Stochastic Runge–Kutta Software Package for Stochastic Differential Equations

M.N. Gevorkyan, T.R. Velieva, A.V. Korolkova, D.S. Kulyabov and L.A. Sevastyanov

Abstract As a result of the application of a technique of multistep processes stochastic models construction the range of models, implemented as a self-consistent differential equations, was obtained. These are partial differential equations (master equation, the Fokker–Planck equation) and stochastic differential equations (Langevin equation). However, analytical methods do not always allow to research these equations adequately. It is proposed to use the combined analytical and numerical approach studying these equations. For this purpose the numerical part is realized within the framework of symbolic computation. It is recommended to apply stochastic Runge–Kutta methods for numerical study of stochastic differential equations in the form of the Langevin. Under this approach, a program complex on the basis of analytical calculations metasystem Sage is developed. For model verification logarithmic walks and Black–Scholes two-dimensional model

M.N. Gevorkyan (✉) · T.R. Velieva · A.V. Korolkova · D.S. Kulyabov · L.A. Sevastyanov
Department of Applied Probability and Informatics,
Peoples' Friendship University of Russia, Miklukho-Maklaya Street 6,
Moscow 117198, Russia
e-mail: mngevorkyan@sci.pfu.edu.ru

T.R. Velieva
e-mail: trvelieva@gmail.com

A.V. Korolkova
e-mail: avkorolkova@gmail.com

D.S. Kulyabov
e-mail: yamadharma@gmail.com

L.A. Sevastyanov
e-mail: leonid.sevast@gmail.com

D.S. Kulyabov
Laboratory of Information Technologies, Joint Institute for Nuclear Research,
Joliot-Curie 6, Dubna, Moscow Region 141980, Russia

L.A. Sevastyanov
Bogoliubov Laboratory of Theoretical Physics, Joint Institute for Nuclear Research,
Joliot-Curie 6, Dubna, Moscow Region 141980, Russia

© Springer International Publishing Switzerland 2016

W. Zamojski et al. (eds.), *Dependability Engineering and Complex Systems*,
Advances in Intelligent Systems and Computing 470,
DOI 10.1007/978-3-319-39639-2_15

are used. To illustrate the stochastic “predator–prey” type model is used. The utility of the combined numerical-analytical approach is demonstrated.

Keywords Runge–Kutta methods · Stochastic differential equations · RED queueing discipline · Active queue management · Computer algebra software · Sage CAS

1 Introduction

The mathematical models adequacy may be largely increased by taking into account stochastic properties of dynamic systems. Stochastic models are widely used in chemical kinetics, hydrodynamics, population dynamics, epidemiology, filtering of signals, economics and financial mathematics as well as different fields of physics [1]. Stochastic differential equations (SDE) are the main mathematical apparatus of such models.

Compared with numerical methods for ODEs, numerical methods for SDEs are much less developed. There are two main reasons for this: a comparative novelty of this field of applied mathematics and much more complicated mathematical apparatus. The development of new numerical methods for stochastic case in many ways is similar to deterministic methods development [2–7]. The scheme, that has been proposed by Butcher [8], gives visual representation of three main classes of numerical schemes.

The accuracy of numerical scheme may be improved in the following way: by adding additional steps (multi-step), stages (multi-stage), and the derivatives of drift vector and diffusion matrix (multi-derivative) to the scheme.

Multi-step (Runge–Kutta like) numerical methods are more suitable for the program implementation, because they can be expressed as a sequence of explicit formulas. Thus, it is natural to spread Runge–Kutta methods in the case of stochastic differential equations.

The main goal of this paper is to give a review of stochastic Runge–Kutta methods implementation made by authors for Sage computer algebra system [9]. The Python programming language and NumPy and SciPy modules were used for this purpose because Python programming language is an open and powerful framework for scientific calculations.

The authors have faced the necessity of stochastic numerical methods implementation in the course of work on the method of stochastization of one-step processes [10, 11]. As The RED [10, 12] model was taken as an example of the methodology application in order to verify the obtained SDE models by numerical experiments.

1.1 The Choice of Programming Language for Implementing Stochastic Numerical Methods

The following requirements to computer algebra system's programming language were taken into account:

- advanced tools for manipulations with multidimensional arrays (up to four axes) with a large number of element are needed;
- it is critical to be able to implement parallel execution of certain functions and sections of code due to the need of a large number of independent calculations according to Monte-Carlo method;
- functions to generate large arrays of random numbers are needed;

Computer algebra system Sage generally meets all these requirements. NumPy module is used for array manipulations and SciPy module—for n-point distribution creation.

