

## One-Step Stochastic Processes Simulation Software Package

E. G. Eferina, A. V. Korolkova, M. N. Gevorkyan,  
D. S. Kulyabov, L. A. Sevastyanov

*Department of Applied Informatics and Probability Theory  
Peoples' Friendship University of Russia  
6, Miklukho-Maklaya str., Moscow, Russian Federation, 117198*

It is assumed that the introduction of stochastic in mathematical model makes it more adequate. But there is virtually no methods of coordinated (depended on structure of the system) stochastic introduction into deterministic models. Authors have improved the method of stochastic models construction for the class of one-step processes and illustrated by models of population dynamics. Population dynamics was chosen for study because its deterministic models were sufficiently well explored that allows to compare the results with already known ones.

To optimize the models creation as much as possible some routine operations should be automated. In this case, the process of drawing up the model equations can be algorithmized and implemented in the computer algebra system. Furthermore, on the basis of these results a set of programs for numerical experiment can be obtained.

The computer algebra system Axiom is used for analytical calculations implementation. To perform the numerical experiment FORTRAN and Julia languages are used. The Runge–Kutta method for stochastic differential equations is used as numerical method.

The program complex for creating stochastic one-step processes models is constructed. Its application is illustrated by the predator-prey population dynamic system.

Computer algebra systems are very convenient for the purposes of rapid prototyping in mathematical models design and analysis.

**Key words and phrases:** stochastic differential equations; “predator–prey” model; master equation; Fokker–Planck equation; computer algebra software; Axiom system.

### 1. Introduction

This work corresponds our research on mathematical models stochastization. This item is interesting due to the following problems: the construction of population models from first principles and the introduction of the stochastic into such models (the population dynamics is studied because of similar models introduction in other areas).

The problem of stochastic term introduction arises during mathematical models stochastization. There are several ways to solve this problem. The easiest option is an in the deterministic equation. But when additive stochastic term is introduced some free parameters that require further definition appears. Furthermore, these stochastic terms usually interpreted as an external (rather than structural) random impact. In this regard, we used and improved the stochastic one-step processes models construction method, based on master equation [1, 2]. Stochastic differential equation is considered as its approximate form. It allows to get the model equations from general principles. Furthermore, deterministic and stochastic parts are derived from the one equation so we can regard it as stochastic and deterministic parts consistency.

The aim of this work is the software complex development for rapid prototyping construction of stochastic one-step processes models. This complex consists of two blocks. The first block generates the equations of dynamic stochastic process model on the principles similar to chemical kinetic relations describing the investigated process. This block is implemented by means of the computer algebra system — system FriCAS, which is offshoot of Axiom.

The second block is used for the numerical analysis of the resulting model. For numerical solution of deterministic and stochastic models equations some Runge–Kutta different orders methods [3, 4] are used.

To illustrate the developed system the well-known population model predator–prey is used [5–7].

The structure of the paper is as follows. The basic notation and conventions are introduced in Section 2. Section 3 is devoted to brief introduction to the one-step processes stochastization method. Further, in the Section 4 the model under investigation is described. In the subsection 4.1 there is a brief reference to standard (deterministic) approach, and in the subsection 4.2 the stochastic extension of our model with the help of the one-step processes stochastization method is obtained.

In the Section 5.1 we justify selection of the system, which implements the model equations generating unit. The actual interface of this part of the program complex is described in the Section 5.2.

The possibility of applying Runge–Kutta methods for the analysis of stochastic differential equations is considered in the Section 6. The software interface of the model equations numerical analysis unit is also described in this section. Calculations example is based on the predator–prey model.

## 2. Notations and Conventions

1. We use abstract indices notation [8]. In this notation tensor as a whole object is denoted just as an index (e.g.,  $x^i$ ), components are denoted by underlined index (e.g.,  $x^{\underline{i}}$ ).
2. We will adhere to the following agreements. Latin indices of the middle of the alphabet ( $i, j, k$ ) will apply to the space of the system state vectors. Latin indices from the beginning of the alphabet ( $a$ ) will relate to the Wiener process space. Latin indices from the end of the alphabet ( $p, q$ ) will refer to the indices of the Runge–Kutta method. Greek indices ( $\alpha$ ) will set a number of different interactions in kinetic equations.
3. A Dot over a symbol denotes differentiation with respect to time.
4. The comma in the index denotes partial derivative with respect to corresponding coordinate.

## 3. One-Step Processes Modeling

Let's briefly review the method of one-step processes stochastization on the basis of [9].

We understand one-step processes as Markov processes with continuous time with values in the domain of integers, which transition matrix allows only transitions between neighbouring portions. Also, these processes are known as birth-and-death processes.

One-step processes are subject to the following conditions:

1. If at the moment  $t$  the system is in state  $i \in \mathbb{Z}_{\geq 0}$ , then the probability of transition to state  $i + 1$  in time interval  $[t, t + \Delta t]$  is equal to  $k^+ \Delta t + o(\Delta t)$ .
2. If at time moment  $t$  the system is in state  $i \in \mathbb{Z}_+$ , then the probability of transition to state  $i - 1$  in the time interval  $[t, t + \Delta t]$  is equal to  $k^- \Delta t + o(\Delta t)$ .
3. The probability of transition to a state other than the neighbouring is equal to  $o(\Delta t)$ .
4. The probability to remain in the same state is equal to  $1 - (k^+ + k^-) \Delta t + o(\Delta t)$ .
5. State  $i = 0$  is an absorbing boundary.

