reactants. To prevent the occurrence of such a scenario, the algorithm allows for only one firing from the critical subset during the specified time leap. Cao's method is described in detail below.

Initialization. We proceed by initializing reactions R_j, $j = 1, ..., M$ (where M is the total number of reactions) and concentrations of species in population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at the initial time.

Step 1. We start with partition of all reactions into two groups, critical J_c and noncritical J_{nc} by checking which reaction is about to deplete its reactants in the range defined by the value of n_c using the condition,

$$\min_{i\in[1,N]}\left(\frac{x_i(t)}{|v_{ij}|}\right) < n_c \tag{38}$$

We continue by calculating the time leap candidate T_{nc} for the subset of noncritical reactions.

$$T_{nc} = \max\left(a_j \frac{(x)}{a_j}\left(x(t+T_{nc})\right) \le \varepsilon\right) \quad \text{where } 0 < \varepsilon \le 1 \tag{39}$$

If the value of the calculated candidate is less than some small multiple (usually 10) of $1/a_0(x)$ we abort execution of following steps and for specified number of times (usually 100) we execute single-reaction steps of the SSA and return to Step 1. Otherwise if the value of the candidate is greater, we proceed to Step 2.

Step 2. Here, we start with calculation of the sum a_c of all propensities over members of the critical subset of reactions,

$$a_c = \sum_{j\in J_c} a_j(x) \tag{40}$$

We generate the second time leap candidate T_c (for critical subset) as a random draw of the exponential random variable with mean value $\lambda = 1/a_c(x)$,

$$T_c = Exp(\lambda) \tag{41}$$

Step 3. From the above, two candidates for time leap τ we choose the smaller one and treat it as an actual time leap. In the case where $T_c < T_{nc}$ single critical reaction is executed. Reaction is chosen by a random sample from the subset J_c. We calculate number of firings k_j for reactions from the subset of non-critical reactions J_{nc} by using Poisson distribution with intensity given by the mean value $a_j(x)$, $j \in J_{nC}$.

Step 4. We update the time of simulation ($t = t + \tau$) and apply the changes in the population state vector $x(t)$ for each reaction R_j from the noncritical subset,

$$x = x + \sum_{j=1}^{M} k_j v_j \quad \text{where } j \in J_{nC} \tag{42}$$

If the current time of experiment does not exceed the total time, we go to Step 1 and follow the subsequent steps of the algorithm, otherwise we terminate the algorithm.

Hybrid stochastic simulation algorithms

A group of the stochastic simulation algorithms, called hybrid stochastic simulation algorithms were developed for bio-molecular systems that include reactions with very wide scope of values of propensities of reaction. The researchers propose that in such a case, fast reaction can be solved by using differential equations (deterministic modeling), while for slow reactions the stochastic simulations machinery is used. This approach is presented in this subsection.

Haseltine and Rawlings (Haseltine and Rawlings, 2002) propose a modification of the standard Gillespie Algorithm resulting in a hybrid method based on partially solving models with the use of differential equations

and partially in a stochastic manner. By applying the appropriate partitioning condition, we split all reactions to two groups, of slow and fast reactions. Clearly, as the propensity greatly relies on population size, the partition threshold is defined by the number of bio-particles. Fast reactions are commonly defined as such, for which we talk about hundreds of thousands of reacting bio-particles. Such reactions occur very frequently during simulation (time leap between occurrence is very small), which allows for approximation of the process by using a system of coupled first-order ODEs guaranteeing sufficient precision. In contrast, for slow reactions, the number of bio-particles is assumed to be of the order of hundreds or less. These reactions are simulated by using stochastic random generators. One can notice that there is an area in between which is difficult to classify as either of the types of reactions.

Based on the idea of partitioning reactions into fast and slow, Haseltine and Rawlings proposed the following hybrid algorithm.

