Compound Program Packages in Random Science Training and Technical Modelling

I.E.Poloskov

Perm State University, Bukirev st. 15, Perm, GSP, 614600, Russia,
E-mail: Igor.Poloskov@psu.ru

Abstract

The present paper is presenting a review of our applications of computer algebra systems (CAS) [10, 11, 40] to different problems: for analysis of random phenomena in nonlinear dynamical systems and primary education in this sphere; for finding of control functions and for mathematical modelling of a mechanical objects movement. To archive final results in various tasks it is required to process a large number of mathematical calculations. Moreover the complexity of investigated objects requires to automate a procedure of model construction that results in the necessity of CAS using (together with numerical accounts) at all stages of scientific research.

1 Introduction

The problems of probabilistic research of processes in nonlinear dynamic systems concern the number of major tasks both in the theoretical and practical sense. The necessity of their solution is urgent for studying various phenomena: vehicles flight under atmospheric turbulence action; traffic on a rough surface; high–altitude structures vibrations in the wind and seismic attacks; ships' rollings on the roughsea; technological processes of production; deviations of space vehicles orbit elements from estimated ones arising due to production inaccuracies of rockets–carriers and errors of control systems; energy networks loads fluctuations depending on consumption; noises of amplifiers in regulation and monitoring systems; noises in radio and electronic devices; unpredictability of demand in economic systems etc.

To analyze these complex and scale phenomena in nonlinear systems called stochastic or random the statistical dynamics is engaged. Such systems permit to describe an operation of real technical objects in which parameters are random processes and/or variables.

When solving a significant number of practical problems it is possible to assume that a random vector process $\vec{x} \in \mathbf{X} \subset \mathbf{R^n}$ describing a status of an object being researched (a phase vector) satisfies stochastic differential equations (SDEqs.) set in the Itô or Stratonovich sense [5, 8, 29, 33, 35, 36, 37] (we use the last)

$$\dot{\vec{x}}(t) = \vec{f}(\vec{x}, t) + G(\vec{x}, t)\vec{\xi}(t) \tag{1.1}$$

or, in a more precise form

$$d\vec{x}(t) = \vec{f}(\vec{x}, t)dt + G(\vec{x}, t)d\vec{w}(t) \tag{1.2}$$

where $\dot{\vec{w}} = \vec{\xi} \in \mathbf{R^m}$ is a vector of independent Gaussian white noises with unit intensities; $\vec{w} \in \mathbf{R^m}$ is the standard Wiener vector process with zero mathematical expectation and identity variances matrix; $\vec{f}(\cdot,\cdot) = (f_i(\cdot,\cdot))^T : \mathbf{R^n} \times [0,\infty) \to \mathbf{R^n}$ and $G(\cdot,\cdot) = (g_{ij}(\cdot,\cdot)) : \mathbf{R^n} \times [0,\infty) \to \mathbf{R^n} \times \mathbf{R^m}$ are deterministic vector and matrix; T is a symbol of the transposition. Assuming this the vector process $\vec{x}(t)$ (the solution of the Eqs. (1.1), (1.2)) will be the continuous Markov process which is also called the diffusion process.

Main probabilistic characteristics of the vector $\vec{x}$ are the probability density $p(\vec{x}, t)$, the transition density $p(\vec{x}, t \mid \vec{y}, \tau)$, moments $m_\alpha = \mathbf{M}[x^\alpha]$ (including the first ones: mean values or mathematical expectations m_i, variances σ_i^2 and mixed moments of the second order m_{ij}, cumulants (semi–invariants) κ_α, covariance functions

$$K_{ij}(t_1, t_2) = \mathbf{M}[(x_i(t_1) - m_i(t_1))(x_j(t_2) - m_j(t_2))]$$

(if $t_1 = t_2$ then $K_{ij} = D_{ij}(t)$ where D_{ij} are elements of a matrix of variances), spectral densities. The probability $\mathcal{P}(\vec{x} \in \Omega)$ of the vector $\vec{x}$ presence in the given area Ω (here and further $\mathbf{M}$ is a symbol of mathematical expectation, α is multiindex) is also interesting.

It is possible to show that if all elements of the vector

$$\vec{a}(\vec{x},t) = (a_1(\vec{x},t), a_2(\vec{x},t), ..., a_n(\vec{x},t))^T$$

where

$$a_i(\vec{x},t) = f_i(\vec{x},t) + \frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{m}\frac{\partial g_{ik}(\vec{x},t)}{\partial x_j}g_{jk}(\vec{x},t)$$

are differentiable and all elements of the matrix $G(\vec{x},t)$ are twice differentiable then the transition density $p(\vec{x},t \mid \vec{y},\tau)$ satisfies the Fokker-Planck-Kolmogorov equation (FPK Eq.)

$$\frac{\partial p(\vec{x},t \mid \vec{y},\tau)}{\partial t} = \mathcal{L}_{xt}p(\vec{x},t \mid \vec{y},\tau) \tag{1.3}$$

with the initial condition

$$\lim_{t\to\tau+0} p(\vec{x},t \mid \vec{y},\tau) = \delta(\vec{x}-\vec{y})$$

The Eq.(1.3) is also called the second or forward Kolmogorov Eq., Fokker-Planck Eq., Einstein-Fokker Eq., diffusion Eq. etc. The operator $\mathcal{L}$ within the Eq.(1.3) is of the form

$$\mathcal{L}_{xt}v = -\sum_{i=1}^{n}\frac{\partial}{\partial x_i}[a_i(\vec{x},t)v] + \frac{1}{2}\sum_{i,j=1}^{n}\frac{\partial^2}{\partial x_i \partial x_j}[b_{ij}(\vec{x},t)v] \tag{1.4}$$

where $a_i(\vec{x},t)$ are called the drift or kinetic coefficients and the diffusion coefficients $b_{ij}(\vec{x},t)$ are the elements of the non-negatively determined diffusion matrix

$$B(\vec{x},t) = G(\vec{x},t)G^T(\vec{x},t).$$

The distribution density $p(\vec{x},t)$ also satisfies Eq.(1.3) but with the initial condition

$$p(\vec{x},t_0) = p_0(\vec{x}).$$

Let t tends to infinity then under certain conditions (the time-independence of the drift and diffusion coefficients and others [5, 8, 33, 36] there exists the solution (the stationary, or steady-state, or equilibrium probability density) $p_s(\vec{x})$ of the stationary FPK Eq.

$$\mathcal{L}p_s(\vec{x}) = 0, \quad \vec{a} = \vec{a}(\vec{x}), \quad B = B(\vec{x}) \tag{1.5}$$

which does not depend on the initial distribution.

In practice the various forms of the FPK Eqs. are used. So in the case of random noises absence the Eq.(1.3) takes the form

$$\frac{\partial p(\vec{x},t \mid \vec{y},\tau)}{\partial t} = -\sum_{i=1}^{n}\frac{\partial}{\partial x_i}[a_i(\vec{x},t)p(\vec{x},t \mid \vec{y},\tau)] \tag{1.3$'$}$$

(the case of a system subjected by random parameters only is reduced to the Eq.(1.3′) too). It exists more simple form of the Eqs. (1.3), (1.4) as follows

$$\frac{\partial p}{\partial t} = \frac{1}{2}\sum_{i,j=1}^{n}\mathsf{A}_{ij}\frac{\partial^2 p}{\partial x_i \partial x_j} + \sum_{i=1}^{n}\mathsf{B}_i\frac{\partial p}{\partial x_i} + \mathsf{C}\,p \tag{1.3$''$}$$

where

$$\mathsf{A}_{ij} = b_{ij}, \qquad \mathsf{B}_i = -a_i + \sum_{j=1}^{n}\frac{\partial b_{ij}}{\partial x_j},$$

$$\mathsf{C} = \sum_{i=1}^{n}\frac{\partial}{\partial x_i}\left(-a_i + \frac{1}{2}\sum_{j=1}^{n}\frac{\partial b_{ij}}{\partial x_j}\right).$$

Next, it is possible to record the Eq. for the natural logarithm of the probability density (equal to the stochastic potential except for sign)

$$\frac{\partial \ln p}{\partial t} = \frac{1}{2}\sum_{i,j=1}^{n} \mathsf{A}_{ij}\left(\frac{\partial^2 \ln p}{\partial x_i \partial x_j} + \frac{\partial \ln p}{\partial x_i}\frac{\partial \ln p}{\partial x_j}\right) +$$

$$+ \sum_{i=1}^{n} \mathsf{B}_i \frac{\partial \ln p}{\partial x_i} + \mathsf{C}. \qquad (1.3''')$$

The characteristic function $\varphi(\vec{\lambda}, t)$, $\vec{\lambda} \in \mathbf{\Lambda} \subset \mathbf{R^n}$ gives as much information about the process $\vec{x}(t)$ as the probability density $p(\vec{x}, t)$ and satisfies the Pugachev integro–differential Eq.:

$$\frac{\partial \varphi}{\partial t} = \frac{1}{(2\pi)^n} \int\limits_{\mathbf{X}} \int\limits_{\mathbf{\Lambda}} e^{i(\vec{\lambda}-\vec{\mu})^T \vec{x}} S(\vec{x}, \vec{\lambda}, t)\varphi(\vec{\mu}, t) d\vec{x} d\vec{\mu}$$

with the initial condition

$$\varphi(\vec{\lambda}, t_0) = \varphi_0(\vec{\lambda}) = \int\limits_{\mathbf{X}} e^{i\vec{\lambda}^T \vec{x}} p_0(\vec{x}) d\vec{x}$$

where

$$S(\vec{x}, \vec{\lambda}, t) = i\vec{\lambda}^T \vec{a}(\vec{x}, t) - \frac{1}{2}\vec{\lambda}^T B(\vec{x}, t)\vec{\lambda}.$$

Constructing mathematical models of modern technical objects it is necessary to take into account their complexity, an existence of many interrelations, influence of various deterministic and random parameters. Therefore the procedure of such objects motion Eqs. deriving can not be practically executed without a computer aid. And here CAS can be helpful. In Section 2 the main principles of the package VMPack structure for the CAS Mathematica environment intended for the facilitation of the researcher's manipulations with vector-matrix expressions including the solution of an estimation problem of random parameters influence on nonlinear systems dynamics are considered.

