Brigham Young University
BYU ScholarsArchive

International Congress on Environmental Modelling and Software

1st International Congress on Environmental Modelling and Software - Lugano, Switzerland - June 2002

Jul 1st, 12:00 AM

Ecological Modelling Using the Spreadsheet Model TabSim

L. Matejicek

Follow this and additional works at: https://scholarsarchive.byu.edu/iemssconference

Matejicek, L., "Ecological Modelling Using the Spreadsheet Model TabSim" (2002). *International Congress on Environmental Modelling and Software*. 239.
https://scholarsarchive.byu.edu/iemssconference/2002/all/239

This Event is brought to you for free and open access by the Civil and Environmental Engineering at BYU ScholarsArchive. It has been accepted for inclusion in International Congress on Environmental Modelling and Software by an authorized administrator of BYU ScholarsArchive. For more information, please contact scholarsarchive@byu.edu, ellen_amatangelo@byu.edu.

Ecological Modelling Using the Spreadsheet Model TabSim

L. Matějíček
Institute for Environmental Studies, Charles University, Prague, 128 01, Czech Republic
(LMATEJIC@MBOX.CESNET.CZ)

Abstract: Simulation of ecological systems represents more sophisticated analysis, which extends possibilities of standard research methods. In contrast to majority of computer languages available for calculation of dynamic models, spreadsheet models (TabSim-**Tab**le **Sim**ulation) offer to spend more effort on investigating the features of a dynamic system, rather than writing a program code. The program structures are replaced by table objects, which represent basic dynamic relations (integrations and time delays) and static relations (described by built-in functions and user functions). In addition to calculation of dynamic and static relations, the wide range of built-in graphs and graphic objects can be used for presentation of simulation results. The TabSim has been developed expressly for modelling systems described by time dependent, non-linear differential equations and transfer functions. Typical applications include energy transformation, chemical process representation, ecological relations and biomedical systems. The numerical solution is carried out by a number of algorithms. The basic level represents Runge-Kutta and Adams-Moulton algorithms with a variable step and order. Each structure of a numerical model can be solved directly by calculation of static functions inside a spreadsheet or transferred to an internal programming code. Simulation of dynamic models, preferentially of ecological systems, is demonstrated on a few examples that illustrate the process of development and solution.

Keywords: dynamic modelling; simulation; spreadsheet

1. INTRODUCTION

A spate of definitions of simulation has been published since the late sixties. One of the most important factors, which raise the importance of simulation, is the progress in computer science. So definitions often include aspirations, which are limited by possibilities of computer equipment. Simulation is defined in the frame of modelling as a set of techniques that allow an examination of the dynamic behaviour of models. The models, mostly described by mathematical relations, represent abstract or hypothetical forms of real systems. The system by itself is considered to be a collection of components, which interact with each other, within defined boundaries, to produce a particular pattern of behaviour. The boundaries facilitate to separate the system from the rest of the universe, which contains all the components and interactions out of the defined resolution and focus. In the broadest range, models and their simulations take a number of forms. The abstract model based on mathematical relations can be divided into discrete or continuous forms, which are block or equation oriented, stochastic or deterministic, static or dynamic.

If the model behaviour changes continuously and the output characteristics should be accessible at every instant time, the continuous form is used. On the other side the discrete model is required. The block-oriented forms correspond with the components of analog computers, which were used in early simulation works. Later, the methods of the model simulation by functional blocks were adopted in the frame of the new digital simulation software. Most of the well-known simulation software packages still uses block-oriented descriptions of models (ACSL). The equation-oriented simulation is mostly developed on the basis of high-level computer programming languages. The model relations are included in lines of a programming code. An uncertainty of an observed behaviour of real systems is transferred into the stochastic or probabilistic modelling elements. If the uncertainties are of a little importance compared to the general behaviour of the model, the stochastic elements are removed. Then the models are treated as being deterministic. In general, simulation evaluates behaviour of models, which describe dynamic phenomena. The dynamic modelling can include static relationships, which

ignore the time dependence of dynamic phenomena. If all the dynamic relations are excluded, the model becomes static. This case covers the modelling of systems or processes that are in steady states. The above contrasting pairs of model characteristics specify various types of models and tools, which are necessary for their simulation [Bennett, 1995].