Initialization. As mentioned before, we proceed by dividing the reactions into two groups. We describe fast reactions separately by using ODEs. After that we initialize slow reactions R_j, $j = 1, ..., M$ (where M is the total number of slow reactions) and concentration of species in population vector $x(t) = (x_1(t), ..., x_N(t))$ (where N is the total number of species) at initial time.

Step 1. We compute the propensities for all slow reactions and sum them at given time t.

$$a_0\big(x(t)\big) = \sum a_j\big(x(t)\big) \tag{43}$$

We proceed by drawing two random numbers r_1 and r_2 from the uniform distribution.

Step 2. We compute deterministic changes by solving differential equations to the time $t + \Delta t$ for which the equation stated below is true:

$$\ln(r_1) + \int_t^{t+\Delta t} a_0\big(x(t)\big)\,dt = 0 \tag{44}$$

Step 3. We check which slow reaction R_μ will occur between t and $t + \Delta t$ and we update properly population state vector $x(t)$,

$$\sum_{j=1}^{\mu-1} a_j(t+\Delta t) \le r_2 a_0(t+\Delta t) < \sum_{j=1}^{\mu} a_j(t+\Delta t) \tag{45}$$

If the current time of experiment does not exceed the total time, we proceed to the Step 1 and follow the subsequent steps of the algorithm, otherwise we terminate the algorithm.

Computational complexity of stochastic simulation algorithms

When describing computational algorithms, it is very desirable to include estimations concerning their computational complexity as functions of parameters describing the size of the analyzed problem. In this section, we provide some comments on this issue.

All stochastic simulation algorithms which we have described in previous sections have simple logical structure. Diagrams of their algorithms, presented in chapter "Deterministic and stochastic simulation algorithms" contain only one loop structure. In conclusion, in order to estimate computational complexity of each algorithm, one has to estimate (i) computational complexity of one pass of the loop and (ii) the number of passes of the loop.

As for (i), computational operations inside the body of the loop include, generation of random numbers, selection of one reaction (several reactions) from all possible reactions, operations related to updating of the population state vectors. These three elements can generally have different computational load, but the third one (updating of the population state vector) is usually negligible compared to the first two. In different methods the computational load of the first two operations can differ. For example the computational load of the one pass of the body of the loop in the Direct Method is estimated as $O(M)$ (Mauch and Stalzer, 2011; Schulze, 2002).

As for (ii), it is rather impossible to estimate, before performing some experiments with the system, as to how many loop executions will the simulation problem require.

When estimating time complexity of approximate algorithm versus exact algorithm, it is possible to propose some rough estimates by calculating the number of firings of reaction inside one leap.

Summarizing the above-mentioned comments, we found basing the estimates of computational complexity on experimental setup as most practical. When reporting computational results, we have added data on the time

of computations and on the number of steps. This makes comparisons between different methods possible.

Computational examples

In this section, we illustrate methods presented above by computational examples. We focus our attention on three different models. The first one is a previously described model explaining $p53 \mid Mdm2$ regulatory mechanisms described in (Hat et al., 2009). For this model, we compared the Direct Method with two different tau-leap algorithms. The second model describes infection of a cell by a virus. For this viral infection model, we additionally performed comparisons of the hybrid stochastic simulation method to the Direct Method and tau-leap family methods. The last model, in contrast to previous examples concerning bio-molecules, describes a behavior of an ecosystem, more precisely interactions of two populations, prey and predator. For this Prey-Predator model, we compared efficiencies of Cao and Chatterjee implementations of the tau-leap method.

In the computational examples, which are mentioned below, we present results obtained in the following scenario: first we perform numerous random simulations of evolution of the studies systems, according to the assumed method and with the chosen parameters. On the basis of performed simulations, we then compare different methods by comparisons of values obtained by (i) averaging across multiple simulation experiments, (ii) comparisons of standard deviations obtained from multiple simulations experiments, (iii) comparisons of frequencies of occurrences of reactions in different stochastic simulation algorithms. In this approach, we considered Direct Method as the "golden standard", i.e., accuracies of different stochastic simulation methods were measured by the distance of the obtained realizations to the realization of the stochastic evolution process obtained by using the Direct Method. Apart from the above-stated comparisons, we have also compared time efficiencies of different methods.