Nowadays there is a significant number of exact and approximate methods intended for solving of the SDEqs., FPK Eqs. and the Pugachev integro–differential Eqs. [2, 5, 7, 8, 22, 23, 29, 33, 34, 35, 36, 37]. But to perform most of them time consuming mathematical calculations are required (transformations, reductions of similars, differentiations, integrations etc.). The advantages of CAS Formac, Reduce, Mathematica applications to this sphere are considered in Section 3.

The availability of a large number of diverse methods is a matter of urgency for the students and young researchers studying the statistical dynamics algorithms. Section 4 is devoted to the description of the primary education system "Statistical Dynamics" for creation of which the CAS Mathematica was mainly used.

2 The CAS applications for complex technical objects dynamics analysis

Nowadays a significant development of computers facilities has substantially contributed to the analysis and control of complex technical designs with many degrees of freedom. The main method of such analysis is a mathematical modelling coupling an objects mathematical model as a DEqs. set building and a computer research of the object fundamental properties after suitable software development.

To build of compound systems models (such as aircraft, spacecraft, vehicles and so forth) one must take account of many different units and their complex interactions. Moreover in the drawing–board stage some tolerances defined by technology demands must be set up on values of developed design basic characteristics (masses, inertia tensors, stiffnesses and others). Therefore numerical computations of an object behavior with the specific parameters values can not bring a full process description. Here a problem arises to estimate an effect on the object motion of a spread within tolerance of deterministic and random engineering and structure parameters.

An exact and complete solution of this problem is connected with a calculation of a multidimensional joint probability density of phase and parameters vectors but it is often impossible due to computer capabilities limitations. Therefore the most well–known simple and effective line of results to reach is the sensitivity theory application [38]. The main task here is a derivation of coupled DEqs. set consisting of the original object motion DEqs. and of DEqs. for random sensitivity functions. If the object has many degrees of freedom and is affected by plenty of random parameters then the above method is impossible to implement without computer processing.

The research difficulties do not only consist of the sensitivity theory DEqs. derivation but of numerical analysis. The point is that the coupled DEqs. set is usually nonlinear. Therefore the approximate stochastic dynamics methods are necessary for statistical characteristics calculation. *The integrator method* (see Section 3) is one of them.

Below the program performing an automation of system dynamics Eqs. building is described. The same program automates some stages of preparation for the numerical decision of the estimation problem of random parameters influence on a complex nonlinear stochastic system in terms of the requirements of the integrator method which, in our opinion, is applicable at a stage of the probabilistic analysis of the random sensitivity functions. It may be used for a linearization ordinary DEqs. (ODEqs.) set procedure too.

The problem of the automation process software creation for the estimation of random parameters influence on complex objects arose already long time ago. Thus the essential part of evaluation techniques is differentiation of the motion DEqs. r.h.s. w.r.t. these parameters.

It is possible to solve this problem manually for small quantities of phase coordinates and parameters. If their number is great than a performance of a large amount of analytical calculations becomes a long, tedious and non-faultless work. This procedure is frequently complicated by the necessity of vector-matrix expressions processing.

On the other hand the software for numerical calculations of deterministic object motion is often created before random analysis software, and high level programming languages (such as Fortran) are used for this purpose. In this case it sometimes does not make sense to begin a stage of differentiation from Eqs. but it is possible to start from ready-make programs in such languages.

To overcome these difficulties a Formac program package [21] has been designed allowing partial derivatives of the r.h.s. of a sequence of algebraic equalities to be found, the equalities written in a Fortran-like manner (particularly, it may be a Fortran program source text to compute the r.h.s. of the set of dynamics DEqs.) in terms of a list of independent variables and to present the differentiation result as a Fortran program package. Among the equalities (formula statements) there may be subroutine call statements (CALL), unmanipulated conditional statements, commentaries and others.

The input of the program consisted of the following:

– identifiers of independent (for example, phase coordinates and random parameters) and target variables;

– the name of the derivatives array;

– the root of the names of the generated subroutines (for example, PRAV00, PRAV01, PRAV02, ...);

– information on the called subroutines (about quantity and names of input and output variables, names of partial derivatives array) et al.

The program algorithm is to read the next text element in a sequence (up to its end) from a set of data (for example, Fortran statements), to find the elements type and the presence of independent variables and aggregates (left–hand side identifiers mentioned before), to differentiate in terms of the arguments list in the case of algebraic equalities, to fill in information tables with left–hand sides and input parameters relations (including the relations through aggregates).

Below Windows Mathematica 3.0 project [28] meant to solve the above mentioned problem is considered.

The programs complex VMPack is made as a problem-oriented package constructed in terms of requirements of the Mathematica environment and consists of the main procedure VMMain and a number of bottom level procedures.

The package VMPack is intended:

– to input vector–matrix (and scalar) expressions containing standard operations of vector–matrix algebra and recorded in the form resembling an ordinary mathematical text;

– to recognize and to transform vector-matrix expressions to an interior representation;

– to take partial derivatives up to the needed power (in vector–matrix and coordinate forms) w.r.t. some formal variables;

– to transfer to an external file initial and resulting expressions in the form of TurboPascal statements such as calls of appropriate procedures to calculate vectors coordinates and matrices elements are generated (that is connected with a sharp growth of the resulting Pascal program text length).

Under differentiation all expressions objects are considered as complicated functions of some number of arguments (for example, random parameters) which were called above as formal arguments. Owing to the differentiation rule of a compound function however many such variables are the structure of the derivatives of any order remains the same.

Therefore to find derivatives up to some order from any sequence of expressions w.r.t. a set of any number of parameters it is enough to differentiate once the expressions for determination of all first derivatives, the result of the previous differentiation to get all second derivatives and so on and then to change only the numbers of parameters in the received expressions instead of derivatives calculating for every possible combination of one, two or more parameters.

Input for the package has the following form:

```
vector /the list of names/;                                    (2.1)
matrix /the list of names/;                                    (2.2)
scalar /the list of names/;                                    (2.3)
constant /the list of names/;                                  (2.4)
derivatives order is /integer/;                                (2.5)
formulae;                                                      (2.6)
.........................;
.........................;
.........................;
end of formulae;                                               (2.7)
```

where the first line describes the names of 3x1-vectors; the second – 3x3-matrices; the third – scalars; the line (2.4) – constants. The integer in the line (2.5) can have the value from 0 up to 3 and defines the order of the senior derivative to be found. The lines (2.1)–(2.4) can follow in any order.

In input between lines (2.6) and (2.7) any correct strings with vectors, matrices and scalars operations can be mixed. Their type is distinguished at processing under the descriptions (2.1)–(2.4). During the package work expressions are checked for correctness.

In the package the processing of the following vector–matrix operations is implemented:

– sum of vectors and matrices (an input designation is "+");

– difference of vectors and matrices ("-");

– scalar product ("<...,...>");

– vector product ("[...,...]");

– diad (tensor) product ("{...,...}");

– multiplication of scalars, vectors and matrices ("*");

– inverse of matrix ("^");

– transposition of matrix ("'");

– summation w.r.t. index (see below).

Frequently in deriving motion DEqs. the summation sign w.r.t. some index in given limits is used, for example, in the expression for the main vector of acting forces. To represent such a procedure the summation operation is introduced. Thus the sum

$$\sum_{i=k_1}^{k_2} f_i$$

in the source text takes the form

$$\$(\texttt{fi}, \texttt{k1}, \texttt{k2})\$.$$

As the result of the software package work in external files the following is formed:

– the result of conversion of vector–matrix expressions to the internal form and result of their differentiation given number of times;

– a sequence of TurboPascal procedures calls which calculate the above-stated vector-matrix expressions.

Among these calls there may be the following: `P_SumV`, `P_SumM`, `P_Scal`, `P_Vect`, `P_Diad`, `P_MultSV`, `P_MultSM`, `P_MultMV`, `P_MultMM`, `P_Inv`, `P_VecZ`, `P_MatZ` (designations are evident and agree with appropriate operations, a symbol "Z" shows that it is needed to clear an object to zero).

The derivatives names for the objects which we have come across are formed as follows:

$$< \text{Object name} >\$XYZ$$

where \$ is a special symbol for derivatives and symbols X, Y, Z depend on the senior order N of derivatives.

The identifiers of temporary variables are generated by the CAS Mathematica means on the basis "wx0xw" for real and "i$wv" for integer variables.

Separated lists of objects used (in free form) can be found at the text end.

An example of package using is in Appendix 1.

As a package development I am going to build two subpackages. The first one must visualize input formulae in the LaTeX form (for a supplementary checking) and the second must manage data bases of models in the Visual FoxPro style (see the description of package "Statistical Dynamics").

