Evaluation of Steady-state Kinetic Parameters of Metabolic Pathways using Neural Networks

Herbert M. Sauro and Douglas B. Kell

Department of Biological Sciences, University College of Wales Aberystwyth, Dyfed SY23 3DA, UK

A question sometimes posed with regard to the study of metabolic systems is whether it is possible to obtain the values of a pathway's parameters, i.e. K_ms, V_{max}s etc., from intact systems. The usual method has traditionally been to obtain such parameters from *in vitro* studies; however, it is doubtful that the values determined *in vitro* accurately reflect the *in vivo* situation. On the other hand, the variables of intact pathways, namely the flux and metabolite concentrations, are more easily estimated. The question then arises as to whether it is possible, given a set of measured variables, to determine the set of parameters on which they depend. Clearly, it is possible to do the reverse, i.e. by computer simulation and a set of parameters one can determine the variables.

The approach which seeks to relate the variables of a system to its parameters is sometimes known as system identification and has been widely applied to chemical engineering and other non-linear dynamic problems. The aim of this project is to train an artificial neural network (Amit, 1989, Maren *et al.*, 1990, Müller & Reinhardt 1990, Freeman & Skapura, 1991, Zurada, 1992.) to relate the variables of an intact metabolic pathway to the parameters of the system. If the net successfully learnt to reflect the correct parameters when presented with the variables, the problem would have been solved.

The approach that is used is as follows. Select the pathway of interest, say a linear chain, and build a computer model of the pathway. Perform many simulations with the model using random sets of parameters to obtain the steady-state values for the variables. Using the simulation data, train a neural net to relate the variables (the inputs) to the parameters (the outputs). Next, present the net with 'random' (or experimental) variables and ask it for the parameters. A check on the network's predictions can be done by running a simulation with the parameters provided by the network and seeing if they generate the variables used as the input to the net. (Note that this check will not quarantee that the parameters predicted from experimental data are the real values, only that the predicted parameters are a consistent set.) The result would be that we could in fact obtain the (enzymatic) parameters of a metabolic network by measuring the variables alone (see Figure 1). The purpose of the present study was therefore to establish whether this might indeed be the case.

By way of illustration, and for computational simplicity, we concentrate here on a simple, two-step linear pathway, with each enzyme (1 substrate/1 product) possessing reversible Michaelis-Menten kinetics. The system is shown below.

$X_0 \rightarrow S_1 \rightarrow X_1$

where X_0 and X_1 are used to indicate that the concentrations of these metabolites are held constant (a necessary requirement for the maintenance of a steady state).