2. ECOLOGICAL MODELLING AND SIMULATION IN TABSIM

Ecological modelling deals mostly with continuous models described by differential equations, either ordinary or partial. The solution of models comes into deterministic and dynamic forms, which can be block or equation oriented. The models can be extended by discrete and stochastic parts interacting together with continuous parts in combined simulation. The earliest programming tools were represented by a high-level programming language such as FORTRAN. Due to the syntax restrictions and the need of programming experience, academic and industry centres became the development of simulation languages, which are more procedural and free from restrictions of any high-level programming language. Their basic structure contains numeric integration routines, which represent the engine of continuous simulation, model description, data input/output modules and graphic tools for display simulation results. The next generations of personal computers with new generations of processors, graphic tools and storage equipment promote new ways of simulation. A number of offered simulation programs documents this situation (ACSL-Graphic Modeller, MATLAB-Simulink, STELLA [Hannon et al., 1994]).

2.1 Structure of Dynamic Models in TabSim

The continuous dynamic models contain dynamic and static relations. The dynamic relations can be reduced into two basic phenomena, which are represented by integrations and time delays. The static relations contain all the mathematical expressions among model variables, which refer to the same actual time. The structure of the dynamic system illustrates figure 1. The general schema can be described by the sets of equations:

$$\begin{aligned} \frac{\mathbf{dx}(t)}{dt} &= \mathbf{f}[\mathbf{x}(t), \mathbf{r}(t), \mathbf{u}(t), t] \\ r_{ij}(t) &= x_j(t - t_{Dij}) \\ \mathbf{y}(t) &= \mathbf{g}[\mathbf{x}(t), \mathbf{r}(t), \mathbf{u}(t), t] \end{aligned} \quad (1)$$

where x and r represent state vector and delayed state vector, u and y are input/output vectors:

$$\begin{aligned} \mathbf{x}(t) &= [x_1(t), x_2(t), \ldots x_n(t)]^T \\ \mathbf{r}(t) &= [r_{11}(t), r_{1k}(t), \ldots, r_{n1}(t), \ldots, r_{nk}(t)]^T \\ \mathbf{u}(t) &= [u_1(t), u_2(t), \ldots u_l(t)]^T \\ \mathbf{y}(t) &= [y_1(t), y_2(t), \ldots y_m(t)]^T \end{aligned} \quad (2)$$

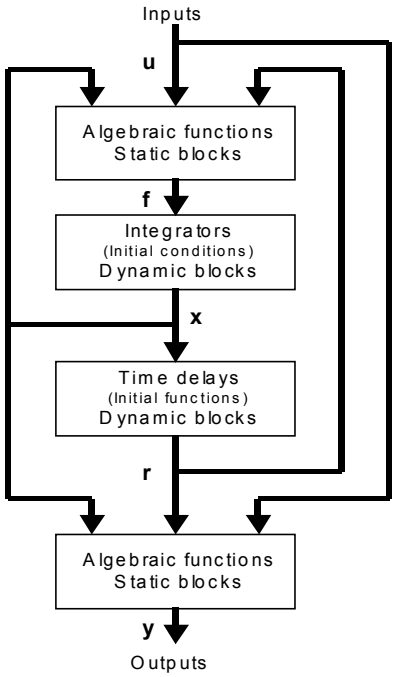

Figure 1. Structure of the dynamic model

The vector f forms static functions that represent internal inputs of integrations. The vector g includes static functions that are used for conversion of state variables, delayed state variables and inputs. The dynamic functions are fixed in the integrator block and the time delay block. In general, the simulation run includes three separate stages, which represent an initialisation, a dynamic stage and a termination. The initialisation covers settings of initial conditions and model parameters. The simulation stage represents the main part of simulation, which covers all the calculation during the simulation period. The storage of data and the display of model behaviour include the termination.

2.2 Structure of Spreadsheet Models in TabSim

Spreadsheet programs have been used for a few years as standard software tools for calculation and economic prediction. They are accessible on nearly each personal computer. The data and

calculation formulas are located in the cells, which are grouped into the tables. The TabSim has been developed to support this data structure and more over to enable full data transparency and to take in advantage a wide range of built-in functions and graphs.