2 Wiener Stochastic Process

In this section only the most important definitions and notations are introduced. For brief introduction to SDE see [13, 14], for more details see [1, 15, 16].

A standard scalar Wiener process $\mathrm{W(t)}, \mathrm{t} \geq 0$ is a stochastic process which depends continuously on $\mathrm{t} \in [\mathrm{t_0}], \mathrm{T}$ and satisfies the following three conditions:

- $\mathrm{P}\{\mathrm{W}(0) = 0\} = 1$, in other words $\mathrm{W}(0) = 0$ almost surely;
- $\mathrm{W(t)}$ has independent increments i.e. $\{\Delta \mathrm{W_i}\}_0^{\mathrm{N}-1}$—independently distributed random variables; $\Delta W_i = W(t_{i+1}) - W(t_i)$ and $0 \leq \mathrm{t_0} < \mathrm{t_1} < \mathrm{t_2} < \cdots < \mathrm{t_N} \leq \mathrm{T}$;
- $\Delta \mathrm{W_i} = \mathrm{W}(\mathrm{t}_{i+1}) - \mathrm{W}(\mathrm{t_i}) \sim \mathcal{N}(0, \mathrm{t}_{i+1} - \mathrm{t_i})$, where $0 \leq t_{i+1} < t_i < T, i = 0, 1, \ldots, N-1$.

Notation $\Delta \mathrm{W_i} \sim \mathcal{N}(0, \Delta t_i)$ denotes that ΔW_i is a normally distributed random variable with zero mean $\mathbb{E}[\Delta W_i] = \mu = 0$ and unit variance $\mathbb{D}[\Delta W_i] = \sigma^2 = \Delta t_i$.

Multidimensional Wiener process $W^{\alpha}(t) : \Omega \times [t_0, T] \to \mathbb{R}^m, \forall \alpha = 1, \ldots, m$, is consist of independent $\mathrm{W}^1(\mathrm{t}), \ldots, \mathrm{W}^{\mathrm{m}}(\mathrm{t})$ scalar Wiener processes. The increments $\Delta \mathrm{W}_{\mathrm{i}}^{\alpha}$ are mutually independent normally distributed random variables with zero mean and unit variance.

2.1 Itô SDE for Multidimensional Wiener Process

Let's consider a random process $\mathbf{x}(t) = \left(x^1(t), \ldots, x^d(t)\right)^T$, where $\mathbf{x}(t)$ belongs to the function space $L^2(\Omega)$ with norm $\|\cdot\|$. We assume that stochastic process $x(t)$ is a solution of Ito SDE [1, 16]:

$$x^{\alpha}(t) = f^{\alpha}(t, x^{\gamma}(t))dt + \sum_{\beta=1}^{m} G^{\alpha}_{\beta}(t, x^{\gamma}(t))dW^{\beta}, \tag{1}$$

where $\alpha, \gamma = 1, \ldots, d, \beta = 1, \ldots, m$, vector value function $f^{\alpha}(t, x^{\gamma}(t)) = f^{\alpha}\left(t, x^{1}(t), \ldots, x^{d}(t)\right)$ is a *drift vector*, and matrix value function $g^{\alpha}_{\beta}(t, x^{\gamma}(t))$ is a *diffusion matrix*, $W^{\alpha} = (W^{1}, \ldots, W^{m})^{T}$ is a multidimensional Wiener process, also known as a driver process of SDE.

Let's introduce the discretization of interval $[t_0, T]$ by sequence $t_0 < t_1 < \cdots < t_N = \mathrm{T}$ with step $h_n = t_{n+1} - t_n$, where $n = 0, \ldots, N-1$ and $h = max\{h_{n-1}\}_1^N$—minimal step. We also consider the step $h_n = h$ to be a constant; x_n—mesh function for random process $x(t)$ approximation, i.e. $\mathbf{x}_0 = \mathbf{x}(t_0)$, $\mathbf{x}_n \approx \mathbf{x}(t_n) \forall n = 1, \ldots, N$.

2.2 Strong and Weak Convergences of Approximation Function

It is necessary to define the criterion for measuring the accuracy of process $\mathbf{x}(t)$ approximation by sequences of functions $\{\mathbf{x}_n\}_1^N$. Usually *strong* and *weak* [2, 7, 16] criteria are defined.