Model of p53|Mdm2 signaling pathway

The model of interactions of protein molecules $p53$ and $Mdm2$, is described by the set of first order differential equations (5)-(7), or equivalently by the list of reactions (8)-(13). For this model, we have performed computational experiments using parameters specified in the paper (Hat et al., 2009) presented in Table 1.

Table 1. Parameters used in experiment for simulation of model of $p53 \mid Mdm2$

m	2
n	6
$s1$	16
$s2$	8
$s3$	80
$s4$	$1 \cdot 10^{5}$
$k1$	$3.5 \cdot 10^{-3}$
$k2$	2300
$kd1$	$1 \cdot 10^{-13}$
$kd2$	$3 \cdot 10^{-3}$

For the Model of $p53 \mid Mdm2$ signaling pathway, we have performed stochastic simulation experiments by executing 3000 separate trials per each of the studied methods, "Direct Method", "Tau-Leap Chatterjee" and "Tau-Leap Cao". For both approximate tau-leap methods, their "run time" parameters were adjusted experimentally, such that accuracy was optimized without significant loss in terms of time efficiency. Due to the interlinked positive and negative feedback loops, we observed oscillations in concentration of each of the protein species. Oscillatory behavior of time plots in Figure 2 has biological explanation of directing properly cellular response to DNA damage presented in detail in (Hat et al., 2009).

Reviewing the results depicted graphically in Figure 2, we noticed that concentrations, particularly for cytoplasmic $Mdm2$, oscillate around 3 million bio-molecules. At such high level of concentrations, reactions occur at a very fast rate. For such systems, mean values obtained from multiple trials practically do not differ from the deterministic solution. We noticed that the loss of accuracy is insignificant in this case. This property is visible in the graph presenting time plots of signals for each method drawn in Figure 2, where only with large "zoom in" we could observe differences between averages. The results for Cao tau-leap method and the Direct Method were almost overlapping, while the Chatterjee tau-leap method provided results with a (slightly) higher error rate. For both approximated methods, the accuracy loss in terms of computational efficiency is insignificant.

However, with the high accuracy of computational results obtained by all of the applied methods, we obser-

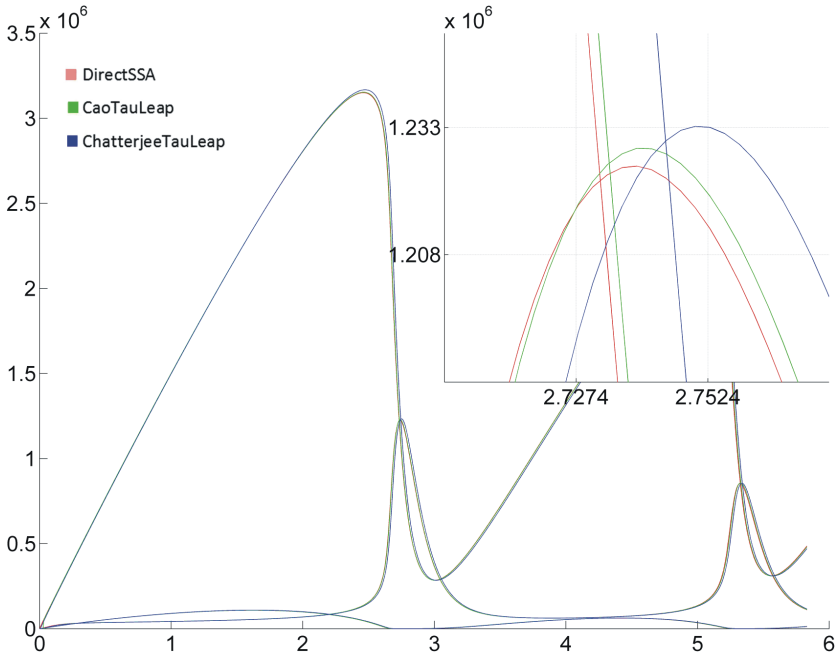

Fig. 2. Chart averaged realizations of $p53|Mdm2$ model using exact algorithm (red) and two approximated tau-leap methods: Cao (green) and Chatterjee (blue). In upper-right corner, we see a large "zoom-in" where we can observe differences between averaged results