The basic structure represents a few tables, which contain dynamic relations: integrations, time delays, inputs and outputs. The input and output blocks serve for simulation studies as the data source and the data target. All the dynamic blocks are illustrated in figure 2. The integrations can be organized into the list or the grid. The grids are used for simulation of spatial dynamic models. The compartments of a model are located in the regular grids, which form layers. State variables-x_i, functions-f_i on the right side of ordinary differential equations and initial conditions ix_i are complemented by errors err_i that are calculated as difference of the 1st order numerical method and the chosen numerical method. The time delays operate with data, which contain last values of model variables. In addition to the delayed state variables r_i, actual state variables x_i and time delays nDT_i., the $data_i$ represent names of the ranges that contain delayed values. k_i are the communication intervals for data writing and reading (expressed as the number of integration steps). The inputs and outputs of data are managed with two blocks. The $data_i$ represent names of the ranges that contain input or output data. k_i are the communication intervals for reading or writing of data. u_i contain the actual values of inputs. The values inside the communication intervals are estimated by the linear interpolation. y_i contain the actual values of outputs, which are captured and stored in the specified ranges. All the static relations are developed by built-in functions that are included in the spreadsheet. More over, the users can develop their own functions in the environment of a spreadsheet's macro language.

The simulation is managed by the table, which includes basic simulation parameters (time period, integration step, numerical method), figure 3. The table can contain one more columns, which represent the individual simulations. The time period n_i are expressed as numbers of integration steps dt_i. The TabSim includes a few of numerical algorithms nm_i (Euler, Runge-Kutta the 2nd and the 4th order, Adams Moulton predictor-corrector the 2nd- 4th order). Other parameters form the estimates of errors of the numerical algorithms err_i and numbers of the iterations $iter_i$. The differences between the 1st order and the chosen numerical method are displayed according to the setting of the deviation (deviation = true). The start of simulation can be initialised by initial conditions (initialisation = true) or by the actual values of the state variables (initialisation = false). The $TabSim20_i$, $integ_i$, $grid_i$, $delay_i$, $input_i$ and $output_i$ specify the locations of data in the environment of a spreadsheet. The TabSim can run one or more simulations at the same time. The time periods n_i, the time steps dt_i and the numerical methods nm_i can be changed during the simulation.

Integrations (list)

x	fx	ix	error
x_1	fx_1	ix_1	err_1
x_1	fx_2	ix_2	err_2
...	...	...	...
x_n	fx_n	ix_n	err_n

Integrations (grid)

states, functions, initial conditions				
x_{11}	...	x_{1j}	...	x_{1n}
state variables				
x_{i1}	...	x_{ij}	...	x_{in}
...	...	...	...	...
x_{m1}	...	x_{mj}	...	x_{mn}
fx_{11}	...	fx_{1j}	...	fx_{1n}
functions				
fx_{i1}	...	fx_{ij}	...	fx_{in}
...	...	...	...	...
fx_{m1}	...	fx_{mj}	...	fx_{mn}
ix_{11}	...	ix_{1j}	...	ix_{1n}
initial conditions				
ix_{i1}	...	ix_{ij}	...	ix_{in}
...	...	...	...	...
ix_{m1}	...	ix_{mj}	...	ix_{mn}

Time delays/weight-delays (list)

r	x	data	k	Delay
r_1	x_1	$Column_1$	k_1	$nDT_1/Column_1$
r_1	x_2	$Column_2$	k_2	$nDT_2/Column_2$
...	...	...	...	
r_n	x_n	$Column_n$	k_n	$nDT_n/Column_n$

Inputs (list)

data	K	u
$Column_1$	k_1	u_1
$Column_2$	k_2	u_2
...	...	...
$Column_n$	k_n	u_n

Outputs (list)

data	K	y
$Column_1$	k_1	y_1
$Column_2$	k_2	y_2
...	...	...
$Column_n$	k_n	y_n

Figure 2. The dynamic blocks in the TabSim

Simulation

time		t_i	
steps		n_i	
step		dt_i	
method		nm_i	
initialisation		true/false	
deviation		true/false	
error		err_i	
iterations		$iter_i$	
sheet		$TabSim20_i$	
integration		$integ_i$	
grid-integration		$grid_i$	
Delay		$delay_i$	
weight-delay		$wdelay_i$	
input		$input_i$	
output		$output_i$	

