

POSSIBILITIES OF DYNAMIC SYSTEMS SIMULATION

Cristina Coculescu¹

ABSTRACT

Modeling dynamic systems can be made using several instruments and techniques, simulation being among these. Simulation of a system operating, allows evaluation of the kind how it will develop in certain conditions or because its management using a specific set of rules. In many cases, simulation is the only possible solution for making such evaluations. In this work we'll show general considerations about possibilities of simulation dynamic systems using dedicated programs for simulating systems.

Keywords: dynamic system, numerical integration, simulation

JEL code: C6, C88

1. BASIC ASPECTS ABOUT DYNAMIC SYSTEMS

An important step for scientific knowledge is passing from a system or real process to corresponding mathematical model and hence, to a physical one. This thing is possible also because in nature there are several real physical phenomena which are described by the same kinds of equations but the functions and variables within that mathematical structure have different meanings. With other words, nature unity appears in an amazing similarity of differential equations from different kinds of phenomena.

A dynamic system is one which develops in time. If the set of times when the system develops is a subset of:

- integer numbers set $\mathbb{Z}$, then the system is called discrete time system (*discrete system*);
- real number set $\mathbb{R}$, then the system has continuous time (*continuous system*).

In dynamic systems modeling, particular rules of mass and energy conservation are used, mathematically expressed as balance equations, that are simply differential equations, where derivation variable is time, t , or with partial derivatives – where there is at least one derivation variable besides time (a space coordinate as example).

Mathematic model of a dynamic system is the set of differential or integer-differential equations which show system behavior under input values action.

Because differential equations which processes of real world are often non-linear and of superior order, finding an analytical solution (exact one) of Cauchy problem is difficult. The alternative is finding an approximate solution that is using of a numerical algorithm. Therefore, solution found using a numerical method, is a string of approximations of the values of exact solutions, computed in step times, usually of same interval.

¹ Ph.D., Associate Professor, Romanian-American University, 1B Expozitiei Bd, Sector 1, Bucharest, E-mail: cristina_coculescu@yahoo.com

In complex systems study, when finding an analytical solution is impossible and pure practice on real system is, for a reason or another, not operational, simulation techniques are normally used.

Computer system simulation gives needed method from the study of their behavior. It is the opportunity of to correlate all factors and to clarify their kind of interference, being a method which gives the possibility of making forecasts about simulated system or process for its dynamic aspects.

In this context, the term of “*simulation*” is associated to time evolution of a dynamic system described using an adequate model, when a known input signal is applied to the system; with other words, there is considered finding system output problem, also named answer to a certain known input.

2. NUMERICAL INTEGRATION OF DIFFERENTIAL EQUATIONS.

Theoretical Basics

In this section, we'll show the main theoretical results which rules numerical methods of solving differential equations and differential equation systems.

Let be differential equation:

$$y' = f(x, y) \quad (1)$$

with initial condition:

$$y(x_0) = y_0 \quad (2)$$

The problem of finding the function $y = y(x)$ that identically proves equation (1) and condition (2) is named *Cauchy problem* or the problem with starting conditions.

Differential equations are:

- *ordinal* (ODE) – where derivation variable is the time, t ;
- *with partial derivatives* (PDE) – where there is at least one derivation variable besides time (a space coordinate as example).

For many ordinal differential equations, classical integration methods aren't applicable or these methods demands big computing. Therefore, approximate methods need, which are of two kinds: analytical, that give the approximate solution as analytical expressions and numerical, within the solution is given as a string of values, beginning from a starting value of the solution.

Through analytical methods, we mention step-by-step approximation method and the method of finding solution of a series of powers.

Numerical solving of a Cauchy problem consists of finding a set of points $y_1, y_2, \dots, y_n$ which approximate exact values $y(x_1), y(x_2), \dots, y(x_n)$ of integral curve that pass through initial point (x_0, y_0) .

Let's consider integration step:

$$h = x_{i+1} - x_i, \quad i = 0, 1, 2, \dots, n - 1 \quad (3)$$

Numerical methods for integration of ordinal differential equations are divided into two classes:

- *methods with separate steps for computing the ordinate y_{i+1}* , need knowledge of the coordinates of prior point (x_i, y_i) and the size of integration step. Many of them use Taylor series development for the function $y = y(x)$ around a point.
- *linked step methods. for computing ordinate y_{i+1}* , demand knowledge of integration step h and of several anterior points $(x_i, y_i); (x_{i+1}, y_{i+1}); \dots$. The values $y_i, y_{i+1}, \dots$ are generally established using numerical methods with separate steps.