Table 2. Summary of averaged results from 3000 realizations per single method

Algorithm	Direct Method	Cao Tau-Leap	Chatterjee Tau-Leap
Parameters		epsilon = 0.5	f = 2000
Average time (s)	13.2728	0.1534	0.157
Average number of steps per trial	33498160	8855.87	16745.01

ved significant differences in their time efficiencies. To calculate single trial with exact stochastic method, around 13 seconds were required, while for both approximate methods simulation takes around 150 milliseconds. These results are summarized in Table 2, where apart from mean times of simulations, mean numbers of steps per one experiment for each of the methods is also reported.

Viral infection model

The second model describes a simple network of a viral infection of a cell (Haseltine and Rawlings, 2002). The model consists of three species: viral structural protein (S) (48) and two copies of viral nucleic acids, template (T) (46) and genomic (G) (47). The deterministic dynamics of interactions of these species is described by the following system of differential equations:

$$\frac{dT}{dt} = k_1 \cdot G - k_2 \cdot T \tag{46}$$

$$\frac{dG}{dt} = k_2 \cdot T - k_1 \cdot G - k_4 \cdot G \cdot S \tag{47}$$

$$\frac{dS}{dt} = k_5 \cdot T - k_6 \cdot s - k_4 \cdot G \cdot S \tag{48}$$

Initial values of reactants are as follows: $T = 1$, $G = 1$, $S = 1$. These values are understood as numbers of interacting molecules. As for parameters, their values were obtained from the paper (Haseltine and Rawlings, 2002) and are reprinted in Table 3.

Table 3. Parameters used in experiment for simulation of model of viral infection

Parameter	Value
k_1	0.025
k_2	0.25
k_3	1
k_4	$7.5 \cdot 10^{-6}$
k_5	1000
k_6	1.99

We used this model to present the effectiveness of stochastic simulation algorithms with regard to the idea of the hybrid method. Reactions described in model are partioned into two sets, slow (treated in a stochastic way). As slow reactions in equations (46)-(48), we treat all reactions influencing directly concentration of viral nucleic acids. This means that evolutions described by equations (46) and (47) are modeled by using stochastic simulation algorithms. In contrast, evolution governed by equation (48) describing viral structural protein was calculated using the deterministic approach.

The total time of single experiment was set to 200 days. For each applied method, we executed 5000 experiments and generated mean results. Below we present outcomes for 4 different methods as graphical plots of signals, along with the table presenting calculation time for each method.

As in the previous example, in this case, exact stochastic method is rather inefficient in terms of the time consumption. Single experiment is computed for approximately 6.5 seconds. Both approximate tau-leap methods proved that the model can be calculated in more efficient manner while still preserving accuracy of the results. We noticed that results obtained for Chatterjee algorithm are more accurate (Fig. 3) in comparison to Cao method in terms of outcome of exact solution. In addition, we found performance increase in the calculation time. Still partitioning of the model into two subsets of slow and fast reactions allowed us for further optimization. For hybrid method, we observed that the results obtained by us are almost overlapping with the exact algorithm.

