Hybrid modelling of bioprocesses.

Benjamin Jake Hodgson

Thesis submitted for the degree of Engineering Doctorate in Biochemical Engineering of the University of London.

Department of Biochemical Engineering
University College London
Torrington Place
London WC1E 7JE

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

For My Father.

Thesis abstract.

The two traditional approaches to modelling can be characterised as the development of mechanistic models from 'first principles' and the fitting of statistical models to data. The so-called 'hybrid approach' combines both elements within a single overall model and is thus composed of a set of mass balance constraints and a set of kinetic functions. This thesis considers methodologies for building hybrid models of bioprocesses. Two methodologies were developed, evaluated and demonstrated on a range of systems of simulated and experimental systems.

A method for inferring models from data using support vector machines was developed and demonstrated on 3 experimental systems; a Murine hybridoma shake flask cell culture, a *Saccharopolyspora erythraea* shake flask cultivation and a 42L *Streptomyces clavulingerus* batch cultivation. On the latter system the method produced models of similar accuracy to previously published hybrid modelling work. While support vector machines have been widely applied in other contexts this method is novel in the sense that there are no previously published papers on the use of support vector machines for kinetic modeling of bioprocesses.

On 50 randomly created dynamical systems it was shown that the hybrid models produced using the support vector machine methodology were generally more accurate than those developed using feed forward neural networks and that could not be distinguished from models produced using a more computationally demanding method based round genetic programming.

Additionally a Bayesian framework for hybrid modelling was developed and demonstrated on simple simulated systems. The Bayesian approach requires no interpolation of data, can cope with missing initial conditions and provides a principled framework for incorporating *a priori* beliefs. These features are likely to be useful in practical situations where high quality experimental data is difficult to produce.

Contents.

List of figures.

List of tables.

List of Abbreviations.

AI	Artificial intelligence.
AIS	Annealed importance sampling.
BIC	Bayesian information criteria.
CER	Carbon evolution rate.
CSTR	Continuous stirred tank reactor
DCW	Dry cell weight.
DNA	Deoxyribonucleic acid.
DOT	Dissolved oxygen tension.
EKF	Extended Kalman filter.
EMEA	European medicines evaluation agency.
FDA	United states food and drug administration.
FFNN	Feed forward neural network.
GA	Genetic algorithm.
GAMP	Good automated manufacturing practice.
GP	Genetic programming.
IP	Intellectual property.
kPCA	Kernel principal component analysis.
MAP	Maximum a posteriori.
MC	Monte Carlo.
MCMC	Markov chain Monte Carlo.
ML	Maximum Likelihood.
MPC	Model predictive control.
NN	Neural network.
OTR	Oxygen transfer rate.
OUR	Oxygen uptake rate.
PAT	Process analytical technology.
PCA	Principal component analysis.
RBF	Radial basis function.
RNA	Ribonucleic acid.
SVM	Support vector machine(s).
TOC	Total organic carbon.
UKF	Unscented Kalman filter.

Nomenclature

Indexing and dimensions.

$x \in \Re$	Variable x is a single real number
$X \in \Re^n$	Variable X is a real valued column vector of length n $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$
$X \in \Re^{1 \cdot n}$	Variable X is a real valued column vector of length n $\begin{pmatrix} a_0 & \cdots & a_{n-1} \end{pmatrix}$
$X \in \Re^{n \cdot m}$	Variable X is a real valued matrix with n rows and m columns $\begin{pmatrix} a_{0,0} & \cdots & a_{0,m-1} \\ \vdots & \ddots & \vdots \\ a_{n-1,0} & \cdots & a_{n-1,m-1} \end{pmatrix}$
$M_{i,j}$	Matrices are indexed using i as the row index and j as the column index. Indexes start from zero so as to be compatible with C++ code. Thus $\sum_{i=0}^{n-1} w_i x$ is written rather than $\sum_{i=1}^{n} w_i x$.
$M^{(j)}$	Refers to the column vector corresponding to column j of matrix M
$M_{(i)}$	Refers to the row vector corresponding to row i of matrix M

Superscripts

*	Indicates that a variable is derived from measurements
$\overline{x}$	Indicates the mean of a variable.
$\hat{x}$	Indicates a variable is estimated. The estimator depends on the context.