3 Analysis of stochastic nonlinear systems and CAS

As it was remarked above there is a vast number of exact and approximate methods for research of the complex phenomena in nonlinear dynamic systems effected by random noises. Among them it is possible to notice:

– *exact methods* (the principle of detailed balance, the potential method, the groups theory, the variables substitution method, the decomposition method and others);

– *methods of a task simplification* (exchange of variables etc.);

– *linearization methods* (direct, harmonic, equivalent, statistical and others);

– *numerical methods* (the Monte Carlo method, the random walks method, the cells mapping method, the integrated method, the interpolating method, the sensitivity theory method, methods of finite differences, the integrator method et al.);

– *methods of integral transforms* (Fourier, Laplace and so on);

– *methods of expansions into a series* (in terms of orthogonal functions, eigenfunctions et al.);

– *variation methods* (Bubnov–Galerkin, maximum of entropy etc.);

– *perturbations methods* (small parameter, stochastic averaging, the multiple time scales method, Krylov-Bogoliubov asymptotic method et al.);

– *iterative schemes* (successive approximations, the Krasovsky iterations method, the method of a functional corrections averaging, the iteration operator method, the method of oscillating functions, the parametrix method, an application of semigroups of operators, the method of functional parameters, a Runge–Kutta scheme for the FPK Eq., the Adomian decomposition method, the method of a conditional density, the path integral method and others);

– *methods reducing to an ordinary DEq.* set (a density expansion into a series in terms of the Hermite functions or other special functions, the usage of quasi–moments, the moments method, the semi–invariants method, the moments–semi-invariants method, the method of the orthogonal expansion, the heat polynomials method, the Gaussian approximation method, the canonical expansions method, the power series method, the expansion into a proper functions series method etc.);

– *methods of integral Eqs.* (an expansion into a Volterra functional series, a Hazen technique etc.).

But at present there is no universal algorithm suitable for solving a significant part of arising problems. It results in constant filling up a list of random analysis methods with new algorithms. But both the new and the known methods of such analysis are very difficult to use.

Attempts to automate a process of stochastic systems research have been undertaken for a long time as it takes place in the deterministic case [6, 9, 24]. But due to technical complexities methods enabling one to find statistical characteristics of a phase vector are basically realized. As such projects it is possible to notice the realizations of the moments–semi-invariants method ($n \leq 6$) [3, 4], the normal approximation method [30], the various updatings of the semi–invariants method [12] and the moments method [31] et al.

CAS applications permit to expand a methods list accessible for automated application, to reduce temporary costs when inputting data about an object under research, to work with a mathematical model in a symbolic form, to derive Eqs. in each particular case that can result in essential decreasing of time expenses at the stage of numerical calculations etc. While the developments with CAS usage are mainly intended for solving of model and training problems [32] but there are some works solving deeper probabilistic problems for example symbolic Itô calculus realizations [13, 14, 39]. There are the CAS Reduce, Formac, Mathematica aided implementations of the iteration method [18], of the modified power series method [25], of the method of probability density expansion into a series in terms of the Hermite functions [25], of the algorithms of the stochastic sensitivity theory [21] etc. An essential part of these projects is to automate the deriving of appropriate finite parts for infinite ODEqs. sets right-hand sides (r.h.s.).

Main features of statistical dynamics methods realization on the basis of the CAS Formac, Reduce, Mathematica are considered below.

The principle of detailed balance [8] is one of the most powerful methods of an exact search of steady-state densities in nonlinear systems.

To apply this principle the following identities should be true

$$\varepsilon_i a_i(\vec{x}_\varepsilon) p_s(\vec{x}) = -a_i(\vec{x}) p_s(\vec{x}) + \sum_{j=1}^{n} \frac{\partial}{\partial x_j}[b_{ij}(\vec{x}) p_s(\vec{x})],$$

$$\varepsilon_i \varepsilon_j b_{ij}(\vec{x}_\varepsilon) = b_{ij}(\vec{x}), \qquad i,j = 1,2,...n. \tag{3.1}$$

Here $p_s(\vec{x})$ is the probability steady-state density, $\varepsilon_i = \pm 1$ depend on the evenness of a variable x_i (for example displacements are considered to be even variables and linear velocities are odd ones); $x_{\varepsilon i} = \varepsilon_i x_i$.

Thus it appears that such checking has mainly a routine character and it permits to automate fulfillment on the computer with the help of CAS for a concrete n at least.

To realize this idea a CAS Mathematica software program was created the algorithm of which is following:

– to input symbolical representations of SDEq. r.h.s.;

– to calculate the drift and diffusion coefficients;

– to check a fulfillment of the equality (3.1);

– to analyze a diffusion tensor degeneracy;

– in the case of its nondegeneracy to build and to solve a linear algebraic Eqs. (LAEqs.) set for partial derivatives of a stochastic potential $\phi = -\ln p_s(\vec{x})$ with respect to (w.r.t.) variables x_i ($\frac{\partial \phi}{\partial x_i} = \phi_i, i = 1,2,...,n$) and then to find the stochastic potential itself with a quadrature if the equalities $\frac{\partial \phi_i}{\partial x_j} = \frac{\partial \phi_j}{\partial x_i}$ are true;

– otherwise an attempt of the allocation and solution of a linearly independent part of such a set for some ϕ_{k_s}, $s = 1,2,...,r$. If appropriate conditions are satisfied the function $\tilde{\phi}(\vec{x}')$ is calculated with a quadrature where $\vec{x}' = (x_{k_1}, x_{k_2}, ..., x_{k_r})^T \in X'(\phi_{k_s} = \phi_{k_s}(\vec{x}'))$. Thus the stochastic potential $\phi(\vec{x})$ will have the form

$$\phi(\vec{x}) = \tilde{\phi}(\vec{x}') + \phi_0(\vec{x}''), \qquad \vec{x}'' \in X'', \qquad R^n = X' \otimes X''.$$

Then an Eq. for the function $\phi_0(\vec{x}'')$ is constructed starting from the Eq. for $\phi(\vec{x})$. If the above conditions are satisfied but $\phi_{k_s} = \phi_{k_s}(\vec{x}', \vec{x}'')$ then an Eq. for $\phi(\vec{x})$ in terms of the expressions for ϕ_{k_s} is derived.

With the help of this program the stochastic potentials for the following systems

$$\dot{x}_i + k_{i1}x_1 + k_{i2}x_2 + \mu_i x_i^3 = \sqrt{g_i}\xi_i, \quad i = 1,2; \tag{3.2}$$

$$\dot{x}_i + \sum_{j=1}^{n} k_{ij}x_j + \mu_i x_i^3 = \sqrt{g_i}\xi_i, \quad i = \overline{1,n};$$

$$\dot{x}_i + c_{i30}x_1^3 + c_{i21}x_1^2 x_2 + c_{i12}x_1 x_2^2 + c_{i03}x_2^3 = \sqrt{g_i}\xi_i, \quad i = 1,2;$$

$$\ddot{x}_i + 2\alpha_i\dot{x}_i + \omega_i^2 x_i + \mu_i x_i^3 = k_i x_{3-i} + \sqrt{g_i}\xi_i, \quad i = 1,2; \tag{3.3}$$

$$\ddot{x}_i + 2\alpha_i\dot{x}_i + \omega_i^2 x_i + \mu_i x_i^3 = \sum_{j=1, j\neq i}^{n} k_{ij}x_j + \sqrt{g_i}\xi_i, \quad i = \overline{1,n}$$

have been considered.

Examples of the steady-state density surfaces for the system (3.2) are presented in Appendix 2.

For the research of nonlinear systems with random parameters and/or initial conditions the *method of Eqs. infinite sets* [26] can be applied. According to it a source DEqs. set by variables substitutions is reduced to an infinite set of linear ODEqs. for a vector with an accounting number of components the probability density of which can be exactly constructed after closure.

For facilitation of a given method using for an analysis of the system

$$\dot{x} + x + x^2 = 0, \qquad x(0) = x_0 \tag{3.4}$$

where x_0 is random a CAS Mathematica code program was created. The tasks of this program are following: to build an ODEq. set for a vector

$$\vec{z} = (z_1, z_2, ..., z_N)^T \equiv (x, x^2, ..., x^N)^T$$

on the basis of the Eq.(3.4) (in the calculations the value $N = 19$ was used); to solve exactly this set relative to $\vec{z}$ with symbolical initial conditions $\vec{z}_0 = (z_{01}, z_{02}, ..., z_{0N})^T \equiv (x_0, x_0^2, ..., x_0^N)^T$. Further linear identities connecting vectors $\vec{z}$ and $\vec{z}_0$ are eliminated relative to $\vec{z}_0$ that permits to construct a density expression. Integrating over coordinates z_2, z_3, ..., z_N and using properties of δ-functions presenting in an expression for $p_N(\vec{z}_0)$ one can receive an approximation of the x density. Thus owing to the necessity to solve an algebraic N-th degree Eq. which has coefficients depending on t and to select the only real roots values the required density was calculated in knots of a given (x,t) grid.

The phenomena in systems subjected by continuous and discrete random excitations and usually described by white noises and Poisson processes are interesting. The study of such systems responses results in the necessity of the Kolmogorov-Feller integro-differential Eq. [37] solving.