Prey-Predator

$$\frac{dN}{dt} = rN \cdot \left(1 - \frac{N}{k}\right) - \frac{aN}{1+\omega N} \cdot P \tag{49}$$

$$\frac{dP}{dt} = P \cdot \left(c \cdot \frac{aN}{1+\omega N} - g\right) \tag{50}$$

Although our previous examples focused around models describing behavior of bio-molecules, in the last example, we performed a comparison of approximated methods in the Prey-predator model describing an eco-system (Pineda-Krch, 2008). The model is given by two differential equations describing rates of change of sizes of populations of prey's (N) (49) and predator's (P) (50). Due to its construction this model presents additional possibilities in analysis of accuracies of the approximate stochastic algorithms versus accurate ones. We noticed that the prey population is bound by the value of parameter k (49), also we could see that predator's birth and death rates are dependent on the current predator's density. In the case of the deterministic realization for the chosen parameters, we observe un damped oscillations for both of the signals, but in the case of stochastic methods there is a possibility of occurrence of a situation in which all predators become extinct. This leads to the state where predators population remains empty and preys density gradually stabilizes at a fixed value determined by the parameter k. In stochastic simulations, we observed the occurrence of extinction of predators with probability one, provided that simulation time was long enough. Averaging over many stochastic simulations led to interesting results, depending on the simulation method used, the pattern of "averaged" oscillations changed. The model is an interesting example of the analysis of accuracy of approximated methods, as we can compare different modes of oscillations suppression.

Table 4. Summary of averaged results from 5000 realizations of per single method

Method	Parameters	Total time (s)	Average time (s)
Direct Method		32410	6.482
Cao Tau-Leap	epsilon = 0.9	3347	0.669
Chatterjee Tau-Leap	f = 1500	216	0.043
Haseltine--Rawlings	deterministic step = 1 stochastic step = 0.01	669	0.134

To perform the experiment, we collected data from 5000 separate trials of each method. The total simulation time per a single trial was set to 100 time units. Parameters and initial condition values can be found in Table 5.

In Table 6, we have reported results of two different tau-leap approximate methods. We noticed that results for both Cao (Cao et al., 2006) and Chatterjee (Chatter-

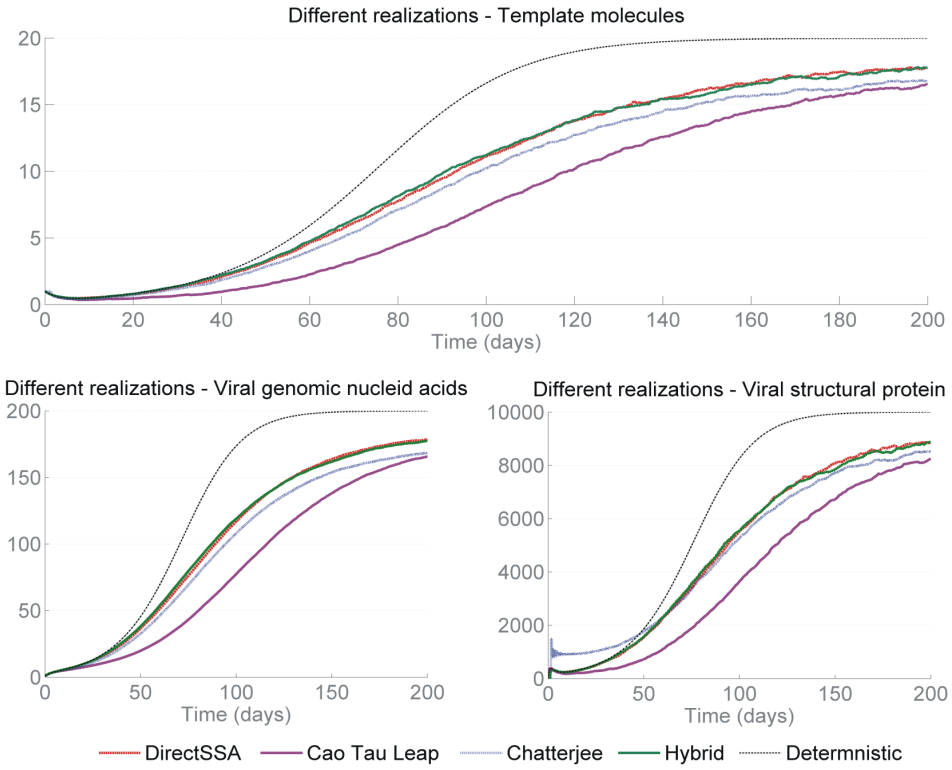

Fig. 3. Time plots of concentrations of protein and viral species obtained by using different stochastic simulation methods (Viral infection model)