The idea of the one-step processes stochastization method is as follows. Based on the patterns of interaction we construct a master kinetic equation, expand it into a series, leaving only the terms up to and including the second derivative. The resulting equation is the Fokker–Planck equation. In order to get more convenient model we record corresponding Langevin equation. In fact, as we shall see, from the patterns of interaction we will immediately obtain the coefficients of the Fokker–Planck equation

(and accordingly, the Langevin equation), so for practical use of the method there is no need to construct the master kinetic equation.

### 3.1. Interaction Schemes

We will describe the state of the system by a state vector  $x^i \in \mathbb{R}^n$ , where  $n$  is the system dimension (the state vector is considered as the set of mathematical values, fully describing system). The operator  $n_j^i \in \mathbb{Z}_{\geq 0}^n \times \mathbb{Z}_{\geq 0}^n$  defines the state of the system before the interaction and the operator  $m_j^i \in \mathbb{Z}_{\geq 0}^n \times \mathbb{Z}_{\geq 0}^n$  — after the interaction. The result of interaction is a system transition to another state [1, 10].

There are  $s$  kinds of different interactions that may happen in the system,  $s \in \mathbb{Z}_+$ . So, instead of  $n_j^i$  and  $m_j^i$  let's consider the operators  $n_j^{i\alpha} \in \mathbb{Z}_{\geq 0}^n \times \mathbb{Z}_{\geq 0}^n \times \mathbb{Z}_{\geq 0}^s$  and  $m_j^{i\alpha} \in \mathbb{Z}_{\geq 0}^n \times \mathbb{Z}_{\geq 0}^n \times \mathbb{Z}_{\geq 0}^s$ .

System elements interaction will be described with the interaction schemes similar to chemical kinetic schemes [11]:

$$n_j^{i\alpha} x^j \xrightleftharpoons[k_{\alpha}^{-}]{k_{\alpha}^{+}} m_j^{i\alpha} x^j. \tag{1}$$

Here Greek indices specify the number of interactions and Latin ones specify dimensionality of the system. The state change is given by the operator

$$r_j^{i\alpha} = m_j^{i\alpha} - n_j^{i\alpha}. \tag{2}$$

Thus, one-step interaction  $\alpha$  in forward and opposite directions can be written as

$$\begin{aligned} x^i &\rightarrow x^i + r_j^{i\alpha} x^j, \\ x^i &\rightarrow x^i - r_j^{i\alpha} x^j. \end{aligned} \tag{3}$$

We can write (1) not in the form of vector equations, but in the more traditional form sums:

$$n_j^{i\alpha} x^j \delta_i \xrightleftharpoons[k_{\alpha}^{-}]{k_{\alpha}^{+}} m_j^{i\alpha} x^j \delta_i, \tag{4}$$

where  $\delta_i = (1, \dots, 1)$ .

Also, we will use the following notation:

$$n^{i\alpha} := n_j^{i\alpha} \delta^j, \quad m^{i\alpha} := m_j^{i\alpha} \delta^j, \quad r^{i\alpha} := r_j^{i\alpha} \delta^j. \tag{5}$$

### 3.2. Master Equation

Transition probabilities per unit of time from the state  $x^i$  to the state  $x^i + r_j^{i\alpha} x^j$  (to the state  $x^i - r_j^{i\alpha} x^j$ ) are proportional to the number of  $x^i$  combination from a set of  $n^{i\alpha}$  elements (of  $x^i$  — combinations from a set of  $m^{i\alpha}$ ) and are given by:

$$\begin{aligned} s_{\alpha}^{+} &= k_{\alpha}^{+} \prod_{i=1}^n \frac{x^i!}{(x^i - n^{i\alpha})!}, \\ s_{\alpha}^{-} &= k_{\alpha}^{-} \prod_{i=1}^n \frac{x^i!}{(x^i - m^{i\alpha})!}. \end{aligned} \tag{6}$$

Thus, the general form of the master kinetic equation for the states vector  $x^i$  (it changes by  $r_j^{i\alpha} x^j$  per step), takes the form:

$$\frac{\partial p(x^i, t)}{\partial t} = \sum_{\alpha=1}^m \left\{ \left[ s_{\alpha}^{-}(x^i + r^{i\alpha}, t) p(x^i + r^{i\alpha}, t) - s_{\alpha}^{+}(x^i) p(x^i, t) \right] + \left[ s_{\alpha}^{+}(x^i - r^{i\alpha}, t) p(x^i - r^{i\alpha}, t) - s_{\alpha}^{-}(x^i) p(x^i, t) \right] \right\}. \quad (7)$$

### 3.3. Fokker–Planck Equation

With the help of the Kramers–Moyal expansion, the Fokker–Planck equation [11] is obtained. For this purpose we will make several assumptions:

- 1) there are only small jumps, ie  $s_{\alpha}(x^i)$  is a slowly varying function with the change of  $x^i$ ;
- 2)  $p(x^i, t)$  also slowly changes with the change of  $x^i$ .

Then in Fokker–Planck equation (7) one can shift from the point  $(x^i \pm r_j^{i\alpha} x^j)$  to the point  $x^i$ , and by expanding the right-hand side in a Taylor series and dropping terms of order higher than the second, we obtain Fokker–Planck equation:

$$\frac{\partial p}{\partial t} = -\partial_i [A^i p] + \frac{1}{2} \partial_i \partial_j [B^{ij} p], \quad (8)$$

where

$$\begin{aligned} A^i &:= A^i(x^k, t) = r^{i\alpha} [s_{\alpha}^{+} - s_{\alpha}^{-}], \\ B^{ij} &:= B^{ij}(x^k, t) = r^{i\alpha} r^{j\alpha} [s_{\alpha}^{+} - s_{\alpha}^{-}], \quad \alpha = \overline{1, m}. \end{aligned} \quad (9)$$

As seen from (9), the coefficients of the Fokker–Planck equation can be obtained directly from (2) and (6), i.e. in practical calculations, there is no need to write the master equation.