The sequence of approximating functions $\{\mathbf{x}_n\}_1^N$ converges with order p to exact solution $\mathbf{x}(t)$ of SDE in moment T in *strong sense* if constant $C > 0$ exists and $\delta_0 > 0$, such as $\forall h \in (0, \delta_0]$ the condition (2) is fulfilled.

$$\mathbb{E}(\|\mathbf{x}(T) - \mathbf{x}_N\|) \leq Ch^p \tag{2}$$

The sequence of approximating functions $\{\mathbf{x}_n\}_1^N$ converges with order p to solution $\mathbf{x}(t)$ of SDE in moment T in *weak seance* if constant $C_F > 0$ exists and $\delta_0 > 0$, such as $\forall h \in (0, \delta_0]$ the condition (3)

$$|\mathbb{E}[F(\mathbf{x}(T))] - \mathbb{E}[F(\mathbf{x}_N)]| \leq C_F h^p \tag{3}$$

is fulfilled.

3 Stochastic Runge–Kutta Methods

3.1 Strong Stochastic Runge–Kutta Methods for SDEs with Scalar Wiener Process

In the case of scalar SDE and Wiener process the following scheme is valid:

$$X_0^i = x_n + \sum_{j=1}^{s} A_0^i f\left(t_n + c_0^j h_n, X_0^j\right) h_n + \sum_{j=1}^{s} B_{0j}^i g\left(t_n + c_1^j h_n, X_1^j\right) \frac{I^{10}(h_n)}{\sqrt{h_n}}, \quad (4)$$

$$X_1^i = x_n + \sum_{j=1}^{s} A_{1j}^i f\left(t_n + c_0^j h_n, X_0^j\right) h_n + \sum_{j=1}^{s} B_{1j}^i g\left(t_n + c_1^j h_n, X_1^j\right) \sqrt{h_n}, \quad (5)$$

$$\begin{aligned} x_{n+1} = x_n &+ \sum_{i=1}^{s} a_i f\left(t_n + c^i h_n, X_0^i\right) h_n \\ &+ \sum_{i=1}^{s} \left(b_i^1 I^1(h_n) + b_i^2 \frac{I^{11}(h_n)}{\sqrt{h_n}} + b_i^3 \frac{I^{10}(h_n)}{h_n} + b_i^4 \frac{I^{111}(h_n)}{h_n} \right) g\left(t_n + c_1^i, X_1^i\right) \end{aligned} \quad (6)$$

In Robler preprint [7] there are two realisations of this scheme for $s = 4$. The first scheme we will denote as SRK1W1, the second—as SRK2W2. The method SRK1W1 has strong order $(p_d, p_s) = (2.0, 1.5)$, the method SRK2W1 has strong order $(p_d, p_s) = (3.0, 1.5)$. Another scheme with strong order $p_s = 1.0$ can be found in [1].

3.2 Strong Stochastic Runge–Kutta Methods for SDE System with Multidimensional Wiener Process

For Itô SDE system with multidimensional Wiener process the stochastic Runge-Kutta scheme with strong order $p_s = 1.0$ can be obtained using single and double Itô integrals [7].

$$\begin{aligned} X^{0i\alpha} &= x_n^\alpha + \sum_{j=1}^{s} A_{0j}^i f^\alpha\left(t_n + c_0^j h_n, X^{oj\beta}\right) h_n + \sum_{l=1}^{m} \sum_{j=1}^{s} B_{0j}^i G_l^\alpha\left(t_n + c_1^j h_n, X^{lj\beta}\right) I^l(h_n) \\ X^{ki\alpha} &= x_n^\alpha + \sum_{j=1}^{s} A_{1j}^i f^\alpha\left(t_n + c_0^j h_n, X^{oj\beta}\right) h_n + \sum_{l=1}^{m} \sum_{j=1}^{s} B_{1j}^i G_l^\alpha\left(t_n + c_1^j h_n, X^{lj\beta}\right) \frac{I^l(h_n)}{\sqrt{h_n}}, \\ X^{0i\alpha} &= x_n^\alpha + \sum_{j=1}^{s} A_{0j}^i f^\alpha\left(t_n + c_0^j h_n, X^{oj\beta}\right) h_n + \sum_{l=1}^{m} \sum_{j=1}^{s} B_{0j}^i G_l^\alpha\left(t_n + c_1^j h_n, X^{lj\beta}\right) I^l(h_n) \\ x^{\overline{\alpha}_{n+1}} &= x_n^\alpha + \sum_{i=1}^{s} a_i f^\alpha\left(t_n + c_0^i h_n, X^{oj\beta}\right) h_n + \sum_{k=1}^{m} \sum_{i=1}^{s} \left(b_i^1 I^k(h_n) + b_i^2 \sqrt{h_n} \right) G_k^\alpha\left(t_n + c_1^j h_n, X^{ki\beta}\right) \end{aligned} \quad (7)$$