Matrix Functions

$\lvert X \rvert$	Indicates the determinant if X is a matrix or absolute value if X is a real valued variable
$\lVert X \rVert$	Indicates Euclidian norm
$Tr[X]$	Trace of matrix A
X^{-1}	Inverse of matrix A
$X^{-\#}$	Moore Penrose pseudo Inverse of matrix A
X^T	Transpose of matrix A

General state space bioreactor model

t	System time.
$t_s^{\langle i \rangle}$	$\in \Re^{1\times l}$ vector of times at which measurements were taken.
τ	Temporary variable replaced by some time such as current simulation time t or a sample time $t_s^{\langle i \rangle}$
N_ξ	Number of state variables included in model
N_{env}	Number of environmental variables included in model
N_{meas}	Number of measurements of the state made during each batch
N_r	Number of degrees of freedom of the system. i.e number of free reactions
v	$\in \Re^{N_{env}}$ Vector of environmental conditions
ξ	$\in \Re^{N_\xi}$ State vector of bulk concentration in the reactor.
ξ_i	$\in \Re$ Concentration of ith species in the reactor.
ξ_{in}	$\in \Re^{N_\xi}$ State vector of concentration in feed.
$\xi(\tau)$	Continuous function returning bulk concentration $\xi \in \Re^{N_\xi}$ at time τ
$\xi^*_{t=\tau}$	$\in \Re^{N_\xi}$ Measured state vector of bulk concentration in the reactor at time τ referenced by $t_s^{\langle i \rangle}$ e.g. $\sum_{i=0}^{n-1} \xi^*_{t=t_s^{\langle i \rangle}}$.
$\xi^P(t)$	Model prediction of the state at time t.
F_{in}	Feed flow rate in.
F_{out}	Feed flow rate out.
V	Volume of media in reactor.
D	Dilution rate $\frac{F_{in}}{V}$
u	$\in \Re^{N_\xi}$ Vector of net inflow/outflow of species into the reactor
$g(\xi)$	$\in \Re^{N_\xi}$ Vector of net inflow and outflow in gas phase
K	$\in \Re^{N_\xi \times N_r}$ Matrix of linear constraints.
$r(\xi,v)$	$\in \Re^{N_r}$ Vector of kinetic functions determining the rates of the free reactions.
$f(\cdot)$	Unspecified non-linear function returning a vector. Text below may indicate type of function e.g. $\underset{FFNN}{f}(\cdot)$ Is a function determined by a feed forward neural network. $\underset{GP}{f}(\cdot)$ Is a function determined by a genetic programming tree.

Literature survey

N_w	Number of parameters
w	$\in \Re^{N_w}$ vector of parameters
$U(.)$	Utility function reflecting desirability of some outcome
$L(.)$	Loss function reflecting undesirability of outcome e.g. $-U(.)$
$h(\cdot)$	Measurement model for Kalman filter
$p(x)$	Probability density function for x
$p(x\|y)$	Conditional probability density function for x given y
$\left(y^{(u)}, x^{(u)}\right)$	The uth input-output pair where $x \in \Re^n$ $y \in \Re$.
$x^{(1)}_j$	The output of the jth neuron in the ith layer when x is the input to the first layer
N_{inputs}	Number of inputs into the first layer =n
$N_{neurons}$	Number of neurons
N_{layers}	Number of layers in network
$\theta^{(\ell)}_j$	Biase for jth neuron in the ith layer of a feed forward neural network.
$w_{i,j}^{(\ell)}$	The weight connecting i^{th} neuron the ℓ^{th} layer to the j^{th} neuron the ℓ-1th layer.
$\varphi(x)$	Non-linear transfer/activation function
α	Learning rate parameter
β	Momentum term
N_{nodes}	Number of nodes(functions or terminals) in GP tree.
$R[f]$	Expected risk or expected loss due to using function f.
$R_{emp}[f]$	Empirical risk or Loss on training data
F	Hypothesis space, set of all possible functions f could be $f \in F$