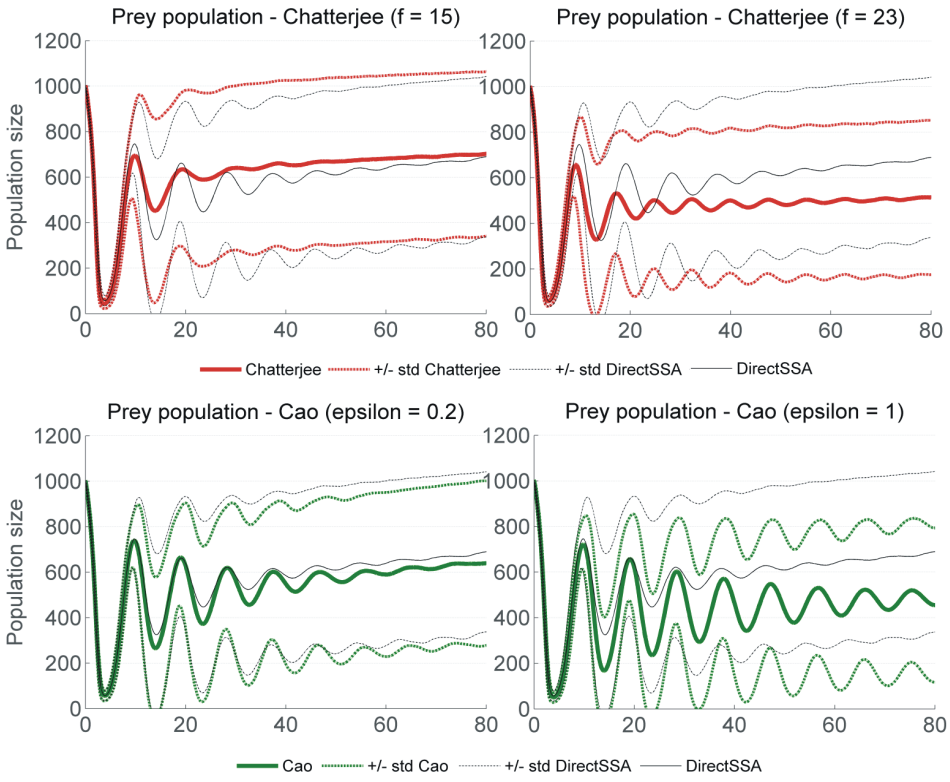

Fig. 4. Plot for prey population – different realizations

Table 5. Parameters used in experiment for simulation of prey-predator model

N	1000
P	100
r	1
k	1000
a	0.007
ω	0.0035
c	2
g	2

Table 6. Summary of results for different methods. In column A, we present average computation time for single simulation. In column B, we report averaged number of steps, plus standard deviation for those numbers in column C. In the last column D, we report information on reaction averaged differences of number of occurrences of reactions between the approximate and exact method of stochastic simulation

Method	Parameter	A	B	C	D
Direct Method		1.96	286334	58%	
Cao	epsilon = 0.2	0.55	38663	37%	8%
Cao	epsilon = 1	0.1	3829	336%	32%
Chatterjee	f = 15	0.39	20450	29%	7%
Chatterjee	f = 23	0.23	11336	17%	33%

jee et al., 2005) methods strongly depend on the choice of their "run time" parameter, both in terms of time consumption and quality of calculated results. We also noticed that average time needed to perform a single trial using exact stochastic method took around 2 seconds while for both approximate methods it oscillated around 500 milliseconds. With the increase in the value of parameter epsilon in the Cao method we observed reduction in the calculation time but also great loss of exactness which can be seen in Figure 4. The same situation was observed for Chatterjee method when we increased the value of the parameter f. Additionally, Table 6 reports information about average number of steps and standard deviation (between different simulations) for those numbers. We have also collected information on the average differences of the number of reaction occurrence for each of the approximate methods compared to the realization obtained by using the exact method. In both cases where we sacrifice exactness to increase calculation performance (changes in method call parameters), we notice (in Fig. 4) the lack of oscillation suppression especially in the case of Cao implementation of approximate simulations. For this experiment, best results were those obtained for Cao algorithm.