where $n = 0, 1, \ldots, N-1; i = 1, \ldots, s; \beta, k = 1, \ldots, m; \alpha = 1, \ldots, d$.

There are two realisations of this scheme for s = 3 in Robler preprint [7]. The SRK1Wm method has the strong order $(p_d, p_s) = (1.0, 1.0)$ of convergence and the method SRK2Wm has the strong order $(p_d, p_s) = (2.0, 1.0)$ of convergence.

It is also important to mention, that the deterministic part of methods SRKp1W1 and SRKp2W1 does not essentially influence the error value.

3.3 Stochastic Runge–Kutta Methods with Weak Convergence of P = 2.0 Order

Numerical methods with weak convergence are the best for approximation of characteristics of the distribution of the random process $x^{\alpha}(t)$. Weak numerical method does not require information about the exact trajectory of a Wiener process W_n^{α}, and the random variables for these methods can be generated on another probability space. So we can use distribution which is easily generated on the computer.

For Itô SDE system with multidimensional Wiener process the following stochastic Runge-Kutta scheme with weak order $p_s = 2.0$ is valid [7].

$$
\begin{aligned}
X^{0i\alpha} &= x_n^{\alpha} + \sum_{j=1}^{s} A_{0j}^{i} f^{\alpha}\left(t_n + c_0^j h_n, X^{oj\beta}\right) h_n + \sum_{j=1}^{s}\sum_{l=1}^{m} B_{0j}^{i} G_l^{\alpha}\left(t_n + c_1^j h_n, X^{lj\beta}\right)\hat{I}^{l},\\
X^{ki\alpha} &= x_n^{\alpha} + \sum_{j=1}^{s} A_{1j}^{i} f^{\alpha}\left(t_n + c_0^j h_n, X^{oj\beta}\right) h_n + \sum_{j=1}^{s} B_{1j}^{i} G_l^{\alpha}\left(t_n + c_1^j h_n, X^{lj\beta}\right)\sqrt{h_n},\\
\widehat{X}^{ki\alpha} &= x_n^{\alpha} + \sum_{j=1}^{s} A_{2j}^{i} f^{\alpha}\left(t_n + c_0^j h_n, X^{oj\beta}\right) h_n + \sum_{j=1}^{s}\sum_{l=1,l\neq k}^{m} B_{2j}^{i} G_l^{\alpha}\left(t_n + c_1^j h_n, X^{lj\beta}\right)\frac{\hat{I}^{kl}}{\sqrt{h_n}},\\
\overline{x^{\alpha}_{n+1}} &= x_n^{\alpha} + \sum_{i=1}^{s} a_i f^{\alpha}\left(t_n + c_1^i, X^{ki\beta}\right) h_n + \sum_{i=1}^{s}\sum_{k=1}^{m}\left(b_i^1\hat{I}^{k} + b_i^2\frac{\hat{I}^{kk}}{\sqrt{h_n}}\right) G_k^{\alpha}\left(t_n + c_1^i h_n, X^{ki\beta}\right)\\
&\quad + \mathrm{I}\sum_{i=1}^{s}\sum_{k=1}^{m}\left(b_i^3\hat{I}^{k} + b_i^4\sqrt{h_n}\right) G_k^{\alpha}\left(t_n + c_2^i h_n, \widehat{X}^{ki\beta}\right)
\end{aligned}
\tag{8}
$$

In the weak stochastic Runge–Kutta scheme the following random variables are used:

$$
\hat{I}^{kl} = \begin{cases} \frac{1}{2}\left(\hat{I}^{k}\hat{I}^{l} - \sqrt{h_n}\tilde{I}^{k}\right), k<l,\\ \frac{1}{2}\left(\hat{I}^{k}\hat{I}^{l} - \sqrt{h_n}\tilde{I}^{l}\right), l<k,\\ \frac{1}{2}\left(\left(\hat{I}^{k}\right)^2 - h_n\right), k = l, \end{cases}
\tag{9}
$$

The random variable $\hat{I}^k$ has three-points distribution, it can take on three fixed values: $\left\{-\sqrt{3h_n}, 0, \sqrt{3h_n}\right\}$ with probabilities 1/6, 2/3 and 1/6 respectively. The variable $\tilde{I}^k$ has two-points distribution with two values $\left\{-\sqrt{h_n}, \sqrt{h_n}\right\}$ and probabilities 1/2 and 1/2.