### 3.4. Langevin Equation

The Langevin equation corresponds to the Fokker–Planck equation:

$$dx^i = a^i dt + b_a^i dW^a, \quad (10)$$

where  $a^i := a^i(x^k, t)$ ,  $b_a^i := b_a^i(x^k, t)$ ,  $x^i \in \mathbb{R}^n$  — is the system state vector,  $W^a \in \mathbb{R}^m$  —  $m$ -dimensional Wiener process. Wiener process is implemented as  $dW = \varepsilon \sqrt{dt}$ , where  $\varepsilon \sim N(0, 1)$  is normal distribution with average 0 and variance 1. Latin indices from the middle of the alphabet denote the values related to the state vectors (dimension of the space is  $n$ ), and Latin indices from the beginning of the alphabet denote the values related to the Wiener process vector (dimension of the space is  $m \leq n$ ).

The connection between the equation (8) and (10) expressed by the following relationships:

$$A^i = a^i, \quad B^{ij} = b_a^i b^{ja}. \quad (11)$$

We will use Ito interpretation. Under the Ito interpretation, differential of complex functions does not obey the standard formulas of analysis. To calculate it *rule* or *Ito lemma* are used.

Let  $f := f(x^k, t)$  is a function of a random process  $x^k(t)$ ,  $f \in C^2$ . Then the formula of the differential is [12]:

$$df = \left[ \partial_t f + a^i f_{,i} + \frac{1}{2} b_a^i b^{ia} f_{,ij} \right] dt + b_a^i f_{,i} dW^a, \quad (12)$$

where  $f := f(x^k, t)$ ,  $a^i := a^i(x^k, t)$ ,  $b_a^i := b_a^i(x^k, t)$ , and  $dW^a := dW^a(t)$ .

## 4. Predator–Prey Model

### 4.1. Deterministic Predator–Prey Model

Systems with the interaction of two predator-prey populations types are extensively studied and there are a lot of various models for these systems. The very first predator-prey model is considered to be a model which was obtained independently by A. Lotka and V. Volterra. Lotka in [13] described some hypothetical chemical reaction:



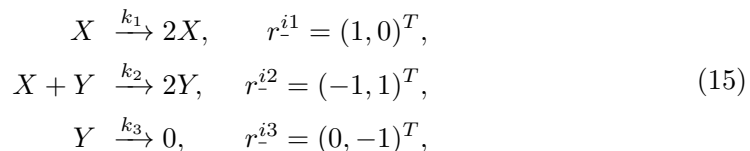
where  $X, Y$  are intermediates substances, coefficients  $k_1, k_2, k_3$  are rates of chemical reactions,  $A$  is a initial reagent, and  $B$  is a resultant. As a result was a system of differential equations:

$$\begin{cases} \dot{x} = k_1 x - k_2 xy, \\ \dot{y} = k_2 xy - k_3 y. \end{cases} \quad (14)$$

This system is identical to the system of differential equations, obtained by Volterra, who considered the growth mechanism of two populations with predator–prey interaction type. In order to get equations [5] Volterra made a series of idealized assumptions about nature of intraspecific and interspecific relationships in the predator–prey system.

### 4.2. Stochastic Predator–Prey Model

Consider a model of predator–prey system, consisting of two individuals species, one of which hunts, second is provided with inexhaustible food resources. Let's introduce the notation, where  $X$  is a prey and  $Y$  is a predator, then we can write the possible processes (4) for the state vector  $x^i = (X, Y)^T$  [14–17]:



which have the following interpretation. The first relation means that the prey which eats the food unit immediately reproduced. The second relation describes the case when predator absorbs the prey and then it is instantaneously reproduced. Only such possibility of prey death is considered. Last ratio is a natural predator death.

All processes are irreversible, so  $s_{\alpha}^{-} = 0$ , and

$$\begin{aligned} s_1^+(x, y) &= k_1 \frac{x!}{(x-1)!} \frac{y!}{y!} = k_1 x, \\ s_2^+(x, y) &= k_2 \frac{x!}{(x-1)!} \frac{y!}{(y-1)!} = k_2 xy, \\ s_3^+(x, y) &= k_3 \frac{x!}{x!} \frac{y!}{(y-1)!} = k_3 y. \end{aligned} \quad (16)$$

With the help of the formula (8) we have the Fokker–Planck equation:

$$\frac{\partial p(x, y)}{\partial t} = -\partial_i (A^i(x, y)p(x, y)) + \frac{1}{2} \partial_i \partial_j (B^{ij}(x, y)p(x, y)), \quad (17)$$

where

$$\begin{aligned} A^i(x, y) &= s_{\alpha}^+(x, y) r^{i\alpha}, \\ B^{ij}(x, y) &= s_{\alpha}^+(x, y) r^{i\alpha} r^{j\alpha}. \end{aligned} \quad (18)$$

