# *Gepasi* version 2

© 1989, 1992 Pedro Mendes

*Gepasi* is a software package for the simulation of metabolic models, both of their transient (dynamic) and steady states. The simulations produce values for the concentrations of metabolites and the magnitudes of fluxes starting from an initial state. The parameters of these models are the kinetic constants of the steps and the concentrations of metabolites that are either fixed (buffered) or constantly flowing into or out of the system. This package was used to obtain data already published (Mendes *et al.* 1992).

The user interaction is handled by a program that runs under MS-Windows (presently on IBM-PC-compatible computers). This program makes extensive use of menus, dialogue boxes, push buttons, radio buttons, list boxes and other controls. This form of input, accompanied by keyboard short cuts, minimises the time taken by a first-time user to get accustomed to the mechanics of the program. By taking advantage of MS-Windows' own help engine, this front-end program has an extensive help system that covers not only immediate instructions on how to use the package but also has explanations of common concepts in metabolic control, for example, those of *internal* or *external* metabolites. There is also a section with full references to articles, reviews and books covering subjects related to metabolic control and modelling. This feature should ensure that *Gepasi* will be a useful tool for education.

From the user's point of view, the model is specified in two parts: first, one must input the stoicheiometric structure of the model, the kinetics associated with each step and which metabolites are modifiers of which reactions; then one must specify the numeric values of the kinetic constants and the initial metabolite concentrations. The numeric data were separated from the structural data since one is likely to input many different sets of parameters for the same structural model. This input is clearly separate as it is done in two separate windows, of which there can be several instances (each one pertaining to a different model) on screen at the same time. The output of the simulated data is formatted as a plain text file, for ease of reading, and/or in data files in columnar format. For the latter the user can specify which element (parameters or variables) will appear in each column. The columns can be separated by spaces, tabs or commas so that this file can be imported into most popular spreadsheet or graphics programs. There is also control over the column width and whether titles are included at the top or bottom of the columns.

*Gepasi* 2 has implemented one feature that is particularly useful for an extensive study of a

model. The user can instruct the simulator to scan various parameters. The simulator will produce several simulations in sequence (each with different values for those parameters) and put the output on different rows of the same file, effectively producing a map of the behaviour of the model (its variables) in a region of parameter space. Virtually all parameters of the model can be scanned (but unless the model is small they will not *all* be scanned as the number of simulations would become prohibitive). These 'scans' can be done with linear or logarithmic intervals: lower and upper limits and the number of intervals must specified for each parameter. They can also be done by assigning random values (with linearly or logarithmically uniform distributions) for the parameters within lower and upper limits. In this case a total number of simulations is selected. The 'random scans' are useful when one is trying to characterise the model in many dimensions. In this case it is the total number of simulations that is set, not a grid mesh. If one were to scan n parameters, each with 10 regular intervals, one would effectively be asking *Gepasi* to produce $10^n$ simulations, while one could instead have, for example, 100 simulations with these n parameters being assigned random values (within the defined boundaries). This feature is useful for producing training sets for artificial neural networks that would 'learn' the system's properties.

While one may be interested in scanning some parameters freely across parameter space, one might want to restrict others to take values as a function of those that are being scanned. This is useful for cases in which one wishes to change some kinetic constants whilst adjusting others to maintain the same equilibrium constants. In *Gepasi* these functions are called links and at present there is only one type of link available: the linked parameter as the product of a constant and the master parameter. In the future other link functions will be implemented.