Constraint matrix

$\Gamma(\tau)$	The flux of a component at time τ due solely to reaction effects.
r_i	Rate of change of component i, where i is an abbreviation e.g. r_{gluc}
μ	Growth rate
$\hat{\mu}$	Specific growth rate
N_e	Number of conserved elements.
E	Elemental composition matrix $\in \Re^{N_e \times N_s}$
G	Stoichiometric matrix $\in \Re^{N_s \times N_m}$ defining reaction network
ϑ	$\in \Re^{N_r}$ Vector of fluxes through each lumped reaction
$\hat{\xi}^*(t)$	Continuous signal for state obtained from measurements
$\hat{\Gamma}^*(t)$	Estimate for $\Gamma(\tau)$ obtained from interpolated measurements

$\eta_{t=t_s}$	$\in \Re^{N_r}$ Cumulative consumption/production at time t_s
E	Permutation i.e. reordering of rows
M	Matrix of principle components
$\hat{K}$	Pseudo stoichiometric matrix (constraint matrix estimated by regression method)
N_{pca}	Number of principle components.
$p(\xi,t)$	$\in \Re^{N_{pca}}$ Vector of Kinetic functions defined in principle components space.
N_{train}	Number of training batches
N_{test}	Number of testing batches
$\underset{\forall Train}{\overline{Fit}}\left(\begin{smallmatrix}Test\ System\\ Model\ type\end{smallmatrix}\right)$	Average fit of model of type 'model type' to training data on test system 'test system'

SVM methodology

$\delta(\xi)$	Function which is 1 if all substrates consumed by the relevant reaction have a non zero concentration in the bioreactor and 0 otherwise.
w	Weight vector
b	Bias term
$\Phi(x)$	Non-linear mapping
x	coordinate in input space
z	coordinate in feature space $z=\Phi(x)$
$y^{(i)}$	Output value of data at point $x^{(i)}$
$f(x^{(i)})$	Prediction of SVM at point $x^{(i)}$
C	Cost/ Regularisation hyper parameter
ε	Range of acceptable error for ε insensitive loss function
σ	Radial basis function width parameter
$K(x^{(i)},x)$	Kernel function 'dot product in feature space'
$k(x^{(i)},x^{(j)})=\Phi(x^{(i)})\cdot\Phi(x^{(j)})$	
S_i, S_i^*	Support vectors above and below e insensitive zone
$\alpha_i,\alpha_i^*,\gamma_i,\gamma_i^*$	Lagrange multipliers corresponding to constraints

Hybridoma model

V	Volume
F	Feed flow rate subscript (in/out)
C_{Xv}	Viable cell
C_{Xd}	Non viable cell
C_{AB}	Antibody
C_{GLC}	Glucose
C_{GLN}	Glutamine
C_{LAC}	Lactose

C_{AMM}	Ammonia
$Y_{a/b}$	Yield of 'a' on 'b'
μ	Specific growth rate
μ_d	Specific death rate
Q_{ab}	Antibody production rate

Bayesian section.