As a result:

$$\begin{aligned} A^i(x, y) &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} k_1 x + \begin{pmatrix} -1 \\ 1 \end{pmatrix} k_2 xy + \\ &+ \begin{pmatrix} 0 \\ -1 \end{pmatrix} k_3 y = \begin{pmatrix} k_1 x - k_2 xy \\ k_2 xy - k_3 y \end{pmatrix}, \\ B^{ij}(x, y) &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} (1, 0) k_1 x + \begin{pmatrix} -1 \\ 1 \end{pmatrix} (-1, 1) k_2 xy + \\ &+ \begin{pmatrix} 0 \\ -1 \end{pmatrix} (0, -1) k_3 y = \begin{pmatrix} k_1 x + k_2 xy & -k_2 xy \\ -k_2 xy & k_2 xy + k_3 y \end{pmatrix}. \end{aligned} \quad (19)$$

In order to write a stochastic differential equation in Langevin form (10) for predator–prey model, it is enough to take the square root of the resulting matrix  $B^{ij}$  in Fokker–Planck equation

$$\begin{aligned} d \begin{pmatrix} x \\ y \end{pmatrix} &= \begin{pmatrix} k_1 x - k_2 xy \\ k_2 xy - k_3 y \end{pmatrix} dt + b_a^i \begin{pmatrix} dW^1 \\ dW^2 \end{pmatrix}, \\ b_a^i b_a^j &= B^{ij} = \begin{pmatrix} k_1 x + k_2 xy & -k_2 xy \\ -k_2 xy & k_2 xy + k_3 y \end{pmatrix}. \end{aligned} \quad (20)$$

It should be noted that the specific form of the matrix  $b_a^i$  is not written out because of the extreme awkwardness of the expression. However, with further studies we will need not actually matrix  $b_a^i$ , but its square, i.e. the matrix  $B^{ij}$ .

## 5. Implementation of the One-Step Stochastic Processes Model in the Computer Algebra System

### 5.1. Justification of the Computer Algebra System Choice

Let's consider systems of analytical calculations, Maxima and Axiom. Maxima is the first system of analytical calculations and it is written in Lisp. Maxima successfully runs on all modern operating systems: Windows, Linux and UNIX, Mac OS and even on PDA running Windows CE/Mobile. Documentation is integrated into the program as a handbook with search. There is no distinction between objects and data

in Maxima, and there is no clear distinction between the operator and function. There is no integrated graphics rendering in the system.

Unlike Maxima Axiom language is strongly typed for better mathematical objects and relationships display. The mathematical basis is written in Spad language. Axiom portability is slightly worse: the system runs under Linux, UNIX, and graphs does not work under Windows. Axiom has its own graphics subsystem.

In 2007 two Axiom open source forks appeared: OpenAxiom and FriCAS. Open Axiom is developed by adhering to the ideology of Axiom, problems that occurred in the Axiom are eliminated. FriCAS developers reorganized the assembly process, expanded functionality. Furthermore, FriCAS supports not only GCL, which operates on limited number of platforms, but ECL, Clisp, sbcl or openmcl, that allows to run FriCAS under wider range of platforms.

## 5.2. Implementation Description in the Axiom Computer Algebra System

Method of one-step processes randomization is organized as a module for the FriCAS computer algebra system. To display all the calculations on the screen the variable `SHOWCALC:=true` is used. To call the method you need to use the main function, which has the following view:

```
osp(Matrix(Integer), Matrix(Integer), Vector, Vector, Vector)
```

where the first argument is before interaction states matrix  $n_j^i$ , the second argument is after interaction states matrix  $m_j^i$ , the third argument is the vector  $k_\alpha^+$ , the fourth argument is the vector  $k_\alpha^-$ , the fifth argument is the state vector  $x^i$ . Let's consider the features of the language FriCAS on auxiliary functions. For example, the function `calcProd` is used to simplify the calculations  $s_\alpha^+$  and  $s_\alpha^-$ . In the implementation of the function operator of the condition and built-in function `reduce` are used:

```
calcProd : (Matrix(Integer), Vector, Integer, Integer) -> Void
calcProd (n, x, a, i) ==
  nai:Integer := n(a,i)
  if nai = 0 then 1 else reduce(*,[x(i) - j for j in 0..(nai-1)])
```

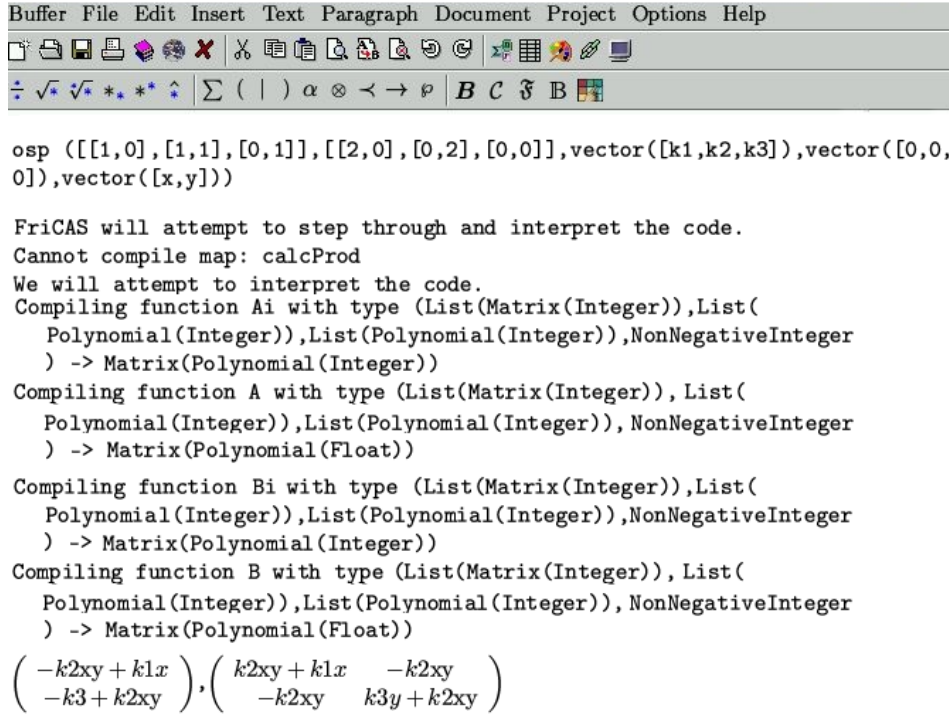
In the function `Bi` intermediate calculations for elements of the matrix  $B^{ij}$  are made:

```
Bi (rv, sp, sm, i) == rv(i) * (transpose rv(i)) * (sp(i) + sm(i))
```