Figure 3. Simulation-setting the parameters

2.3 Example 1: Simulation of the Modified Predator-Prey Model in the TabSim

The models carried out by Lotka and Volterra have been used nearly for a century. Many extensions have been accepted to adjust the model outputs to the real systems. As an example, one of the models extended by time delays describes ordinary differential equations [Matějíček, 1994, 1995]:

$$\begin{aligned} \frac{dx_1}{dt} &= a_1 x_1 - b_1 x_1^2 - c_1 x_1 x_2 + m_1 \\ \frac{dx_2}{dt} &= -a_2 x_2 + c_1 r_1 r_2 + m_2 \end{aligned} \qquad (3)$$

$$r_1(t) = \int_0^{t_D} w_1(\tau) x_1(t-\tau) d\tau$$

$$r_2(t) = \int_0^{t_D} w_2(\tau) x_2(t-\tau) d\tau$$

where x_1 is the abundance of the prey, x_2 is the abundance of the predator, a_1, a_2, c_1, c_2 are the parameters. The simulation of the model (3) is carried out in figure 4. The variables r_1 and r_2 represent the delayed state variable x_1 and x_2. The relations are described by integration of delayed value, which are multiplyed by the weight coefficients w. The relations are realized in TabSim with the modified time delay block. The time dependent variables m_1 and m_2 include the migrations. All the data (initial values for the time delays with their weight coefficients, values of the inputs and outputs) are available in the blocks of data. The user can directly set the data and create the graphs. The models and their data structures can be managed through the user interfaces, forms that can include selected model parameters and output graphs. One of the user interfaces, which are developed in the spreadsheet to simplify the setting of the model parameters and outputs, are illustrated in figure 5.

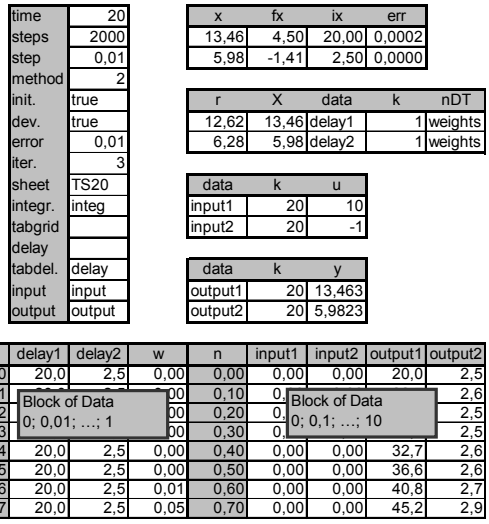

Figure 4. The structure of the modified predator-prey model in the TabSim environment

Annotations to figure 4:
The table of simulation - *time*: the actual time of simulation; *steps*: the time period of simulation expressed in the number of the integration steps; *step*: the integration step; *method*: numeric method; *init.* (true): simulation is started with the initial values of the state variables; *dev.* (true): the relative errors are calculated; *iterations*: the number of iterations in some numerical methods; *sheet*: the name of the spreadsheet list, which contains the model; *integr.*, *tabdel.*, *input* and *output*: the names of the tables that contain individual TabSims's blocks;

The table of the integrations - *x*: state variables; *fx*: $a_1 x_1 - b_1 x_1^2 - c_1 x_1 x_2 + m$; $-a_2 x_2 + c_1 r_1 r_2 + m_2$; *ix*: the initial conditions; *err*: the relative errors.

The table of the time delays - *r*: the delayed state variables; *x*: the actual state variables; *data*: the name of the tables with the last state variables; *k*: the communication interval in the tables; *nDT*: the name of the table with the weights coefficients.

The table of the inputs - *data*: the name of the table with the inputs values; *k*: the communication interval; *u*: the actual values of the inputs;

The table of the outputs – *data*: the name of the table with the outputs values; *k*: the communication interval; *y*: the actual values of the outputs.

The bottom tables – *n*: time; *delay1*, *delay2*: the tables with the last state variables; *w*: the table with the weight coefficients; *input1*, *input2*: the tables with the input values; *output1*, *output2*: the tables with the output values.