The data for transient analysis of the model are obtained by integration of the ordinary differential equations (ODE) that describe the change of concentrations with time. These are determined from the stoicheiometric structure of the pathway and the rate laws of each step. The data for steady states are calculated by setting all equations equal to zero and solving the resulting system of (non-differential) equations (Hofmeyr 1986). The algorithm used for the integration of the ODE is the Livermore Solver of Ordinary Differential Equations with Method Switching (LSODA, Petzold 1983), which uses a procedure that detects if the system of ODE is stiff (Gear 1971) or not and uses the most efficient method for each case, respectively the Adams or backward differentiating formula (BDF) methods (Gear 1971, Hindmarsh 1980). LSODA is impressively fast compared to the previous algorithm used in *Gepasi* version 1 (Mendes 1992) which was itself much faster than the well-known Runge-Kutta. This speed made the use of integration combined with the damped Newton method (Comte & de Boor 1980) feasible to solve the steady state. Whenever the Newton method fails to converge (either by reaching the limit of iterations or by getting stuck in local minima) the estimate is used as a starting point and the system is integrated for 10 units of time, then the solution is used again as the estimate for the Newton method. If it fails once more, it will be integrated for 100 units of time, and this process goes on for 10 iterations (over $10^{10}$ units of time). If no steady state is reached the program informs the user about it. This method of getting out of local minima or improving the initial estimate is much more robust than that of retrying the Newton method by randomly perturbing the initial guess (as in the former case it is in reality *driving* the guess towards the solution and not making it jump in a random direction).

Steady states can be analysed with metabolic control analysis (MCA, Kacser & Burns 1973, Heinrich & Rapoport 1974, Cornish-Bowden & Cárdenas 1990). *Gepasi* calculates the elasticity coefficients from the partial derivatives of the rate equations and control coefficients using the method described by Reder (1988). It is able automatically to detect moiety-conserved cycles (Reich & Sel'kov 1981, Hofmeyr 1987) by reducing the stoicheiometry matrix using Gaussian

elimination. Transition times (Easterby 1981) are also computed. The user can select from any elasticities, concentration- or flux-control coefficients for output, as well as products of elasticities and control coefficients.

*Gepasi* is composed of two separate entities: a user interface that handles all the interactions with the user, and a simulating engine that performs the actual numeric calculations. This modular structure of the software was designed to keep the numerical module portable (there may be the need to have simulations running on high speed computers) whilst still having a front-end supported by a graphical user interface (GUI). It leaves open the possibility of the development of a language-based description of the models and retaining the GUI based interface. At present, though, only the GUI has been implemented.

### *References:*

Comte, SD & de Boor, C (1980), *Elementary numerical analysis. An algorithmic approach.*, McGraw Hill International Book Company, Singapore.

Cornish-Bowden, A & Cárdenas, ML eds. (1990), *Control of Metabolic Processes.* Plenum Press, New York.

Easterby, JS, (1981), A generalized theory of the transition time for sequential enzyme-reactions. *Biochem. J.* **199**, 155-161.

Gear, CW (1971), *Numerical initial value problems in ordinary differential equations.* Prentice-Hall, Englewood Cliffs, N.J.

Hindmarsh, AC (1980), LSODE and LSODI, two initial value ordinary differential equation solvers. *A.C.M. Signum Newsletter*, **15**, 4 December, 10-11.

Heinrich, R and Rapoport, TA (1974), A linear steady-state treatment of enzymatic chains. General properties, control and effector strength. *Eur. J. Biochem.*, **42**, 89-95.

Hofmeyr, J-HS (1986), Steady-state modelling of metabolic pathways: a guide for the prospective simulator. *Comput. Applic. Biosci.*, **2**, 5-11.

Hofmeyr, J-HS, Kacser, H & Van der Merwe, KJ (1986), Metabolic control analysis of moiety-conserved cycles. *Eur. J. Biochem.* **155**, 631-641.

Kacser, H & Burns, JA (1973), The Control of Flux. *Symp. Soc. Exp. Biol.* **27**, 65-104.

Mendes, P (1992), GEPASI - A Simulator of the Dynamical Behaviour of Biochemical Pathways. in *Biothermokinetics* (Westerhoff, HV ed.), Intercept, Andover, U.K., in the press.

Mendes, P, Kell, DB & Westerhoff, HV (1992), Channelling can decrease pool size. *Eur. J. Biochem.* **204**, 257-266.

Petzold, L (1983), Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. *SIAM J. Sci. Stat. Comput.*, **4**, 136-148.

Reder, C (1988), Metabolic control-theory: a structural aproach. *J. Theoret. Biol.* **135**, 175-201.

Reich, JG and Sel'kov, EE (1981), *Energy metabolism of the cell. A theoretical treatise.* Academic Press, London.