In order to use the module for predator-prey system model, we call the function with the following arguments:

```
osp ([[1,0], [1,1], [0,1]], [[2,0], [0,2], [0,0]],
      vector([k1,k2,k3]), vector([0,0,0]),vector([x,y]))
```

Fig. 1 represents the result obtained in  $\text{\TeX}$ macs shell. In fact, we repeated the results obtained in (19).



```

osp ([[1,0],[1,1],[0,1]],[[2,0],[0,2],[0,0]],vector([k1,k2,k3]),vector([0,0,0]),vector([x,y]))

FricAS will attempt to step through and interpret the code.
Cannot compile map: calcProd
We will attempt to interpret the code.
Compiling function Ai with type (List(Matrix(Integer)),List(
  Polynomial(Integer)),List(Polynomial(Integer)),NonNegativeInteger
) -> Matrix(Polynomial(Integer))
Compiling function A with type (List(Matrix(Integer)),List(
  Polynomial(Integer)),List(Polynomial(Integer)), NonNegativeInteger
) -> Matrix(Polynomial(Float))
Compiling function Bi with type (List(Matrix(Integer)),List(
  Polynomial(Integer)),List(Polynomial(Integer)),NonNegativeInteger
) -> Matrix(Polynomial(Integer))
Compiling function B with type (List(Matrix(Integer)),List(
  Polynomial(Integer)),List(Polynomial(Integer)), NonNegativeInteger
) -> Matrix(Polynomial(Float))

( -k2xy + k1x ) ( k2xy + k1x  -k2xy )
( -k3 + k2xy ) ( -k2xy   k3y + k2xy )

```

Figure 1. The output of the module for predator–prey model in graphic  $\text{\TeX}$ macs shell

## 6. Numerical Experiment for the Program Complex

### 6.1. Stochastic Runge–Kutta Methods

Euler–Maruyama method is one of well-known numerical methods for solving SDE, it is a special case of a more general Stochastic Runge–Kutta method. Classical Runge–Kutta method can be generalized to the case of the SDE system (10) in the following manner [3, 4]:

$$\begin{cases} X_k^i = x_0^i + hR_k^l a^i(X_l^1, \dots, X_l^n) + \hat{R}_k^l J^\alpha b_\alpha^i(X_l^1, \dots, X_l^n), \\ x_1^i = x_0^i + hr^l a^i(X_l^1, \dots, X_l^n) + \hat{r}^l J^\alpha b_\alpha^i(X_l^1, \dots, X_l^n). \end{cases} \quad (21)$$

Indexes  $k = 1, \dots, s$  and  $l = 1, \dots, n$  refer to stochastic Runge–Kutta method.  $J \sim N(0, h)$  or  $J \sim \sqrt{h}\varepsilon$ ,  $\varepsilon \sim N(0, 1)$  are normal distributed random variables. Such a choice of these numerical values for approximation is made because the Wiener process is implemented as  $dW = \varepsilon\sqrt{dt}$ . You should also pay attention to double summation in the third term of both numerical scheme formulas as well as the fact that each number  $J^1, \dots, J^n$  should be generated separately.

The method coefficients, as well as for the classical analogue, can be grouped into a table called the *Butcher table*:

$$\begin{array}{c|c} R_{ij} & \hat{R}_{ij} \\ \hline r_j & \hat{r}_j \end{array}.$$



For calculations we used a method with the table

0	0	0	0	0	0
2/3	0	0	2/3	0	0
-1	1	0	-1	1	0
0	3/4	1/4	0	3/4	1/4

## 6.2. Software Implementation Description

The purposes of the programs complex were to automate the SDE coefficients  $A^i$  and  $B^{ij}$  computation with the help of general principles described above, and to find a numerical solution of the equation obtained by means of stochastic Runge–Kutta methods. From a programming standpoint we can derive three subtasks:

1. coefficients  $A^i$  and  $B^{ij}$  generation using the computer algebra system;
2. generation of source code in languages Fortran and Julia, implementing the SDE on the basis of the coefficients, saved as a text file;
3. writing subroutines/functions implementing stochastic Runge–Kutta methods in Fortran and Julia, and their subsequent compilation together with automatically generated source codes.

As a result of its work Axiom module creates a text file which contains the coefficients  $A^i$  and  $B^{ij}$  in the following form:

```
# A
A[1]
...
A[N]
# B
B[1,1] B[1,2] .. B[1,N]
...
B[N,1] B[N,2] .. B[N,N]
```

Matrix  $b_\alpha^i = \sqrt{b_\alpha^i b^{j\alpha}} = \sqrt{B^{ij}}$  is calculated numerically with the help of the singular value matrix decomposition (a subroutine DGESVD from library LAPACK is used).