4 Sage SDE Module Reference

Our library is a common python module. To connect it to Sage users simply perform a standard command |import sde|.

The library contains a number of functions for internal use. The names of these functions begin with the double bottom underscore, as required by the style rules for python code PEP8. with the double bottom underscore, as required by the style rules for python code PEP8.

- $(dt, t) = \_time(N, interval = (0.0, 1.0))$—function divides the time interval into Nsubintervals and returns numpy array t with step dt;
- $(dW, W) = \_scalar-$ wiener-process(N,dt,seed = None)—function generates a trajectory of a scalar Wiener process W from N subintervals with step dt;
- $(dW, W) = \_multidimensional\_wiener\_process(N, dim, dt, seed = None)$—similar function for generating multidimensional (dim dimensions) Wiener process;
- $(\mathrm{dW}, \mathrm{W}) =$ _cov_multidimensional_wiener _process$(\mathrm{N}, \mathrm{dim}, \mathrm{dt}, \mathrm{seed} = \mathrm{None})$—another function for generating multidimensional (dim dimensions) Wiener process, which uses numpy.multivariate_normal. $\mathrm{W}^1, \mathrm{W}^2, \ldots, \mathrm{W}^{\mathrm{m}}$ processes;
- $(\mathrm{dt}, \mathrm{t}, \mathrm{dW}, \mathrm{W}) =$ wiener_process $(\mathrm{N}, \mathrm{dim} = 1, \mathrm{interval} = (0.0, 1.0), \mathrm{seed} = \mathrm{None})$—the main function which should be used to generate Wiener process. Positional argument N denotes the number of points the interval (default [0,1]). Argument dim defines a dimension of Wiener process. Function returns sequence of four elements $\mathrm{dt}, \mathrm{t}, \mathrm{dW}, \mathrm{W}$, where dt is a step, $(\Delta \mathrm{t_i} = \mathrm{h} = \mathrm{const}), \mathrm{t}$ is one-dimensional numpy-array of time points $\mathrm{t}_1, \mathrm{t}_2, \ldots, \mathrm{t_N}; \mathrm{dW}, \mathrm{W}$ are also $\mathrm{N} \times \mathrm{m}$ dimensional numpy-arrays, consisting of increments $\Delta \mathrm{W}_{\mathrm{i}}^{\alpha}$ and trajectory points $\mathrm{W}_{\mathrm{i}}^{\alpha}$, where $\mathrm{i} = 1, \ldots, \mathrm{N}; \alpha = 1, \ldots, \mathrm{m}$.
- $\_strong\_method\_selector(name)\_weak\_method\_selector(name)$—set Butcher table for specific method.

The following set of functions are made for Itô integrals approximation in strong Runge-Kutta schemas. All of them get an array of increments `dW` as positional argument. Step size `h` is the second argument. All functions return a list of integral approximations for each point of driving process trajectory. For multiple integrals this list contains arrays.

Now we will describe core functions that implement numerical methods with strong convergence.

- EulerMaruyama(f, g, h, x_0, dW)EulerMaruyamaWm(f, g, h, x_0, dW)—Euler Maruyama method for scalar and multidimensional Wiener processes, where f—a drift vector, g—a diffusion matrix;
- strongSRKW1(f, g, h, x_0, dW, name = 'SRK2W1')—function for SDE integration with scalar Wiener process, where f(x) and g(x) are the same as in above functions; h—a step, x_0—an initial value, dW—an array of Wiener process increments N; argument name can take values 'SRK1W1' and 'SRK2W2';
- oldstrongSRKp1Wm(f, G, h, x_0, dW, name = 'SRK1Wm')—stochastic Runge–Kutta method with strong convergence order p = 1.0 for multidimensional Wiener process. This function is left only for performance testing for nested loops realisation of multidimensional arrays convolution;
- strongSRKp1Wm(f, G, h, x_0, dW, name = 'SRK1Wm')—stochastic Runge–Kutta method with strong convergence order p = 1.0 for multidimensional Wiener process. We use NumPy method numpy.einsum because it gives sufficient performance goal.