Generation of the data

The net was trained by varying the V_{max} values of each enzyme and both

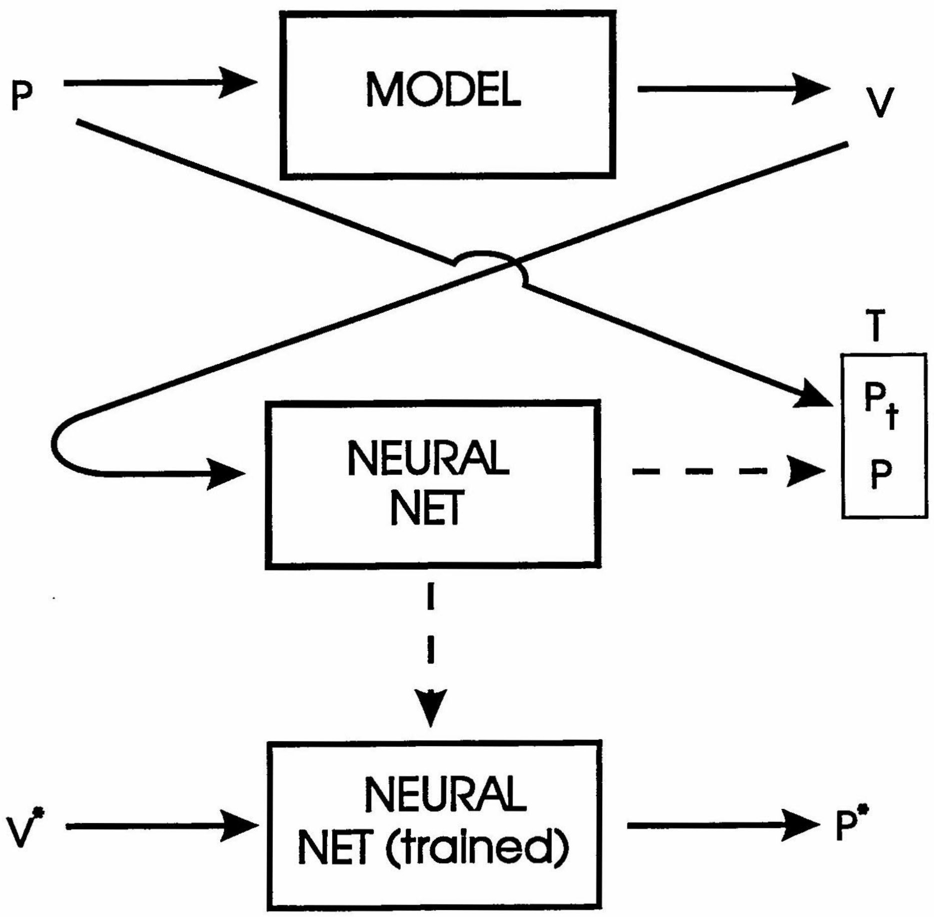


Figure 1 Diagram showing the general approach to the problem as described in the text. Notation is as follows. *V* refers to the variable matrix and *P* the parameter matrix. The box labelled *T* comprises the training block, made up of the training set P_t, derived from the model and the network's present estimates of the parameters. V^* refers to the 'experimental' intput variable matrix and *P* refers to the output parameter set derived from the experimental set.

BINARY Vol. 4 © 1992 Academic Press Limited

forward K_m values. The reverse K_ms were held constant. The concentration of S_1 and the steady-state flux were recorded for each parameter set. In addition, each parameter set was applied using three different concentrations of the starting metabolite X_0 so that we could obtain three sets of variables for the same set of parameters (in an experimentally realizable fashion), and so aid the training process. The parameters of the system were varied as follows. Each parameter was first assigned a uniform random number, v, between 0 and 2.0 which was then used to generate a non-skewed distribution spanning two decades using the formula 10^v. The reason why the parameters were generated in this manner was so that the distribution of values would not be confined to the upper decades of the range and thus there would be a roughly equal number of random values between 1 and 10 as there would be between 10 and 100. In this way, up to 200 random sets of K_m and V_{max} values were generated, from which the steady-state concentrations of S_1 and the fluxes were obtained. In addition, each set of random parameters was applied to three values of X_0, namely at 1.0, 10.0 and 100.0. In all, 600 different sets of data were generated and presented to the neural network.

The neural network

All neural network runs were performed on an IBM PC clone (486 incorporating an accelerator board containing an AT & T DSP32C digital signal processing chip) using the software package *NeuralDesk* (Neural Computer Sciences, Totton, Southampton, UK) running under *Windows* 3.1. This set-up is by no means unique and the neural network runs could just as easily have been done using a different computer (Macintosh, Unix workstation, etc.) and different software.

Training was done on a fully connected feedforward net which had nine inputs (corresponding to the three sets of three variables from each value of X_0), four outputs corresponding to the parameters, and one hidden layer of four units. All data presented to the net were first transformed using $\log_{10}$ and then normalized to fall between 0 and 1.0. Training was performed using a stochastic back-propagation algorithm (Neural Computer Sciences, 1991) until the mean square error fell to below 0.01.