For the second subtask scripting language Python was chosen (version 3). This language has a wide set of tools to work with strings and text files. Except matrices  $A^i$  and  $B^{ij}$  additional information about the mathematical model was specified as dictionary (standard data type in Python), with model name, list of variables, list of parameters, initial values of variables, parameters values and parameters of the numerical method (integration section and step size).

On the basis of these data, the script automatically generates two files `functions.f90` and `main.f90`, where the first is a module with functions defining the SDE, and the second one is a main program file. While compiling these files the third additional module with auxiliary procedures with Stochastic Runge–Kutta method is added.

## 6.3. The Numerical Experiment Description

For the programs complex work verification a well-known predator–prey model was chosen with vector  $a^i$  components

$$a^1 = \alpha x - \beta xy, \quad a^2 = -\gamma y + \delta xy \quad (22)$$

and matrix  $B^{ij}$ :

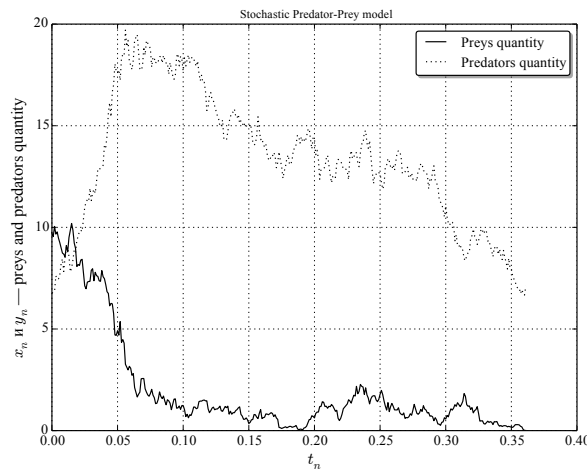
$$\begin{bmatrix} \alpha x + \beta xy & -\beta xy \\ -\beta xy & \beta xy + \gamma x \end{bmatrix}, \quad (23)$$

$x$  is the number of preys,  $y$  is the number of predators. Coefficients also have the following physical (biological) meaning:  $\alpha$  is the growth rate of the prey population,  $\beta$

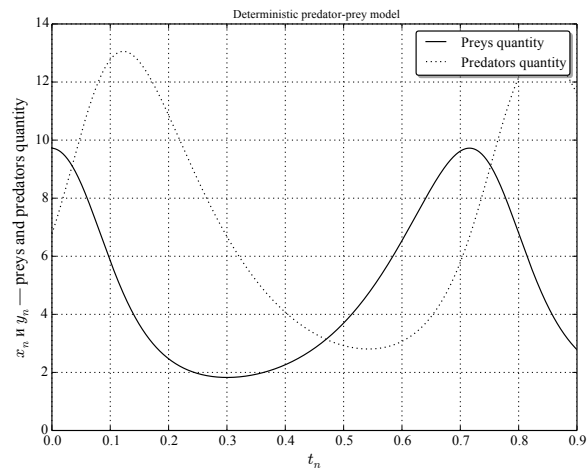
is a frequency of predators and prey meetings,  $\gamma$  is an intensity of predators death or migration in a lack of preys,  $\delta$  is a predator population growth rate on the assumption of the excess of the prey.

During numerical simulations it was taken into account that the value of variables  $x, y$  could not be less than zero (program stop working when one of the variables becomes equal to zero).

Numerical simulation shows that the addition of stochastic to the classical predator–prey model leads to the fact that after a certain time death of one of the competing species comes. So, for the following parameters:  $\alpha = 10$ ,  $\beta = 1.5$ ,  $\gamma = 8.5$ ,  $\delta = 1.8$  and the initial values:  $x = 9.7$ ,  $x = 6.77$ , victims are first to die, and after that predators die due to lack. This case is illustrated in Fig. 2. For comparison in Fig. 3 is a graph for the deterministic case.



**Figure 2.** Stochastic predator–prey model, prey die



**Figure 3.** Deterministic predator–prey model

Under other conditions ( $\alpha = 10$ ,  $\beta = 1.5$ ,  $\gamma = 8.5$ ,  $\delta = 0.5$ ,  $x = 22$ ,  $y = 6.76$ ) predators die, and the number of victims is increasing rapidly, as for their model assumes an infinite source of food. Graphics for this case are shown in Fig. 4, and Fig. 5 shows for comparison with deterministic case.