5 Stochastic RED Model

We use our library to calculate the numerical approximation for stochastic network queue management protocol RED (Random Early Detection) model [10, 11, 13, 17].

RED was first introduced in [12] and there was a lot of in the future original algorithm improvements, particularly ARED algorithms. Basic principles of operation of all modifications of the algorithm RED are very similar, so it makes sense to talk about the RED family-type algorithms. RED model is defined by the following SDE:

$$\begin{cases} dW(t) = \left(\frac{1}{T(t)} - \frac{W^2(t)P\left(\widehat{Q}\right)}{T(t)}\right)dt + \sqrt{\frac{1}{T(t)} + \frac{W^2(t)P\left(\widehat{Q}\right)}{T(t)}}dV^1, \\ dQ(t) = \left(\frac{W(t)}{T(t)} - C\right)dt + \sqrt{\frac{W(t)}{T(t)} - C}dV^2, \\ d\widehat{Q}(t) = w_q C\left(Q(t) - \widehat{Q}(t)\right). \end{cases} \tag{10}$$

There $W(t)$—TCP window size, $Q(t)$—instant queue length, $\widehat{Q}(t)$—an exponentially weighted moving average queue length, $T(t)$—time dual circulation (during the dual circulation come all the confirmation at the TCP sent box), $P(\widehat{Q})$—function calculating dropping probability package, C—the intensity of service, dV^1, dV^2—Wiener processes, the corresponding random processes $W(t)$ and $Q(t)$.

We use weak Runge–Kutha method to calculate exact realizations of SDE numerical solutions. Based on this data we find time evolution of three variables involved in RED SDE model. Thus from Fig. 1 we can see an evolution of RED

queue characteristics over time, also Fig. 2 illustrates the same behaviour for ARED protocol (modification of RED).

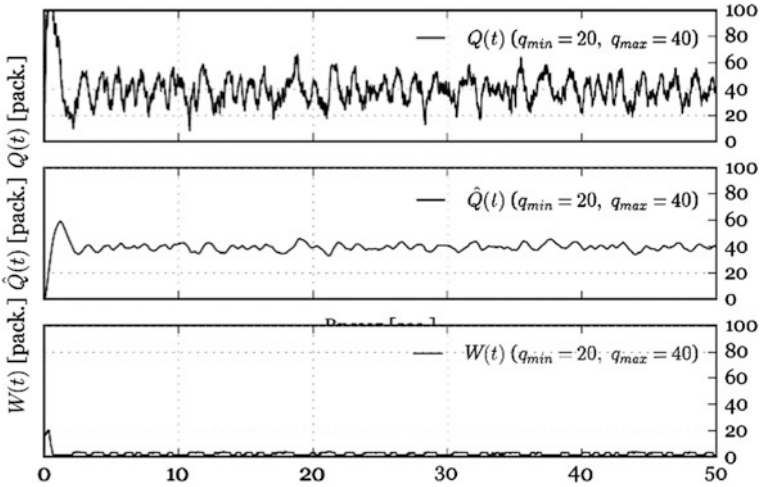

Fig. 1 The solution of SDE for RED protocol

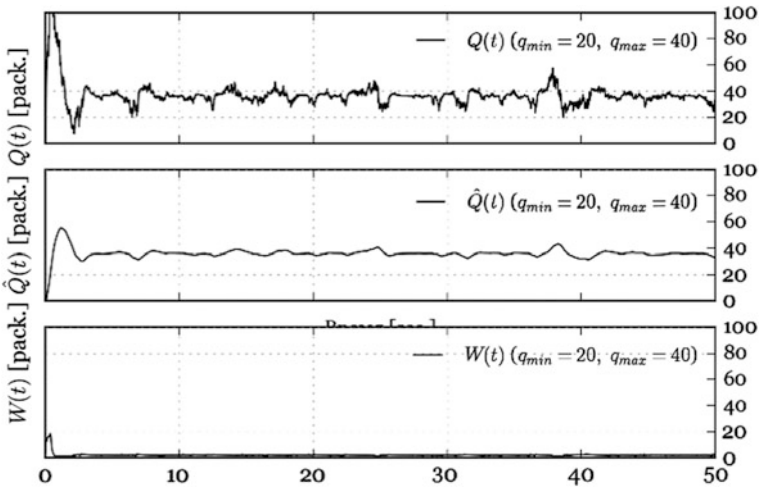

Fig. 2 Solution of SDE for ARED protocol

6 Conclusions

Some realizations of stochastic Runge-Kutta methods were considered in this article. The authors gradually developed and refined the library by adding a new functionalities and optimizing existing ones. To date, the library uses numerical methods by the Rossler's article [7], as the most effective of the currently known to the authors. However, the basic functions strongSRKW1 and weakSRKp2Wm are written in accordance with the general algorithm and can use any Butcher table with appropriate staging. This allows any user of the library to extend functionality by adding new methods.