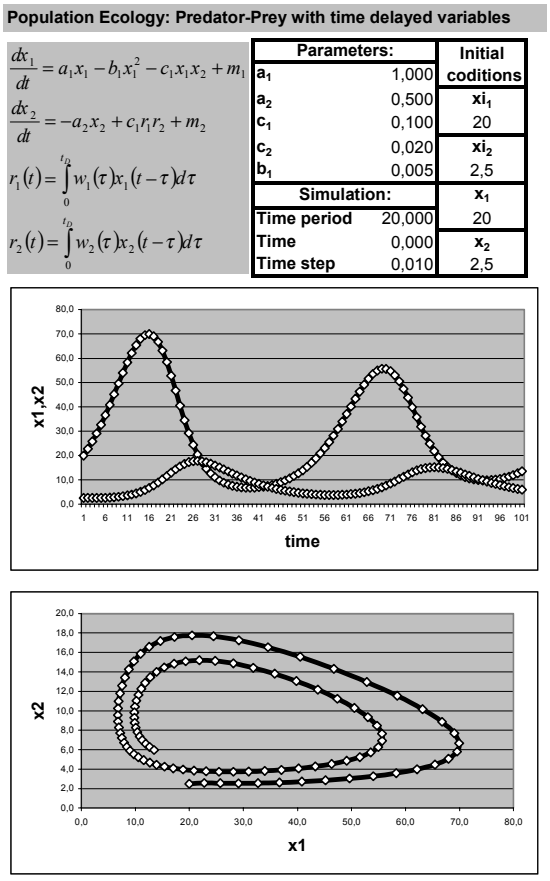

Figure 5. The user interface of the modified predator-prey model in the TabSim

2.4 Example 2: Simulation of the Population Model Described by Logistic Growth with Passive Diffusion and Advection

As an example, the standard model of the population logistic growth is extended by diffusion and advection phenomena [Okubo, 1980]. The model can be described by the partial differential equation:

$$\frac{\partial S}{\partial t} = rS\left(1-\frac{S}{K}\right) + D\left(\frac{\partial^2 S}{\partial x^2} + \frac{\partial^2 S}{\partial y^2}\right) + v\frac{\partial S}{\partial y}. \qquad (4)$$

The solution of the equation has to be translated to the discrete formula, which can be calculated by the TabSim. The translation represents the discrete modification of the diffusion and advection part of the equation (4):

$$\frac{dS_{i,j}}{dt} = r_{i,j}S_{i,j}\left(1-\frac{S_{i,j}}{K_{i,j}}\right) + \qquad (5)$$
$$+D_{i,j}\frac{S_{i-1,j}+S_{i,j-1}-4S_{i,j}+S_{i+1,j}+S_{i,j+1}}{h^2} + v_{i,j}\frac{S_{i+1,j}-S_{i,j}}{h}$$

where the $S_{i,j}$ is the abundance of species in the cell, which is in the i^{th} row and in the j^{th} column. $r_{i,j}$ represent the growth ratios. $K_{i,j}$ are the capacities of species in the specified cells. $D_{i,j}$ and $v_{i,j}$ are diffusion and advection parameters. The constant h represents the distance between neighbour cells. The simulation of the modified model (5) is illustrated in figure 6. The grids on the right side contain consecutively state values, functions and initial conditions. The following tables below include the corresponding parameters of the model (growths, capacities, diffusions, advections). The initial abundance in each cell is set to1000 individuals, which is the capacity. To examine the model, the initial abundance in the middle of the range is changed to 3000 individuals.

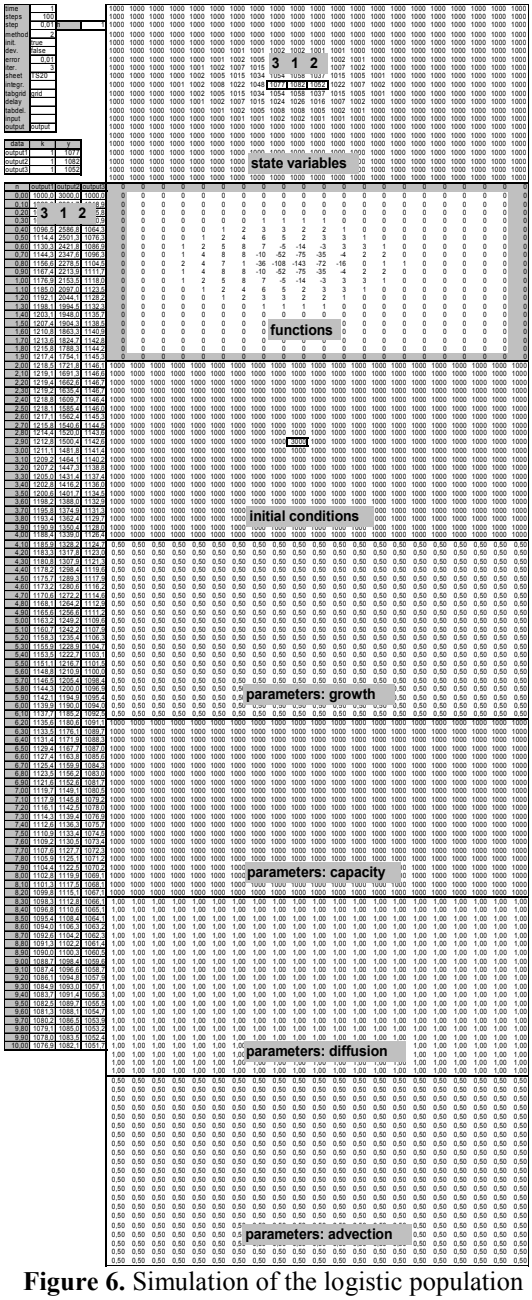

Figure 6. Simulation of the logistic population growth with the passive diffusion and advection