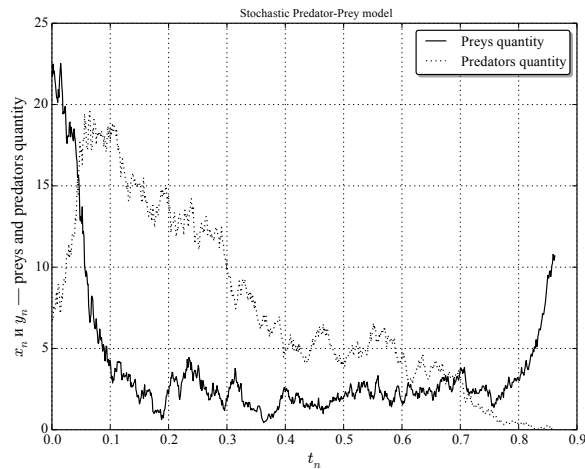


Figure 4. Stochastic predator–prey model, predators die

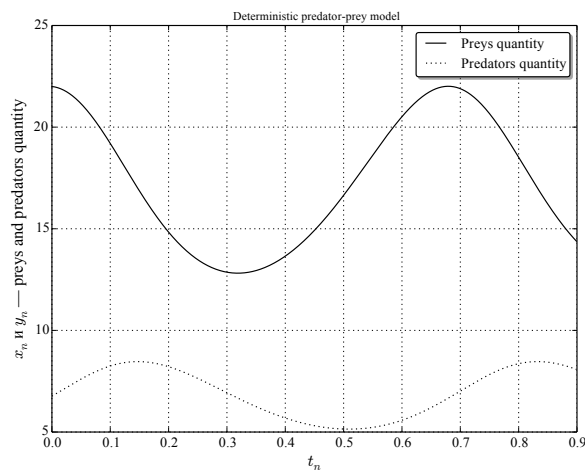


Figure 5. Deterministic predator–prey model

## 7. Conclusions

This work demonstrates the application of the developed initial physical system formalization method. The system is presented in the form of one or more one-step processes. Formalization of the system is done by introducing the evolution operator. Wherein the analytical description of the model requires a lot of routine operations. To simplify the work we propose to use the computer algebra system (Axiom fork FriCAS).

We have developed an analytical software package block that receives inlet evolution operator and produces the SDE, which describes the original model. For numerical studies of obtained SDE system a second software unit that converts the resulting system of equations into the program code in Fortran and gives its numerical solution was developed. Thus, the software system is applicable for both analytical and numerical study of the original model.

Currently the software package does not cover all possibilities, incorporated in the proposed method of formalizing the original physical system. Since the original system description uses ODE, we should introduce the boundary conditions by ties

or indicator functions. Partial differential equations can help to solve this problem. Further objective is the development of a complete software complex for a method of one-step original physical system model construction.

## References

1. A. V. Demidova, D. S. Kulyabov, The Introduction of an Agreed Term in the Equation of Stochastic Population Model, *Bulletin of Peoples Friendship University of Russia. Series "Mathematics. Information Sciences. Physics"* (3) (2012) 69–78, in Russian.
2. A. V. Demidova, L. A. Sevastyanov, D. S. Kulyabov, Application of Stochastic Differential Equations to Model Population Systems, in: *Third International Conference on Mathematical Modelling of Social and Economical Dynamics MMSED-2010*, Russian State Social University, 2010, pp. 92–94, in Russian.
3. K. Debrabant, A. Röbler, Classification of Stochastic Runge–Kutta Methods for the Weak Approximation of Stochastic Differential Equations, *Mathematics and Computers in Simulation* 77 (4) (2008) 408–420. doi:<http://dx.doi.org/10.1016/j.matcom.2007.04.016>.
4. A. Tocino, R. Ardanuy, Runge–Kutta Methods for Numerical Solution of Stochastic Differential Equations, *Journal of Computational and Applied Mathematics* 138 (2) (2002) 219–241. doi:[http://dx.doi.org/10.1016/S0377-0427\(01\)00380-6](http://dx.doi.org/10.1016/S0377-0427(01)00380-6).  
URL <http://www.sciencedirect.com/science/article/pii/S0377042701003806>
5. V. Volterra, *Mathematical Theory of the Struggle for Existence*, Nauka, 1976.
6. A. D. Bazykin, *Nonlinear Dynamics of Interacting Populations*, World Scientific, 1998.
7. A. S. Bratus, A. S. Novozhilov, A. P. Platonov, *Dynamical Systems and Biology Models*, Fizmatlit, M., 2010, in Russian.
8. Roger Penrose and Wolfgang Rindler, *Spinors and Space-Time: Two-Spinor Calculus and Relativistic Fields*, Vol. 1, Cambridge University Press, 1984.
9. A. V. Demidova, M. N. Gevorkyan, A. D. Egorov, A. V. Korolkova, D. S. Kulyabov, L. A. Sevastyanov, Influence of Stochastization to One-Step Model, *Bulletin of Peoples' Friendship University of Russia. Series "Mathematics. Information Sciences. Physics"* (1) (2014) 71–85, in Russian.
10. A. V. Demidova, A. V. Korolkova, D. S. Kulyabov, L. A. Sevastyanov, The Method of Stochastization of One-Step Processes, in: *Mathematical Modeling and Computational Physics*, JINR, 2013, p. 67.
11. C. W. Gardiner, *Handbook of Stochastic Methods: for Physics, Chemistry and the Natural Sciences*, Springer Series in Synergetics, 1985.
12. B. K. Øksendal, *Stochastic Differential Equations: An Introduction with Applications*, Springer, Berlin, 2003.
13. A. J. Lotka, *Elements of Physical Biology*, BiblioBazaar, 2011.  
URL <http://books.google.ru/books?id=tFN9pwAACAAJ>
14. A. V. Demidova, The Equations of Population Dynamics in the Form of Stochastic Differential Equations, *Bulletin of Peoples' Friendship University of Russia. Series "Mathematics. Information Sciences. Physics"* (1) (2013) 67–76, in Russian.
15. A. V. Demidova, The Method of Stochastization of Mathematical Models for the Example of the "Predator–Prey", in: *A Scientific Session NRNU MEPHI-2013*, 2013, p. 127, in Russian.
16. A. V. Korolkova, D. S. Kulyabov, Methods of Stochastization of Mathematical Models on the Example of Peer to Peer Networks, in: *A Scientific Session NRNU MEPHI-2013*, MEPHI, Moscow, 2013, p. 131, in Russian.
17. A. V. Demidova, D. S. Kulyabov, L. A. Sevastyanov, The Agreed Stochastic Term in Population Models, in: *XI Belarusian Mathematical Conference*, Institute of Mathematics of the National Academy of Sciences of Belarus, Minsk, 2012, p. 39, in Russian.