Even for the very simple form of this Eq. such as:

$$\frac{\partial p(x,t)}{\partial t} = \frac{g}{2}\frac{\partial^2 p(x,t)}{\partial x^2} + \frac{\partial}{\partial x}[cxp(x,t)] - \nu p(x,t) + \\ + \frac{\nu}{c}\int_{-\infty}^{+\infty} \mathcal{K}(x - x')p(x',t)dx'$$

one cannot obtain its solution exactly. An approximate analysis algorithm is as follows. To calculate a finite number of moments it was solved a ODEq. set satisfied by them and then the density was expanded into a series in terms of quasi–moments.

To build moments analytical expressions as functions of t and to calculate quasi–moments due to complicated identities a Mathematica code program was created and that factor has allowed to receive density approximations for various values of parameters g, c, ν and forms of a nucleus $\mathcal{K}$. An example of this scheme realization is in Appendix 3.

If transition processes in the system (1.2) have been finished it can change its status to the one specified by the steady-state probability density $p_s(\vec{x})$. Therefore the problem of this density finding arises.

Next we shall consider *the variation iteration algorithm* [21] for $p_s(\vec{x})$ computation. Let assume that a nonlinear object is described by the following Eqs.

$$d\vec{x}(t) = [A_0\vec{x}(t) + \vec{f}^0(\vec{x}(t))]dt + B_0 d\vec{w}(t) \tag{3.5}$$

where A_0 and B_0 are constant matrices, $f_i^0 = \sum\limits_{|\alpha|=2}^{N} f_{i\alpha}^0 x^\alpha$. We shall find an approximation $\tilde{p}_s(\vec{x})$ of the Eq.(1.5) solution as

$$\tilde{p}_s(\vec{x}) = \sum_{k=0}^{M} c_k h_k(\vec{x}) \tag{3.6}$$

with a linear independent functions system $(h_k(\vec{x}))$, $k = 0, 1, 2, ...$ constructed through iterations

$$h_k(\vec{x}) = \mathcal{L}h_{k-1}(\vec{x}) \qquad (k > 0) \tag{3.7}$$

in terms of the function $h_0(\vec{x})$ which is the steady–state density of the linear system

$$d\vec{y}(t) = A_0\vec{y}(t)dt + B_0 d\vec{w}(t)$$

corresponding to (3.5). This density is known to be a multidimensional Gaussian one. The coefficients c_k are found as solutions of a LAEqs. set of the Galerkin method:

$$\int\limits_{R^n} h_q(\vec{x})\mathcal{L}p_s(\vec{x})d\vec{x} = 0$$

or

$$\sum_{r=0}^{M} c_r \int\limits_{R^n} h_q(\vec{x})h_{r+1}(\vec{x})d\vec{x} = 0, \qquad q = 1, 2, ..., M \tag{3.8}$$

($c_0 = 1$ due to the density normalization condition).

Using Formac and Fortran-IV programs which implement the above method the steady-state density approximation $\tilde{p}_s(x)$ of the system

$$\dot{x} + cx + \mu x^3 = \sqrt{g}\xi \tag{3.9}$$

displacement x has been formed. As for the system (3.2) a program for the Mathematica environment was created. To analyze the system with two degrees of freedom such as Eq.(3.3) the algorithm (3.6)–(3.8) has been implemented in terms of the Reduce and Fortran77 code. In all these cases the CAS using allowed to build functions $h_k(\vec{x})$ (3.7) symbolically.

One of the simplest FPK Eq. solving procedures is the following. Let us write Eq.(1.3) in the form:

$$p = p_0 + \int\limits_0^t \mathcal{L}_\tau p d\tau \equiv p_0 + \mathcal{U}\mathcal{L}p.$$

The iteration operator method [16, 20] is to construct the functions

$$p_k = \mathcal{U}\mathcal{L}p_{k-1}, \qquad p_0 = p_0(\vec{x})$$

and to present a density approximation as

$$\tilde{p}_M = \sum_{k=0}^{M} p_k.$$

The analysis of complex nonlinear systems by the given method results in numerous multivariable functions differentiation and integration operations which in general do not allow the study of the above systems to be completed. The method can simply and economically be implemented to study stochastic systems with a polynomial structure and time–independent functions f_i and g_{ij}.

Using a Formac program the above method has been applied to investigate various systems and to approximate the probability density in a problem of a rigid body rotating around a fixed point and excited by external random torques [21]

$$J_1\dot{p} - (J_2 - J_3)qr = \sqrt{g_1}\xi_1,$$
$$J_2\dot{q} - (J_3 - J_1)rp = \sqrt{g_2}\xi_2,$$
$$J_3\dot{r} - (J_1 - J_2)pq = \sqrt{g_3}\xi_3.$$

Here J_1, J_2, J_3 are inertia moments; p, q, r are body angular velocity projections on constrained coordinate system axes; g_1, g_2, g_3 are constants.

For time homogeneous Markov systems it is interesting to find the transition probability density $\pi(\vec{x},\vec{y},t) = p(\vec{x},s+t|\vec{y},s)$ the knowledge of which allows to calculate such statistical characteristics as a covariance function, the number of process intersections of a given level et al. But the Dirac multi–delta functions being present in the initial condition

$$\pi(\vec{x},\vec{y},0) = \delta(\vec{x}-\vec{y})$$

make it difficult to determine $\pi(\vec{x},\vec{y},t)$.

One can overcome the arising difficulties by writing the transition density π in the following form (*the method of a formal representation*) [21]

$$\pi(\vec{x},\vec{y},t) = \sum_{\alpha=0}^{\infty} \pi_\alpha(\vec{y},t)\delta^{(\alpha)}(\vec{x}-\vec{y}).$$

Assuming the FPK Eq. coefficients to be presented as

$$A_{ij} = \sum_{\beta=0}^{\infty} A_{ij\beta}\frac{z^\beta}{\beta!}, \quad B_i = \sum_{\beta=0}^{\infty} B_{i\beta}\frac{z^\beta}{\beta!}, \quad C = \sum_{\beta=0}^{\infty} C_\beta\frac{z^\beta}{\beta!}$$

($\vec{z} = \vec{x} - \vec{y}$) the following infinite ODEq. set has been formed to determine the coefficients $\pi_\alpha(\vec{y},t)$

$$\frac{\partial \pi_\alpha}{\partial t} = V_\alpha(\pi), \qquad \alpha \geq 0, \qquad \pi_\alpha(\vec{y},0) = \delta_{\alpha 0} \tag{3.10}$$

where

$$V_\alpha = \sum_{\beta=0}^{\infty}(-1)^\beta\left[\frac{1}{2}\sum_{i=1}^{n} A_{ii\beta}\pi_{\nu-2e_i} + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} A_{ij\beta}\pi_{\nu-e_{ij}} + \right.$$
$$\left. + \sum_{i=1}^{n} B_{i\beta}\pi_{\nu-e_i} + C_\beta\pi_\nu\right], \qquad \nu = \alpha - \beta. \tag{3.11}$$

The derivation of the r.h.s. V_α for particular SDEqs. represents a reasonably tiresome process essentially becoming complicated with a growth of a problem dimension. Here CAS application may facilitate researcher's efforts. So a Mathematica code program generates an expression for a function $V_{\alpha_1\alpha_2}$ in a general form for systems with polynomial nonlinearities (in this case such expression contains a finite number of terms) that permits if necessary to organize a calculation of a r.h.s. sequence by substitution of indexes particular values.

The similar scheme was realized to approximate the probability density $p(x_1,x_2,t)$ due to an expansion into a series in terms of the Hermite functions.

Notice that in Eqs.(3.10), (3.11) the vector $\vec{y}$ enters as parameter that also hinders an application of the considered method. However a CAS application (especially in a combination with the small parameter method) permits to be advanced further in a necessary direction too. So for a system of the type (3.9) where $c = 1$, $\mu = \varepsilon$ is a small parameter the functions $\pi_1(y,t)$ and $\pi_2(y,t)$ were determined accurate to ε^5 that has allowed to make out approximate identities for $a_x(t)$, $\sigma_x^2(t)$ and $K_x(t)$ in an analytical form.

One of the best known and well–developed techniques for vibrations in quasi–linear mechanical systems treating is *the Krylov–Bogoliubov averaging method* [1] which has been extended to solve SDEqs. [22, 34, 36]. In terms of the method for stochastic systems of the form

$$\ddot{x}_i + \omega_i^2 x_i = \varepsilon f_i(\vec{x}, \dot{\vec{x}}, t) + \sqrt{\varepsilon} \sum_{j=1}^{m} g_{ij}(\vec{x}, \dot{\vec{x}}, t)\xi_j(t), \quad i = \overline{1, n}$$

where ω_i are eigenfrequences, ε is a small parameter variables are changed as

$$x_i = a_i(t)\cos(\omega_i t + \theta_i(t)), \quad \dot{x}_i = -\omega_i a_i(t)\sin(\omega_i t + \theta_i(t)) \tag{3.12}$$

and then SDEqs. for slowly varying amplitudes a_i and phases ω_i are derived. The Eqs. obtained are averaged over a period w.r.t. an explicit time variable that frequently results in uncoupling of the new variables a_i and ω_i and therefore in a simplified problem. Notice that such averaging may be done in FPK Eq. too. This method requires a great amount of analytical manipulations. But the implementation of the averaging method by means of CAS offer no difficulties.