Both kinds of methods can use for computing approximate values $y_1, y_2, \dots, y_n$ explicit or implicit kind algorithms.

For a separate step method, an explicit algorithm is of kind:

$$y_{i+1} = y_i + h\varphi(x_i, y_i, h), \quad i = 0, 1, 2, \dots, n - 1 \quad (4)$$

and an implicit one:

$$y_{i+1}^c = y_i + h\varphi(x_i, y_i, x_{i+1}, y_{i+1}^P, h), \quad i = 0, 1, 2, \dots, n - 1 \quad (5)$$

The approximate y_{i+1}^P which appears in the right branch of equality is named prediction value and it is computed by an explicit algorithm and y_{i+1}^c is called correct value of the coordinate y_{i+1} . Similarly we can build such explicit and implicit algorithms for linked steps numerical methods.

Between separate steps numerical methods we remember: Taylor series development method, Euler method (polygonal lines method), Runge-Kutta method, improved Euler method and from those which linked steps: Adams-Moulton, Milne etc.

Numerical methods encountered within ordinal differential equations can be easily extended for ordinal differential equations systems.

Dynamic systems which have more than an independent variable (time, space coordinates) are characterized by differential equations with partial derivatives (PDDE). Simulation of this kind of systems, therefore consists of numerical integration of compound PDDE.

Numerical integration of PDDE is relied on derivatives approximation using finite differences that means fuzzification of the ranges of independent variables. Clearly, numerical integration will be as exact as integration steps are smaller.

3. METHODS OF DYNAMIC SYSTEM SIMULATION IN MATLAB

Matlab is a program set of high performance for numerical computing and graphic representations in science and engineering. Developed during several years, *Matlab* is now a standard in university media and also in research and practical solving of problems of experimental data processing, statistical control, signal processing, finding systems etc.

To simulate a dynamic system is the same thing to solve (integrate) the differential equation which describes it, in known initial conditions, which represents initial starting of the system.

Integrated environment Matlab gives a set of specialized function for numerical integration of ODE that implements methods, from the easiest direct methods to multistep ones, improved by complicated algorithms for variation of integration step for approximation error reducing. The name of these functions begins with the character string `ode` (ordinary differential equations).

In what concerns implementation of numerical integration methods of PDDE, in Matlab there are no dedicated functions, because of the richness of methods kind and variety.

Mainly, final objective of every numerical method is to establish as accurate as possible an approximate solution of the problem using minimum memory and amount of computing operations. The difference between exact and approximate value (computed using a numerical method) of an ordinate, is total error of computing step. Total error on step consists of approximation errors (because numerical method), of rounding (used because digit number limitation) and of propagation, which appears because the errors of prior steps.

Approximation method depends on used numerical method. Exactly, on integration step h . therefore, decreasing of approximation method is possible if the step h is correspondingly decreased.

Rounding error depends on computer facilities (how many important digits we can use). It can be reduced if double-precision mode is used. However, rounding error increases according to the amount of steps because error propagation from one step to another. Accordingly, if integration step is decreased for approximation error reducing, the number of steps will increase and in consequence, rounding error will be bigger. Therefore, there is an optimal value of integration step, h , wherefore total step error is minimal.

The kind how rounding errors modify computed values precision, is clarified in numerical stability analysis of integration methods. This consists of to study if small variations in initial conditions give small variations in found solution. If this happens, we can consider the method as being numerically stable.

In certain cases, an integration step enough for insure desired precision, can be much bigger than would be necessary for assure method stability. Such ODE's are named stiff, in what concerns numerical integration. In systemic context, stiffness appears at ODE's having model phenomena with several different time scales.

Implicit Euler method is extremely easy but demands much smaller integration steps then those used by other methods for reach the same precision. Hence, this is practically used for small complexity systems. However, this is worth to mention that its using with same step, properly chosen, can be a good solution for simulating systems having stiff behavior.

Runge-Kutta methods are right for large categories of dynamic systems characterized by a (linear or non-linear) balanced dynamic (without stiff behavior tendencies)

Adams predictor-corrector methods are recommended for simulating systems with an easy or moderate stiff behavior.