Conclusion

There is a great interest in using stochastic algorithms for simulations in system biology models. However, several problems, particularly those concerning compromises between computational efficiency and exactness of results, have been reported in the previously published literature.

In this paper, we have surveyed existing methodologies for simulating systems of interacting bio-molecules. We have discussed their advantages and drawbacks and we have demonstrated computational examples of their applications.

Our computational examples, developed for rather simple systems of interacting biomolecules, confirm that the proper choice of the stochastic simulation method as well as proper tuning of the parameters of the method, may be very important for the obtained results.

Implementing different methods of stochastic simulations and comparing their outcomes seems like a good strategy. Distributions of simulation errors can be obtained by repeated multiple simulations. Comparisons of outcomes of different methods can provide knowledge on the exactness of solutions and on the reliability of obtained results.

Acknowledgments

This work was supported by the European Community from the European Social Fund (Paweł Lachor UDA-POKL.04. 01.01-00-106/09-00) and by The Foundation for Polish Science (Krzysztof Puszyński N N518 287540).

References

Butcher J.C. (2003) *Numerical methods for ordinary differential equations,* John Wiley and Sons.

Cao Y., Gillespie D., Petzold L. (2006) *Efficient Stepsize Selection for the Tau-Leaping Method.* J. Chem. Phys. 124: 044109.

Chatterjee A., Vlachos D.G., Katsoulakis M.A. (2005) *Binomial distribution based τ-leap accelerated stochastic simulation.* J. Chem. Phys., 122: 024112.

Geva-Zatorsky N., Rosenfeld N., Itzkovitz S., Milo R., Sigal A., Dekel E., Yarnitzky T., Liron Y., Polak P., Lahav G., et al. (2006) *Oscillations and variability in the p53 system.* Mol. Syst. Biol. 2: 0033.

Gibson M.A., Bruck J. (2000) *Efficient exact stochastic simulation of chemical systems with many species and many channels*. J. Phys. Chem. 104: 1876-1889.

Gillespie D. (1976) *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions*. J. Comput. Phys. 22: 403-434.

Gillespie D. (2001) *Approximate accelerated stochastic simulation of chemically reacting systems*. J. Chem. Phys. 115: 1716.

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

Hat B., Puszyński K., Lipniacki T. (2009) *Exploring mechanisms of oscillations in p53 and NF-kappa B systems*. IET Sys. Biol. 3: 342-355.

Krishna S., Jensen M., Sneppen K. (2006) *Minimal model of spiky oscillations in NF-κB signaling*. PNAS 104: 29-10845.

Mauch S., Stalzer M. (2011) *Efficient formulations for exact stochastic simulation of chemical systems*. IEEE/ACM Trans. Comput. Biol. Bioinform. 8(1).

Meng T.C., Somani S., Dhar P. (2004) *Modeling and simulation of biological systems with stochasticity*. Silico Biol. 4: 0024.

Pineda-Krch M. (2008) *Implementing the stochastic simulation algorithm in R using the GillespieSSA package*. J. Stat. Software 25(12): 1-18.

Schulze T.P. (2002) *Kinetic Monte Carlo simulations with minimal searching*. Phys. Rev. E 65: 036704.

Swainston N., Mendes P. (2009) *libAnnotationSBML: a library for exploiting SBML annotations*. Bioinformatics 25(17): 2292-2293.

Waage P. Guldberg C.M. (1864) *Videnskabs-Selskabet i Christiana*. Forhandlinger, 35.