In complicated cases the variables substitution may be different (3.12). As for the SDEqs. set

$$\begin{aligned} \ddot{x}_1 + 2\varepsilon\alpha_1\dot{x}_1 + \omega_1^2 x_1 + \varepsilon\mu x_1^3 &= k x_2, \\ \ddot{x}_2 + 2\varepsilon\alpha_2\dot{x}_2 + \omega_2^2 x_2 &= \sqrt{\varepsilon g}\xi(t) \end{aligned} \tag{3.13}$$

the change

$$\begin{aligned} x_1 &= a_1\cos\psi_1 + \frac{k}{\omega_1^2 - \omega_2^2} a_2\cos\psi_2, \\ x_2 &= a_2\cos\psi_2, \\ \dot{x}_1 &= -\omega_1 a_1\sin\psi_1 - \omega_2\frac{k}{\omega_1^2 - \omega_2^2} a_2\sin\psi_2, \\ \dot{x}_2 &= -\omega_2 a_2\sin\psi_2, \\ \psi_i &= \omega_i t + \theta_i, \qquad i = 1, 2, \qquad \omega_1 \neq \omega_2 \end{aligned}$$

is more convenient. For the system (3.13) analysis there has been developed a Mathematical software program that was doing all the necessary manipulations such as the Sides. deriving for phases a_1, a_2 and amplitudes θ_1, θ_2; FPK Eq. building governing the probability density $p(a_1, a_2, \theta_1, \theta_2, t)$; its averaging and writing of resulting Eq.

It is natural that by CAS using statistical dynamics problems cannot be solved from the beginning up to the end. Therefore analytical and numerical parts of algorithms have to combine.

Examples of such combinations are the following (see Section 4 too).

One of the methods enabling one to find the first moments of the phase vector is *the integrator method* [17, 27].

The method is intended to evaluate numerically phase coordinates mean values, variances and covariances for nonlinear systems with Gaussian initial conditions and random excitations (parameters and/or processes) in the assumption that the joint probability density of the phase and fluctuations vectors is nearly Gaussian.

Let the Eqs. of an object motion are

$$\dot{\vec{x}} = \vec{f}(\vec{x}, \vec{v}, t), \qquad \vec{v} \in R^m,$$

where $\vec{v}$ is the excitations vector. A time–line segment $[0, T]$ is broken into equal parts by points t_k

$$0 = t_0 < t_1 < ... < t_N = T.$$

Let us make the following designations:

$\vec{x}_k = \vec{x}(t_k), \qquad \vec{v}_k = \vec{v}(t_k), \qquad r = n + m,$
$\vec{z}_k = col\{\vec{x}_k, \vec{v}_k\}, \quad \vec{m}_k = M[\vec{z}_k],$
$D_k = M[(\vec{z}_k - \vec{m}_k)(\vec{z}_k - \vec{m}_k)^T] = \Big(D_{kij}\Big).$

For representation of $\vec{z}_{k+1}$ at the moment t_{k+1} through $\vec{z}_k$ the formulae of a one–step–by–step integrator intended for numerical solving of ODEqs. set are applied:

$$\vec{z}_{k+1} = \vec{G}(\vec{z}_k), \qquad k = \overline{0, N-1}.$$

To calculate the required statistical characteristics the formulae for the integrator $\vec{G}$ are expanded in the point (appropriate to $\vec{m}_k = M[\vec{z}_k]$) into a Taylor series up to the second order

$$z_{k+1,s} = G_s(\vec{m}_k) + \sum_{i=1}^{r} \tilde{A}_{ksi}(z_{ki} - m_{ki}) +$$

$$+ \sum_{i,j=1}^{r} \tilde{B}_{ksij}(z_{ki} - m_{ki})(z_{kj} - m_{kj}) \tag{3.14}$$

where

$$\tilde{A}_{ksi} = \frac{\partial G_s(\vec{z}_k)}{\partial z_{ki}}\bigg|_{\vec{z}_k = \vec{m}_k}, \qquad \tilde{B}_{ksij} = \frac{1}{2}\,\frac{\partial^2 G_s \vec{z}_k)}{\partial z_{ki}\partial z_{kj}}\bigg|_{\vec{z}_k = \vec{m}_k},$$

$$s = 1, 2, ..., r.$$

Expectating both parts of the equality (3.14) one can receive the formulae connecting the mean values of the vector $\vec{z}$ at $t = t_{k+1}$ with the vector $\vec{m}_k$ and the matrix D_k of variances and covariances at time t_k

$$m_{k+1,s} = G_s(\vec{m}_k) + \sum_{i,j=1}^{r} \tilde{B}_{ksij} D_{kij}.$$

Further let's multiply the relation (3.14) by the similar (at $s = p$) and also expectate the both parts of the equality received. With regard to relations for moments functions of the third and fourth orders for the Gaussian distribution we shall obtain the following formulae to find the matrix D_{k+1} elements

$$D_{k+1,sp} = \sum_{i,j=1}^{r} \tilde{A}_{ksi}\tilde{A}_{kpj} D_{kij} +$$

$$+ \sum_{i,j,\alpha,\beta=1}^{r} \tilde{B}_{ksij}\tilde{B}_{kp\alpha\beta}(D_{ki\alpha}D_{kj\beta} + D_{ki\beta}D_{kj\alpha}).$$

The above method was implemented due to the following scheme:

– symbolical building of the design-basis formulae with the CAS Reduce or Mathematica using;

– a numerical calculation of mean values and variances with Fortran programs or the CAS Mathematica software itself.

The similar schemes have been applied at projects such as a combination of the small parameter and power series methods for a calculation of the probability density logarithm for displacement in the Eq. (3.9); an expansion of the various systems phase vectors probability density into a series in terms of the Hermite functions etc.

For this purpose the CAS Formac, Reduce, Mathematica and programming languages Fortran and TurboPascal have been used. Except ODEqs. r.h.s. deriving for expansion coefficients subroutines in appropriate numerical programming languages were automatically generated with the CAS help .

Computer algebra using gives great advantages in the control theory. Let a controlled system (with a full feedback) has the form

$$d\vec{x}(t) = \vec{f}(\vec{x}, \vec{u}, t)dt + G(\vec{x}, t)d\vec{w}(t), \qquad \vec{x}(t_0) = \vec{x}_0, \tag{3.15}$$

$$t_0 \le t \le T < \infty.$$

Here $\vec{u} \in \mathbf{U} \subset \mathbf{R}^s$ is a control vector. One of tasks is to find such a control vector $\vec{u}_*(t)$ that allows to reach a minimum of the functional

$$\mathcal{I}[\vec{u}]= \mathbf{M}\left[\varphi(\vec{x}(T)) + \int\limits_{t_0}^{T} \mathcal{F}_0(\vec{x},\vec{u},t)dt\right]$$

Let $\mathcal{V}(\vec{x},t)$ be a loss function. Then it will satisfy to the Bellman Eq.

$$\frac{\partial \mathcal{V}}{\partial t} + \min\limits_{\vec{u}\in\mathbf{U}}\left[\mathcal{L}^*_{x,t}\mathcal{V}(\vec{x},t) + \mathcal{F}_0(\vec{x},\vec{u},t)\right]= 0, \qquad \mathcal{V}(\vec{x},T) = \varphi(\vec{x}), \tag{3.16}$$

$$\mathcal{L}^*_{x,t}[\cdot]= \sum_{i=1}^{n} a_i \frac{\partial}{\partial x_i}[\cdot]+\frac{1}{2}\sum_{i,j=1}^{n} b_{ij}\frac{\partial^2}{\partial x_i \partial x_j}[\cdot].$$

Here the vector $\vec{u}_*(t)$ is a minimizing vector-function of the appropriate expression in Eq.(3.16).

Let us consider the more concrete copy of system (3.15):

$$\dot{x}_i = h_i(\vec{x},t) + \sum_{j=1}^{s} h_{ij}(\vec{x},t)\,u_j + \sum_{j=1}^{m} g_{ij}(\vec{x},t)\xi_j.$$

In this case let us assume that the functional has the form

$$\mathcal{I}_0[\vec{u}] = \mathbf{M}\left[\varphi(\vec{x}(T)) + \int\limits_{t_0}^{T} \mathcal{F}(\vec{x},t)dt + \frac{1}{2}\int\limits_{t_0}^{T}\sum_{j=1}^{s}\left(\frac{u_j}{k_j}\right)^2 dt\right].$$

Then the Bellman Eq. is the following

$$\frac{\partial \mathcal{V}}{\partial t} + \sum_{i=1}^{n} h_i^* \frac{\partial \mathcal{V}}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^{n} b_{ij}\frac{\partial^2 \mathcal{V}}{\partial x_i \partial x_j} - \frac{1}{2}\sum_{j=1}^{s} k_j^2\left(\sum_{i=1}^{n} h_{ij}\frac{\partial \mathcal{V}}{\partial x_i}\right)^2 = -\mathcal{F}$$

and the control vector is

$$u_{j*} = -k_j^2 \sum_{i=1}^{n} h_{ij}\frac{\partial \mathcal{V}}{\partial x_i}.$$

To find the control vector it is needed to calculate the function $\mathcal{V}$. Our simply realized (with a help of Mathematica) iterative algorithm developing the Krasovsky scheme [15] and consisting of the following steps

$$\tilde{\mathcal{V}}(\vec{x},\tau) = \sum_{k=0}^{\infty}\tilde{\mathcal{V}}_k(\vec{x})\frac{\tau^k}{k!}, \qquad \tau = T - t,$$

$$\tilde{\mathcal{V}}_0 = \varphi,$$

$$\tilde{\mathcal{V}}_{k+1} = \sum_{i=1}^{n}\sum_{l=0}^{k}\mathbf{C}_k^l \tilde{h}^*_{i,k-l}\frac{\partial \tilde{\mathcal{V}}_l}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^{n}\sum_{l=0}^{k}\mathbf{C}_k^l \tilde{b}_{ij,k-l}\frac{\partial^2 \tilde{\mathcal{V}}_l}{\partial x_i \partial x_j} + \tilde{\mathcal{F}}_k -$$

$$-\frac{1}{2}\sum_{j=1}^{s} k_j^2 \sum_{r=0}^{k}\mathbf{C}_k^r\left(\sum_{i=1}^{n}\sum_{l=0}^{r}\mathbf{C}_r^l \tilde{h}_{ij,r-l}\frac{\partial \tilde{\mathcal{V}}_l}{\partial x_i}\right)\left(\sum_{i=1}^{n}\sum_{l=0}^{k-r}\mathbf{C}_{k-r}^l \tilde{h}_{ij,k-r-l}\frac{\partial \tilde{\mathcal{V}}_r}{\partial x_i}\right),$$

$$k \geq 0,$$

allows to solve the problem both in the deterministic and in the stochastic cases.