Additional examples of library usage and of all files source codes are available at https://bitbucket.org/mngev/red-modeling-public. The authors supposed to maintain the library by adding new functions.

Acknowledgments The work is partially supported by RFBR grants No's 14-01-00628, and 15-07-08795, and 16-07-00556.

Calculations were carried out on computer cluster "Felix" (RUDN) and Heterogeneous computer cluster "HybriLIT" (Multifunctional center storage, processing and analysis of data JINR).

References

1. Kloeden, P.E., Platen, E.: Numerical Solution of Stochastic Differential Equations, 2nd edn. Springer, Berlin (1995)
2. Roßler, A.: Runge-Kutta Methods for the Numerical Solution of Stochastic Differential Equations, Ph.D. thesis, Technischen Universit at Darmstadt, Darmstadt (February 2003)
3. Burrage, K., Burrage, P.M.: High strong order explicit Runge-Kutta methods for stochastic ordinary differential equations. Appl. Numer. Math. **22**, 81–101 (1996)
4. Burrage, K., Burrage, P.M., Belward, J.A.: A bound on the maximum strong order of stochastic Runge-Kutta methods for stochastic ordinary differential equations. BIT **37**, 771–780 (1997)
5. Burrage, K., Burrage, P.M.: General order conditions for stochastic Runge-Kutta methods for both commuting and non-commuting stochastic ordinary differential equation systems. Appl. Numer. Math. **28**, 161–177 (1998)
6. Burrage, K., Burrage, P.M.: Order conditions of stochastic Runge-Kutta methods by B-series. SIAM J. Numer. Anal. **38**, 1626–1646 (2000)
7. Roßler, A.: Strong and Weak Approximation Methods for Stochastic Differential Equations—Some Recent Developments, http://www.math.uni-hamburg.de/research/papers/prst/prst2010-02.pdf (2010)
8. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations, 2nd edn. Wiley, New Zealand (2003)
9. Stein, W., et al.: Sage Mathematics Software (Version 6.5), The Sage Development Team, http://www.sagemath.org (2015)
10. Eferina, E.G., Korolkova, A.V., Gevorkyan, M.N., Kulyabov, D.S., Sevastyanov, L.A.: One-step stochastic processes simulation software package, bulletin of peoples' friendship University of Russia. Math. Inf. Sci. Phys. (3), 46–59 (2014). arXiv:1503.07342

11. Velieva, T.R., Korolkova, A.V., Kulyabov, D.S.: Designing installations for verification of the model of active queue management discipline RED in the GNS3. In: 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), IEEE, 2014, pp. 570–577. arXiv:1504.02324, doi:10.1109/ICUMT.2014.7002164
12. Floyd, S., Jacobson, V.: Random early detection gateways for congestion avoidance. IEEE/ACM Trans. Netw. **1**, 397–413 (1993)
13. Hnatič, M., Eferina, E.G., Korolkova, A.V., Kulyabov, D.S., Sevastyanov, L.A.: Operator approach to the master equation for the one-step process. EPJ Web Conf **108**, 02027 (2016). doi:10.1051/epjconf/201610802027
14. Higham, D.J.: An algorithmic introduction to numerical simulation of stochastic differential equations. SIAM Rev. **43**(3), 525–546 (2001)
15. Malham, S.J.A., Wiese, A.: An introduction to SDE simulation (2010). arXiv:1004.0646
16. Platen, E., Bruti-Liberati, N.: Numerical Solution of Stochastic Differential Equations with Jumps in Finance. Springer, Heidelberg (2010)
17. Demidova, A.V., Korolkova, A.V., Kulyabov, D.S., Sevastyanov, L.A.: The method of constructing models of peer to peer protocols. In: 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). IEEE, pp. 557–562 (2014). doi:10.1109/ICUMT.2014.7002162