УДК 004.94, 519.21

**Программный комплекс стохастического моделирования  
одношаговых процессов****Е. Г. Ефери́на, А. В. Коро́лькова, М. Н. Геворкян,  
Д. С. Кулябов, Л. А. Севастьянов***Кафедра прикладной информатики и теории вероятностей  
Российский университет дружбы народов  
ул. Миклухо-Маклая, д. 6, Москва, Россия, 117198*

Нашим коллективом разработана методика согласованного (зависящего от структуры системы) введения стохастичности в детерминистические модели. На данном этапе методика ограничена классом одношаговых процессов.

Для оптимизации работы по созданию моделей следует автоматизировать как можно больше рутинных операций. В данном случае процесс составления уравнений модели можно алгоритмизировать и реализовать в системе компьютерной алгебры. Кроме того, на базе этих результатов можно получить и набор программ для проведения численного эксперимента.

Для реализации аналитических расчётов используется система компьютерной алгебры Axiom. Для проведения численного эксперимента используются язык FORTRAN и Julia. В качестве численного метода используется метод Рунге–Кутты для стохастических дифференциальных уравнений.

Разработан программный комплекс для создания стохастических моделей одношаговых процессов. Проиллюстрировано его применение на примере системы популяционной динамики типа «хищник–жертва». Детерминистические модели для таких процессов достаточно хорошо исследованы, что позволяет сравнить полученные результаты с уже известными.

Системы компьютерной алгебры очень удобны для целей быстрого прототипирования при создании и исследовании математических моделей.

**Ключевые слова:** стохастические дифференциальные уравнения; модель «хищник–жертва»; основное кинетическое уравнение; уравнение Фоккера–Планка; системы компьютерной алгебры; система Axiom.

**Литература**

1. Кулябов Д. С., Демидова А. В. Введение согласованного стохастического члена в уравнение модели роста популяций // Вестник РУДН. Серия «Математика. Информатика. Физика». — 2012. — № 3. — С. 69–78.
2. Demidova A. V., Sevastianov L. A., Kulyabov D. S. Application of Stochastic Differential Equations to Model Population Systems // Труды Третьей Международной конференции «Математическое моделирование социальной и экономической динамики (MMSSED-2010)» / Российский государственный социальный университет. — 2010. — С. 92–94.
3. Debrabant K., Röbler A. Classification of Stochastic Runge–Kutta Methods for the Weak Approximation of Stochastic Differential Equations // *Mathematics and Computers in Simulation*. — 2008. — Vol. 77, No 4. — Pp. 408–420. — ISSN 0378-4754.
4. Tocino A., Ardanuy R. Runge–Kutta Methods for Numerical Solution of Stochastic Differential Equations // *Journal of Computational and Applied Mathematics*. — 2002. — Vol. 138, No 2. — Pp. 219–241. — ISSN 0377-0427. — <http://www.sciencedirect.com/science/article/pii/S0377042701003806>.
5. Volterra V. *Mathematical Theory of the Struggle for Existence*. — Nauka, 1976.
6. Базыкин А. Д. *Нелинейная динамика взаимодействующих популяций*. — Москва-Ижевск: Институт компьютерных исследований, 2003.
7. Братусь А. С., Новожиллов А. С., Платонов А. П. *Динамические системы и модели биологии*. — М.: Физматлит, 2010.
8. Пенроуз Р., Рундлер В. *Спиноры и пространство-время. Два-спинорное исчисление и релятивистские поля*. — М.: Мир, 1987. — Т. 1, 528 с.

9. Влияние стохастизации на одношаговые модели / Анастасия Вячеславовна Демидова, Мигран Нельсонович Геворкян, Александр Дмитриевич Егоров и др. // Вестник РУДН. Серия «Математика. Информатика. Физика». — 2014. — № 1. — С. 71–85.
10. The Method of Stochastization of One-Step Processes / A. V. Demidova, A. V. Korolkova, D. S. Kulyabov, L. A. Sevastianov // Mathematical Modeling and Computational Physics. — JINR, 2013. — P. 67.
11. *Гардинер К. В.* Стохастические методы в естественных науках. — Мир, 1986.
12. *Øksendal B. K.* Stochastic Differential Equations: An Introduction with Applications. — Berlin: Springer, 2003.
13. *Lotka A. J.* Elements of Physical Biology. — BiblioBazaar, 2011. — ISBN 9781178508116, 492 p. — <http://books.google.ru/books?id=tFN9pwAACAАJ>.
14. Демидова А. В. Уравнения динамики популяций в форме стохастических дифференциальных уравнений // Вестник РУДН. Серия «Математика. Информатика. Физика». — 2013. — № 1. — С. 67–76.
15. Демидова А. В. Метод стохастизации математических моделей на примере системы «хищник–жертва» // Научная сессия НИЯУ МИФИ-2013. — 2013. — С. 127.
16. Королькова А. В., Кулябов Д. С. Методы стохастизации математических моделей на примере пиринговых сетей // Научная сессия НИЯУ МИФИ-2013. Аннотации докладов. В 3 томах. — Москва: МИФИ, 2013. — С. 131.
17. Демидова А. В., Кулябов Д. С., Севастьянов Л. А. Согласованный стохастический член в популяционных моделях // XI Белорусская математическая конференция. — Минск: Институт математики НАН Беларуси, 2012. — С. 39.