4 Using of the CAS Mathematica in the primary education system "Statistical Dynamics"

Nowadays all most perspective technologies in the university–level education require to use modern computers.

The development of the training system "Statistical Dynamics" is devoted to progress in this direction [28]. The tasks of this system are:

– to give to students and beginning researchers an accessible tool for solving simple problems of statistical dynamics;

– to acquaint these persons with a reasonably wide spectrum of random analysis methods.

When realizing this project we wanted first of all to satisfy educational needs. Therefore the main system properties are following:

– dynamic systems are described by SDEqs. sets in the Stratonovich sense;

– nonlinearities in Eqs. r.h.s. are polynomial (except the statistical linearization method);

– a dimension of SDEqs. sets is equal to 2 (excepting few algorithms);

– a number of described or mentioned methods and techniques is more than 70;

– methods of research are analytical, numerical and numerically analytical.

A CAS Mathematica software package is a nucleus of our system. Separate auxiliary programs using the TurboPascal software are realizing algorithms requiring long–time accounts with a floating point (integration of ODEqs. with the help of the Runge–Kutta method; a calculation of statistical characteristics of the phase coordinates vector within the power series method with the Gauss type formulae etc.).

The theoretical material was prepared with the LaTeX system enabling to include mathematical formulae and can be looked through with the Viewer program.

The research method for a concrete system analysis is chosen by means of a menu. In our new release of package the menus subsystem and the data preparing stage were realized with the help of Microsoft Visual FoxPro 5.0 environment subpackage. Also a technique of a LaTeX visualizing have been worked off for statistical dynamics foundations and methods showing.

Main system menu consists of such pads as "Methods and algorithms", "Models", "Data for calculations", "Format of output", "Service operations" and others. After any pad selection an appropriate popup submenu opens. This submenu allows to choose and to start a concrete action such as input. A number of screen forms for such input, storage and updating of data was built.

In the present version 11 methods have been realized. There are the Monte Carlo, iteration, iteration operator, moments, statistical linearization, power series, direct linearization, integrator methods, Gaussian approximation, detailed balance principle and potential algorithms.

The main principles of some methods realization within the system "Statistical Dynamics" where CAS Mathematica properties were essentially used are considered below.

4.1^0 *The iteration method* (see Section 3). It is supposed that an initial probability density is the product of the Gaussian one and a polynomial. Taking this into account successive operator $\mathcal{L}$ iterations are found by use of recursive identities. The procedure results in presenting probability density approximations as well as plots of the first moments calculated due to these approximations.

4.2^0 *The iteration operator method* (see Section 3). In addition to the assumption specified for the previous method it is required that FPK Eq. coefficients are exactly integrable w.r.t. t. The principles of realization are the same as in 4.1^0.

4.3^0 To apply *the moments method* [2] it is needed to build of an infinite ODEqs. set for phase vector moments functions using FPK Eq. and than to solve this set.

Due to a huge number of Eqs. for practice it is required to limit the above set. In our project the Mathematica code program is symbolically building Eqs. r.h.s. expressions up to a user's chosen order. A closure of the infinite Eqs. set is executed owing to the quasi-Gaussian hypothesis.

A Mathematica code program transforms closure relations and ODEqs. set for moments r.h.s. to a Pascal procedure which are compiled and linked together with the main program Runge and the procedure RungeKut. These processes are controlled by the system Mathematica environment. Input

for calculations is formed in an external file before running of the created EXE-module. After the termination of ODEqs. integration the results are read out by the same procedure and displayed on a screen in the form of plots.

4.4^0 *The modified power series method* [25] is intended for an approximate calculation of the phase vector probability density logarithm. According to the method this logarithm is expanded into an infinite series in terms of phase coordinates powers. To calculate coefficients of this series an infinite ODEqs. set is recorded. When using the method in practice this set is neglected and the procedure realizing this method generates only its finite part. Further the algorithm steps are the same as in the previous item. The results are presented in the same way as in item 4.1^0.

4.5^0 *The direct linearization method* is intended for the finding of the first moments of the phase vector if coefficients of white noises are independent of phase coordinates.

On the basis of SDEqs. terms symbolical representations by means of the CAS Mathematica an appropriate made linear in mathematical expectations SDEqs. set is derived and then a joint ODEqs. set satisfied by mean values, variances and covariances is under construction. Suitable r.h.s. are transformed to Pascal procedure statements. Further the algorithm steps are the same as in item 4.3^0.

4.6^0 *The integrator method* (see in Section 3). Using symbolical representations of SDEqs. and the CAS Mathematica properties relations connecting values of united vector $\vec{z}$ in consecutive moments of time are formed (the classical Runge-Kutta scheme is used). These relations are differentiated in a point appropriating to a mean values vector and the received coefficients of phase coordinates and parameters powers up to the second order are used in formulae for the calculation of the first moments. At the end of procedure running the received values are displayed on a screen.

4.7^0 *The Gaussian approximation method* [7] is intended to calculate approximately the first two moments of the phase vector. Our realization consists of automatic deriving of moments ODEqs. set and of further numerical accountings of needed characteristics.

4.8^0 *The detailed balance principle* allows to find the exact steady-state density (in some cases). An algorithm of this method was described in Section 3.

4.9^0 Main features and a realization technique of *the potential method* [36] are analogous the previous.

4.10^0 *Auxiliary procedures.* Main procedures for the above-described methods use some auxiliary ones intended for:

– the building of the drift and diffusion coefficients due to symbolic representations of SDEqs. r.h.s.;

– the generation of Pascal procedures for calculating ODEqs. set r.h.s. If their volume is large then r.h.s. are transformed to a programs complex the first program of which is main and other ones are UNIT modules etc.

As a package development it is supposed to form a new function (on the base of Visual FoxPro) such as a management for knowledge data bases about a simplest stochastic systems behavior.

5 Conclusions

Some applications of the computer algebra systems Formac, Reduce, Mathematica in different problems of an object mathematical modelling and random effects analysis have been described.

The above-considered examples specify a utility and an efficiency of such applications especially at a reasonable connection with numerical accounts on the base of complex packages combining such different software as computer algebra systems, systems for data bases development, graphic editors and ordinary programming languages for numerical purposes.

References

[1] Bogoliubov, N. and Mitropolski, U. (1961) *Asymptotic Methods in the Non–Linear Oscillations.* Gordon & Breach, New York.

[2] Bolotin, V.V. (1979) *Random Vibrations of Elastic Systems.* Nauka, Moscow (in Russian).

[3] Dashevsky, M.L. (1976) Technical realization of the moments–semi-invariants method of random processes analysis. *Automatica and Telemechanica.* (10), 23–26 (in Russian).

[4] Dashevsky, M.L. (1978) Algorithmization of the semi–invariants method of nonlinear systems studying. *Problems of Control and Information Theory.* **7** (5), 305–316 (1978) (in Russian).

[5] Dimentberg, M.F. (1980) *Nonlinear Stochastic Problems of Mechanical Vibrations.* Nauka, Moscow (in Russian).

[6] Drozdov, M.U. and Malanin, V.V. (1985) On building of Fortran programs by the CAS Reduce means. *Proc. of 3d Intern. Conf. On Computer Algebra In Physical Research* (Dubna, USSR, 1985), 114–119. JUNR, Dubna (in Russian).

[7] Evlanov, L.G. and Konstantinov, V.M. (1976) *Systems with Random Parameters.* Nauka, Moscow (in Russian).

[8] Gardiner, C.W. (1985) *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences.* 2nd edition. Springer–Verlag, Berlin.

[9] Grosheva, M.V., Efimov, G.B. et al. (1983) Computer Algebra Systems: Analytical Program Packages. *Informator*, (1). IPM, Moscow (in Russian).

[10] Hearn, A.C. (1983) *REDUCE3 User's Manual.* RAND publication CP78 (4/83).

[11] Heck, A. (1993) *Introduction to Maple.* Springer–Verlag, Berlin.

[12] Kashkarova, A.G. and Shin, V.I. (1986) Modified semi–invariants methods of stochastic systems analysis. *Automatica and Telemechanica.* (2), 69–79 (in Russian).

[13] Kendall, W.S. (1991) *Symbolic Itô calculus: an overview.* Research Report 223. University of Warwick. Department of Statistics.

[14] Kendall, W.S. (1992) *Itovsn3: doing stochastic calculus with Mathematica.* Research Report 238. University of Warwick. Department of Statistics.