Annotations to the model implementation (figure 6):
The table of simulation: *time*, *steps*, *step*, *method*, *init.*, *dev.*, *error*, *iter.* and *sheet* are similar to the parameters in the model in figure 4.; *tabgrid* and *output*: the names of the tables that contain individual TabSims's blocks;

The table of the grid integrations - the integrations are located in the upper tables on the right side (the state variables, the functions and the initial conditions);

The table of the outputs - the outputs are located on the left side.

The output of the simulation is illustrated in figure 7. The upper graph represents final abundance in each cell of the range. The bottom graph contains the time changes of the abundances in the middle cell and the two neighbour cells (marked in figure 6). The simulation of the logistic growth with passive diffusion and advection represents one of the demonstration versions. The real simulation can be enlarged and complemented by other dynamic and static relations. The static relations can be built-in or user functions. In this case, the user manages the simulation directly. Also the user interface can be built to simplify management of the model. The more complex the model the less the efficiency of the calculation in the environment of the TabSim. The improvement can produce calculations of the model in the compiled modules or the standalone applications. On the opposite side this way usually decreases flexibility of simulation [Matějíček, 2000].

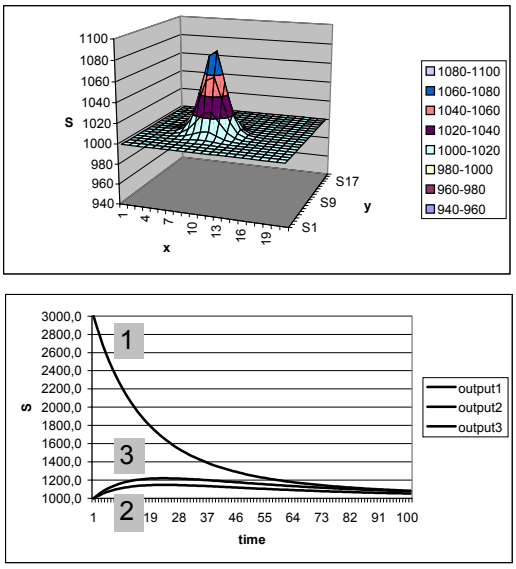

Figure 7. The output of the logistic growth with the passive diffusion and advection

3. CONCLUSIONS

There is a number of simulation systems on the market, which include sophisticated numerical algorithms and can solve ecological models. In spite of this, the TabSim has been developed to simplify the model definition and data management. Because the spreadsheets represent known and popular environment for data processing, the user can concentrate more on simulation and validation of the model. All the input and output data together with graphs and comments can be saved in the standard way of the spreadsheet processors. The previous examples demonstrate simulations of elementary problems, which are use to be extended by other interactions and phenomena. In spite of the TabSim can be used for simulation of large models with the limits of the spreadsheet, the efficiency of calculation depends on computer equipment.

4. ACKNOWLEDGEMENTS

The paper was carried out in the frame of the project VZ MSM 113100007, which is generally focused on modelling natural environment and society. The project primarily associates geographic departments of Charles University in Prague.

5. REFERENCES

Bennett, B.S., *Simulation fundamentals*, Prentice Hall, 329 pp., London, 1995.

Hannon, Ruth, M., *Dynamic modeling*, Springer-Verlag, 248 pp., New York, 1994.

Matějíček, L., Simulation of selected continuous models of ecological systems on personal computers, Ph.D. thesis, Charles University, Prague, 1994.

Matějíček, L., Simulation of selected predator-prey models with ACSL/PC, *Acta Universitatis Carolinae Environmentalica 9*, 39-47, 1995.

Matějíček, L., *Dynamic modeling of the Prague environment with GIS and remote sensing*, paper presented at 5th International Symposium and Exhibition on Environmental Contamination in Central and Eastern Europe, Prague, Czech Republic, September 12-14, 2000.

Okubo, A., *Diffusion and ecological problems: mathematical models*, Springer-Verlag, 254 pp., Berlin, 1980.