An example

Fixed parameters were as follows:

K_{eq1} = 567.0
K_{eq2} = 13.4
X_1 = 0.1 mM
K_{mr1} = 23.4 mM
K_{mr2} = 12.3 mM

The two rate laws governing each reaction were of the form:

$$v = \frac{V_{max}/K_{mf}(S_1 - S_2/K_{eq})}{(1 + S_1/K_{mf} + S_2/K_{mr})}$$

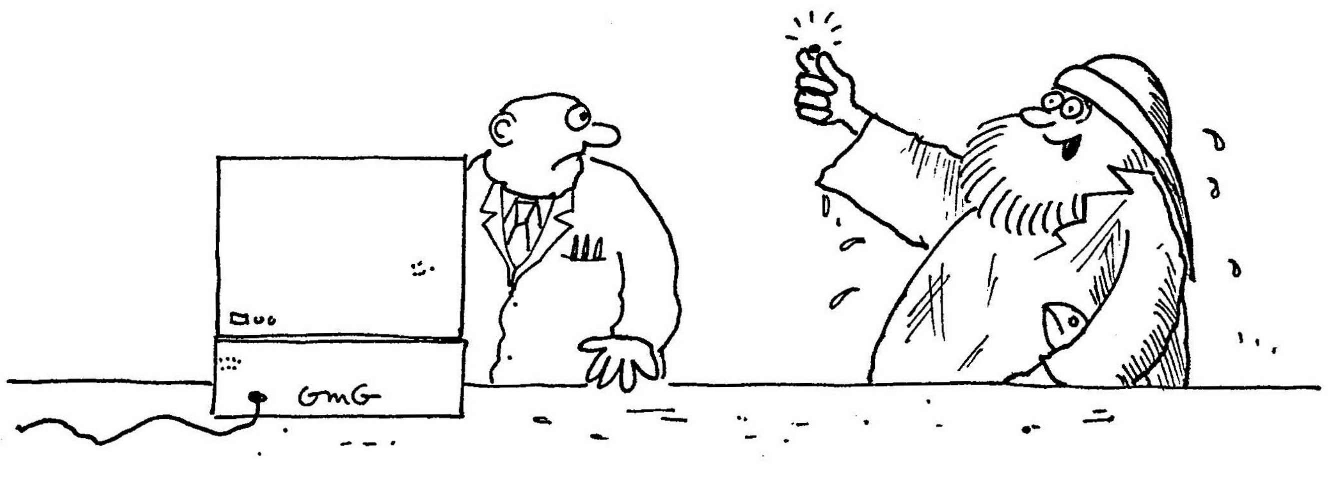
The steady state of the pathway was evaluated using a Newton-based iterative method (written in-house using a Zortech C/C++ compiler) and the test for convergence was the sums of squares deviation of the rate of change of S_1 from zero. The different parameter sets were calculated according to the method described above. The results of each simulation were written out to two files, one designated the *input file* which contained the variables and another designated the *output file* which contained the parameters. The net was then trained using these files. Convergence was very rapid (of the order of seconds) and further test data (a separate set from those used in the training phase) confirmed that the net had generalized.

Acknowlegement

We are grateful to the Wellcome Trust for financial support.

References

AMIT, D.J. (1989). *Modelling Brain Function.* Cambridge: Cambridge University Press.
FREEMAN, J. A. & SKAPURA, D.M. (1991). *Neural Networks. Algorithms, Applications, and Programming Techniques.* Addison-Wesley Publishing Company.
MAREN, A., HARSTON, C. & PAP, R. (1990). *Handbook of Neural Computing Applications.* London: Academic Press.
MÜLLER, B. & REINHARDT, J. (1990). *Neural Networks. An Introduction.* Springer-Verlag.
NEURAL COMPUTER SCIENCES (1991). *Neuraldesk's User's Guide.* Southampton, UK: Neural Computer Sciences.
ZURADA, J.M. (1992). *Introduction to Artificial Neural Systems.* West Publishing Company.



"CHIP AHOY!"