[15] Krasovsky, A.A. (1973) *Systems of flight automatic control and their analytical desinging.* Nauka, Moscow (in Russian).

[16] Krasovsky, A.A. (1974) *Phase Space and Statistical Theory of Dynamical Systems.* Nauka, Moscow (in Russian).

[17] Looney, C.G. (1985) Numerical solution of systems of random differential equations with Gaussian statistics. *Journal of Math. Anal. and Appl.* **105**, 222–230.

[18] Lumpov, V.I. and Malanin, V.V. (1982) The iteration method realization for the Fokker–Planck–Kolmogorov equation solved in computer algebra system REDUCE2. *Dynamics of Controlled Mechanical Systems*,. 158–163. IPI, Irkutsk (in Russian).

[19] Lurje, A.I. (1961) *Analytical Mechanics.* Fizmatgiz, Moscow (in Russian).

[20] Malanin, V.V. and Zhdanov, G.A. (1982) On the iteration operator method of dynamical systems with random fluctuations analysis. *Problems of Controlled Motion Mechanics. Nonlinear Dynamical Systems*, 103–113. Perm (in Russian).

[21] Malanin, V.V. and Poloskov, I.E. (1991) On CA application in solving some statistical dynamical problems. *Proc. of IV Intern. Conf. on Computer Algebra in Physical Research* (Dubna, USSR, 1990), 335–339. World Scientific, Singapore.

[22] Mitropolsky, U.A. and Kolomiets, V.G. (1971) Averaging in stochastic systems. *Ukrainski Math. Journal.* **23** (3), 318–345 (in Russian).

[23] Paraev, U.I. (1976) *Introduction in Statistical Dynamics of Control and Filtration Processes.* Sovetskoe radio, Moscow (in Russian).

[24] Pogorelov, D.U. (1993) About coding of symbolical expressions at synthesis of rigid bodies systems motion equations. *Izvestija RAN. Tekhnicheskaja Kibernetika.* (6), 209–213 (in Russian).

[25] Poloskov, I.E. (1985) Application of FORMAC system for probability analysis of nonlinear mechanical systems. *Proc. of 3d Intern. Conf. On Computer Algebra In Physical Research* (Dubna, USSR, 1985), 289–294. JUNR, Dubna (in Russian).

[26] Poloskov, I.E. (1992) On application of the infinite linear sets method for systems with random parameters analysis. *Problems of Controlled Motion Mechanics. Optimization of Control Processes*, 105–109. Perm (in Russian).

[27] Poloskov, I.E. (1993) On integrator method application for nonlinear stochastic dynamics problems. *Problems of Controlled Motion Mechanics. Nonlinear Dynamical Systems*, 54–61. Perm (in Russian).

[28] Poloskov I.E. (1998) Compound program packages and a nonlinear random fluctuations analysis *Proc. of the 1998 International Symposium on Symbolic and Algebraic Computation (ISSAC-98), August 13-15, 1998, University of Rostock, Germany*, 70–75. ACM Press.

[29] Pugachev, V.S. and Sinitsyn, I.N. (1985) *Stochastic Differential Systems*. Nauka, Moscow (in Russian).

[30] Pugachev, V.S., Sinitsyn, I.N. and Shin, V.I. (1987) Program realization of the normal approximation method in problems of nonlinear stochastic systems analysis. *Automatica and Telemechanica*. (2), 62–68 (in Russian).

[31] Pugachev, V.S., Sinitsyn, I.N., Cherednichenko, A.A. and Shin, V.I. (1991) Software for multidimensional nonlinear stochastic systems analysis. *Automatica and Telemechanica*. (1), 87–97 (in Russian).

[32] Rehak, M.L., Dimaggio, F.L., Benaroya, H. and Elishakov, I. (1987) Random vibrations with MACSYMA. *Computer Methods in Applied Mechanics and Engineering*. **61**, 61-70.

[33] Risken, H. (1984) *The Fokker-Planck Equation. Methods of Solution and Applications*. Springer–Verlag, Berlin.

[34] Roberts, J.B. and Spanos, P.D. (1986) Stochastic averaging: an approximation method of solving random vibration problems. *Int. Journal Non–Linear Mech.* **21**(2), 111–134.

[35] Song, T.T. (1973) *Random Differential Equations in Science and Engineering*. Academic Press, New York, London.

[36] Stratonovitch, R.L. (1963) *Topics in the Theory of Random Noise*, Vols 1 and 2. Gordon & Breach, New York.

[37] Tikhonov, V.I. and Mironov, M.A. (1977) *Markov Processes*. Sovetskoe radio, Moscow (in Russian).

[38] Tomovic, R. (1963) *Sensitivity Analysis of Dynamic Systems*. Belgrade.

[39] Valkeila, E. (1991) Computer algebra and stochastic calculus – some probabilities. *CWI Newsletter*.

[40] Wolfram, S. (1996) *The Mathematica book*. 3d ed. Cambridge University Press.

6 Appendix

6.1 Modelling of relative movements of an absolutely rigid bodies chain

Let us consider the problem of deriving vector–matrix Eqs. of relative movements of an absolutely rigid bodies tree (or chain) [6, 19], taking first partial derivatives of DEqs. r.h.s. and building a part of a Pascal program.

It is supposed that any mechanical system analyzed may be described as a tree consisting of rigid bodies. Their own coordinates system (c.s.) is connected with each body of this system. The relative motion of body number i of the tree is understood as the motion of its connected c.s. in axes of the c.s. of previous body number $i-1$ from that branch. The c.s. of body number 0 is accepted for the inertial c.s.

The vector of absolute angular velocity of the body i c.s. in axes of the connected c.s. of this body can be calculated from the formula:

$$\vec{\omega}_i^A = A_{i,i-1}\vec{\omega}_{i-1}^A + \vec{\omega}_i^e,$$

where i is a body index (number); $A_{i,i-1}$ is the matrix of coordinates transformation from the body $i-1$ connected c.s. to body i c.s. at the expense of the rotation of the latter w.r.t. the former; $\vec{\omega}_i^e$ is the vector of relative angular velocity of the body i c.s. in its connected c.s.

The vector of absolute angular acceleration of the body i c.s. in axes of its connected c.s. ($\dot{\vec{\omega}}_i^A$) is calculated from the formula:

$$\dot{\vec{\omega}}_i^A = T_{ci}^{-1}(mom_{ci}\vec{F}_i - \vec{\omega}_i^A \times T_{ci}\vec{\omega}_i^A)$$

where T_{ci} is a body i inertia tensor calculated w.r.t. its mass center in axes parallel to axes of the connected c.s. of this body; $mom_{ci}\vec{F}_i$ is the main moment of forces acting on body i and calculated relatively its mass center; $\vec{F}_i$ is the main vector of forces acting on body i in axes of its connected c.s.

The vector of relative angular acceleration of the body i c.s. concerning the body $i-1$ connected c.s. in axes of the body i c.s. is calculated from the formula:

$$\dot{\vec{\omega}}_i^e = \dot{\vec{\omega}}_i^A - (A_{i,i-1}\dot{\vec{\omega}}_{i-1}^A + \vec{\omega}_i^A \times \vec{\omega}_i^e).$$

The vector of the first derivatives of the orientation angles for the body i c.s. relative to the body $i-1$ c.s. is calculated from the formula:

$$\dot{\vec{\alpha}} = A_i^E \vec{\omega}_i^e$$

where A_i^E is the matrix inverse to the Euler matrix and used at the kinematic Euler relations.

The vector of absolute linear acceleration of origin of the body i connected c.s. in axes of the latter is calculated from the formula

$$\vec{w}_{0i}^A = \frac{\vec{F}_i}{M_i} - [\dot{\vec{\omega}}_i^A \times \vec{r}_{ci} + \vec{\omega}_i^A \times (\vec{\omega}_i^A \times \vec{r}_{ci})]$$

where M_i is the body i mass, $\vec{r}_{ci}$ is the radius-vector of the body i mass center in the axes of its connected c.s.

The vector of relative linear acceleration of origin of the body i connected c.s. relative to body $i-1$ in the axes of the latter is calculated from the formula:

$$\vec{w}_{0i}^r = A_{i-1,i}\vec{w}_{0i}^A -$$

$$-[\vec{w}_{0,i-1}^A + \dot{\vec{\omega}}_{i-1}^A \times \vec{r}_{ci} + \vec{\omega}_{i-1}^A \times (\vec{\omega}_{i-1}^A \times \vec{r}_{0i}) + 2\vec{\omega}_{i-1}^A \times \tilde{\vec{r}}_{0i}]$$

where $\vec{r}_{0i}$ is the radius vector of the body i c.s. origin in axes of the body $i-1$ connected c.s.; $\tilde{\vec{r}}_{0i}$ is the vector of relative linear velocities of origin of the body i c.s. relative to the body $i-1$ connected c.s. in axes of the latter.