Burlish-Stoer algorithm can replace, sometimes with better results, Adams predictor-corrector algorithms.

Gear kind methods are the only which can give correct results in simulation systems having a strong stiff behavior. Their using is recommended only in such cases, for other kind of problems it being inefficient.

In consequence, the speed and precision wherewith Cauchy problem attached to a differential equations system is solved depends on used integration method, size of integration step and on maximum error accepted by user. Errors overview and the fitness of integration step will avoid numerical instability but integration step needed for keeping numerical stability is too small for computing a solution very easy to approximate. As a conclusion, integration step results from an accord between method rightness and precision.

In what concerns implementation, a stiff ODE can be solved faster using better a dedicate code than a standard one. This reason enabled the classification of ode functions of Matlab library in functions for stiff and non-stiff ODE.

As the most functions of Matlab library, ode functions can have a changeable number of input variables. Whatever the case, minimal (and most used) calling shape is:

$$[t,y]=\text{odexyz}('f',tspan,y0)$$

where odexyz is one of ode functions.

First input argument show what must be solved. The second shows integration step (simulation time step). The third input argument shows the conditions for solving given ODE: the vector y_0 , of dimension equal to given ODE degree, consists of starting conditions.

First output argument, t is the vector of values wherein step-by-step times has performed (in the case of system answer simulation).

Second output argument, y , is a matrix having so lines as elements of t , that is the number of integration steps and the number of column is equal to the degree of ODE to be solved.

There is the chance to modify implicit integration parameters using additional input arguments.

The most Matlab functions of integration ODE are:

- for standard equations (for non-stiff ODE's): ode23, ode45, ode113;
- for stiff ODE's: ode15s, ode23s, ode23t, ode23tb.

The functions ode23 and ode45 uses variation of integration step through the alternation of direct Runge-Kutta formulas (RK) of order 2, with RK of third degree, RK of order 4 with

RK of order 5, the precision of given solutions being enough for a large class of applications.

ODE functions are in the base for using Simulink (part of Matlab integrated environment, composed by functional blocs which implements the concept of model for a dynamic system). visual libraries in dynamic system simulation.

Simulink is a useful interactive environment for modeling, analysis and simulation of a big number of mathematic and physic systems. Of optional extension of Matlab program kit, Simulink gives a user graphic interface, for making dynamic systems models represented in bloc-scheme.

Dynamic simulation for studied system, is relied on the knowledge of differential equation system, on making the bloc scheme and of course, on using a numerical integration method. Simulation results can be displayed both graphically and as numerical tables.

Using S-Functions library blocs, user blocs can be made, which are integrated in existing schemes, realized by the means of blocs of standard Simulink library.

4. CONCLUSIONS

Simulation of a system operation, allows evaluation of the kind how it will develop in certain cases or because its management using a specific set of rules. In many cases, simulation is the only possible solution for making such evaluations.

For solving dynamic modes, simulation programs need to be fast, easy and developable. In most cases, these relies on the practice and developments of Fortran programming language which has sturdy numeric algorithms and well developed program library. However, last years, the facilities given by Windows, especially the graphics, are superior so that specific programs with better graphic facilities have developed.

Simulation of Simulink models, demands numerical integration of differential equations systems. Simulink gives a set of integration algorithms for simulation such equations. Unfortunately, because of variety of dynamic behavior of a system, none of these methods warrants efficient and exact simulation of any kind of model.

The results of simulations, in what concerns speed and precision, varies depending on model and initial conditions. Right choosing of integration method and carefully setting of simulation parameters, are necessary conditions for gaining exact results.

5. REFERENCES

- [1] Arrowsmith, D. K., Place, C. M., *An Introduction to Dynamical Systems*, Cambridge University Press, 1990;
- [2] Băzăvan, P., *Algoritmi Numerici în Studiul Sistemelor Dinamice*, Editura Sitech, Craiova, 2005;
- [3] Demidovitch, B, Maron, I., *Éléments de Calcul Numérique*, Mir, Moscow, 1973;
- [4] Ebăncă, D., *Metode de Calcul Numeric*, Editura Sitech, Craiova, 1994;

- [5] Stuart, A. M., Humphries, R., *Dynamical Systems and Numerical Analysis*, Cambridge University Press, 1996;
- [6] [Mat93] *** , *SIMULINK, Dynamic System Simulation Software*, User=s Guide, The MathWorks Inc.,