Here the input is as follows

```
vector wai1,wei,wai,wa1i,fi,rfci,we1i;
matrix aii1,  tci , aei;
vector wa1 i1, alf1i, w0ai, rci, w0ri, w0ai1;
scalar two, m 1i ,k0,ki;
vector wa1i1,r0i,v0ri;
constant two,k0,ki;
derivatives order is 1;
formulae;
wai=aii1*wai1+wei;
wa1i=tci^*($([rfci,fi],k0,ki)$-[wai,tci*wai]);
we1i=wa1i-(aii1*wa1i1+[wai,wei]);
alf1i=aei*wei;
w0ai=m1i*$(fi,k0,ki)$-([wa1i,rci]+[wai,[wai,rci]]);
w0ri=aii1*w0ai-(w0ai1+[wa1i1,r0i]+[wai1,[wai1,r0i]]+two*[wai1,v0ri]);
end of formulae;
```

where the following designations are introduced:

$$
\begin{array}{lll}
wai1-\vec{w}_i^A, & wei-\vec{w}_i^e, & fi-\vec{F}_i, \\
wei1-\dot{\vec{\omega}}_i^e, & wa1i1-\dot{\vec{\omega}}_{i-1}^A, & aii1-A_{i,i-1}, \\
tci-T_{ci}, & aei-A_i^E, & wa1i-\vec{w}_i^A, \\
alf1i-\dot{\vec{\alpha}}, & w0ai-\vec{w}_{0i}^A, & rci-\vec{r}_{ci}, \\
r0i-\vec{r}_{0i}, & w0ri-\vec{w}_{0i}^r, & w0ai1-\vec{w}_{0,i-1}^A, \\
two-2, & m1i-1/M_i, &
\end{array}
$$

$rfci$ are the radii–vectors of the forces attachment points on body i, ki is the number of forces acting on body i.

The first part of the results in the vector–matrix form is submitted below

```
wai:=aii1 wai1 + wei;
wx0xw1:=vect[rfci, fi];
wx0xw2:=sumi[wx0xw1, k0, ki];
wx0xw4:=tci wai;
wx0xw3:=vect[wai, wx0xw4];
wx0xw5:=wx0xw2 - wx0xw3;
wx0xw6:=inv[tci];
wa1i:=wx0xw6 wx0xw5;
wx0xw7:=vect[wai, wei];
wx0xw8:=aii1 wa1i1 + wx0xw7;
we1i:=wa1i - wx0xw8;
alf1i:=aei wei;
wx0xw9:=sumi[fi, k0, ki];
wx0xw10:=vect[wa1i, rci];
wx0xw11:=vect[wai, rci];
wx0xw12:=vect[wai, wx0xw11];
.............................
wx0xw36:=vect[wai1, wx0xw15$100];
wx0xw16$100:=wx0xw35 + wx0xw36;
wx0xw37:=vect[wai1$100, v0ri];
wx0xw38:=vect[wai1, v0ri$100];
wx0xw17$100:=wx0xw37 + wx0xw38;
wx0xw18$100:=w0ai1$100 + wx0xw14$100 + wx0xw16$100 + two wx0xw17$100;
w0ri$100:=aii1$100 w0ai + aii1 w0ai$100 - wx0xw18$100;
```

and a peace of the second part is here

```
m$one:=-1.0;  {----Working var----}
{=== 1 ===}
P_MultMV(aii1,wai1,wx0xw39);
{----------Sum[{wx0xw39, wei}]----------}
P_VecZ(wai);
P_SumV(wai,wx0xw39);
P_SumV(wai,wei);
{=== 2 ===}
P_Vect(rfci,fi,wx0xw1);
{=== 3 ===}
{-----sumi[wx0xw1,k0,ki]----------}
P_VecZ(wx0xw2);
For i$wv:=k0 To ki Do
   P_SumV(wx0xw2,wx0xw1[i$wv]);
.........................
{=== 68 ===}
P_MultMV(aii1$100,w0ai,wx0xw66);
P_MultMV(aii1,w0ai$100,wx0xw67);
P_MultSV(m$one,wx0xw18$100,wx0xw68);
{----------Sum[{wx0xw66, wx0xw67, wx0xw68}]----------}
P_VecZ(w0ri$100);
P_SumV(w0ri$100,wx0xw66);
P_SumV(w0ri$100,wx0xw67);
P_SumV(w0ri$100,wx0xw68);
```

6.2 Surfaces of steady-state densities

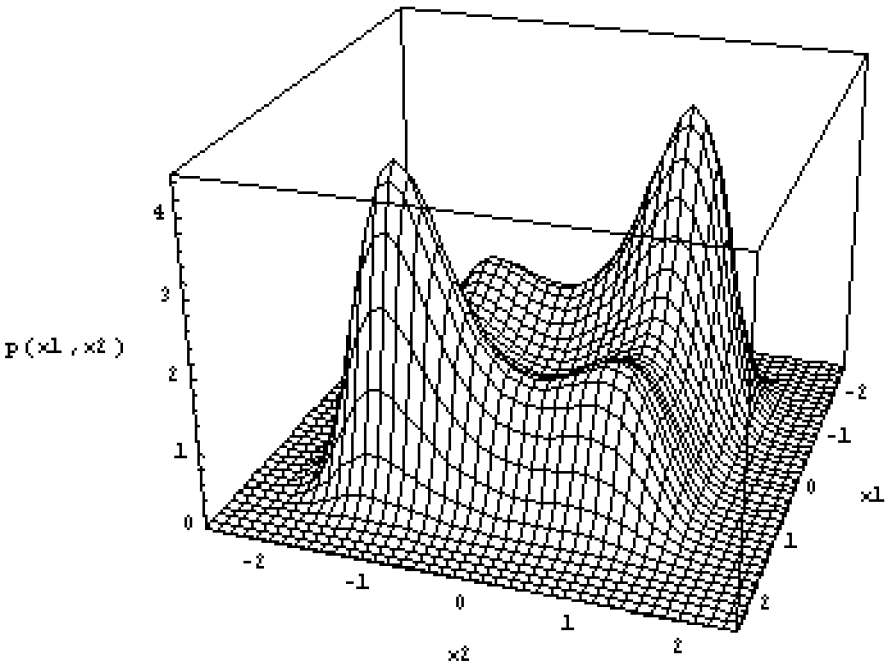

Fig. : 6.2.1

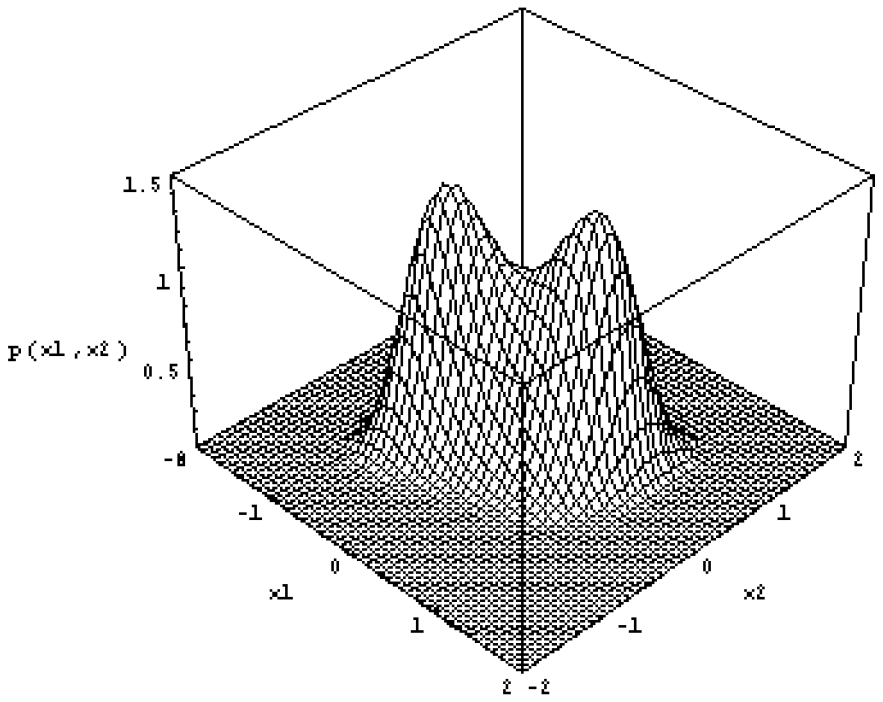

Fig. : 6.2.2

6.3 Solution of Kolmogorov-Feller Eq.

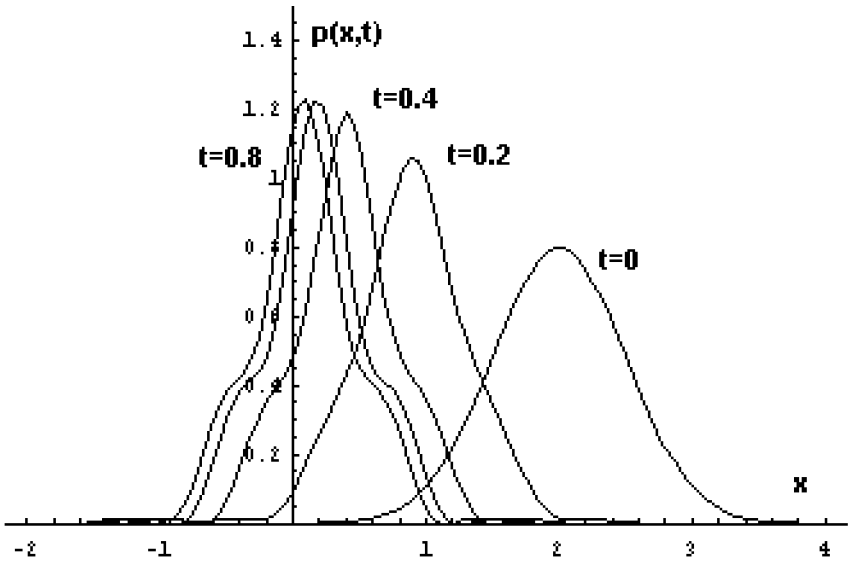

Fig.6.3