$H_i(x_i, w)$	Output of model H_i given input x_i and parameter values w
$p(w\|H_i)$	Prior probability of parameters given model structure H_i
$p(D\|w, H_i)$	Likelihood of parameters w and model H_i given data D.
$p(w\|D, H_i)$	Posterior probability of parameters given model H_i and the data D.
w_{MAP}	Maximum posteriori values of parameters $= \arg\max_w p(w\|D, H_i)$.
$p(D\|H_i)$	Evidence for model H_i.
$\hat{z}$	Estimate of the evidence.
σ_i	Measurement error for series i.
$V_{\sim}$	Measurement error covariance matrix.
V_w	Parameter uncertainty covariance matrix.
$E[f]$	Expectation of f.
$\hat{\Theta}$	Expected value
w^t	Individual sample indexed by t.
$w^t \sim p(w)$	w^t is drawn from probability distribution $p(w)$.
N_s	Number of Monte Carlo samples.
s	Expected length scale of parameters
x_*	Uncertain process conditions.
$q(w)$	Importance distribution.

Notes:

1) Curly brackets are sometimes used $\underbrace{\overbrace{\alpha}^{variable} \overbrace{\times}^{multiply} \overbrace{\beta}^{variable}}_{comment}$ these do not alter the meaning of equations and are simply used to comment on terms in important equations.

2) As well as the above nomenclature section, symbols are defined in the text. For example while σ is always a variance depending on the context its precise meaning can be to variance of; measurement error, radial basis function, Gaussian kernel function

“ Life is very strange” said Jeremy.
“Compared with what?” replied the spider.

”

-Norman Moss

Acknowledgements.

I would like to thank Sam Demby, Misty Ushio from UCL and Hans Roubos, Rudd Leuden from DSM for making available their data without which this thesis would be mere algorithms. The EPSRC for funding this project. My supervisor Frank Baganz for giving me both freedom and guidance. Ranna Patel for persuading me to do this in the first place. Chris Taylor and David Sweeny for their code and conversation. Ron Lee of Brunel University for his advice and pub lunches. My mother for putting up with me being a student indefinitely. And finally everyone at UCL especially Colin Jaques, Helen Irvine and Preben Kraben of the Cloisters office for listening to my weird ideas.

1 Introduction

Modelling is central to engineering design, optimisation and control. However, unlike most other areas of engineering, mathematical models are not widely applied in the bioprocess industries. Due to the overwhelming complexity and heterogeneous nature of biological systems they cannot in general be reduced to tractable models derived from physical laws. This thesis considers the problem of inferring useful dynamic models of the behaviour of bioprocesses from available data and knowledge.

This chapter provides a gentle introduction to bioprocesses and bioprocess modelling. The scope of and structure of this thesis is then outlined.

1.1 An introduction to Bioprocesses.

1.1.1 General introduction.

Biotechnology is an important industry producing a wide range of products from Monoclonal antibodies to bulk chemicals. Similar systems can be found outside the biotechnology industry in waste water treatment, brewing and bioremedification.

Bioprocesses consist of the deliberate and controlled cultivation of micro organisms or cell culture. The objective can be to produce the biomass itself (bakers yeast, tissue engineering) some product of cellular processes (antibiotics, monoclonal antibodies) or to break down some substance (waste water treatment). In order to successfully operate the process, necessary nutrients and growth factors must be provided and the environmental conditions (pH, temperature, dissolved oxygen) controlled.

This is accomplished in a bioreactor, typically a stirred vessel or airlift reactor although more exotic designs such as wave bioreactors exist and indeed some other systems such as landfills can be viewed as bioreactors. (Dunn et al(2003)) A diagram of a stirred tank bioreactor is shown in Figure 1. Typically available online measurements of the process include: pH; temperature; dissolved oxygen (DOT) and measurements of carbon dioxide and oxygen concentrations in the inlet and outlet gas[1]. Measurements of biomass and media components are obtained by periodic sampling of the broth and subsequent laboratory analysis although online instruments

[1]The differences between the concentrations in the inlet gas and outlet gas of oxygen and carbon dioxide are used to calculate the oxygen uptake rate (OUR) and carbon dioxide production rate (CER).

for measuring key components are becoming more common[2]. Bioreactors can be operated in batch, fed-batch or continuous mode.

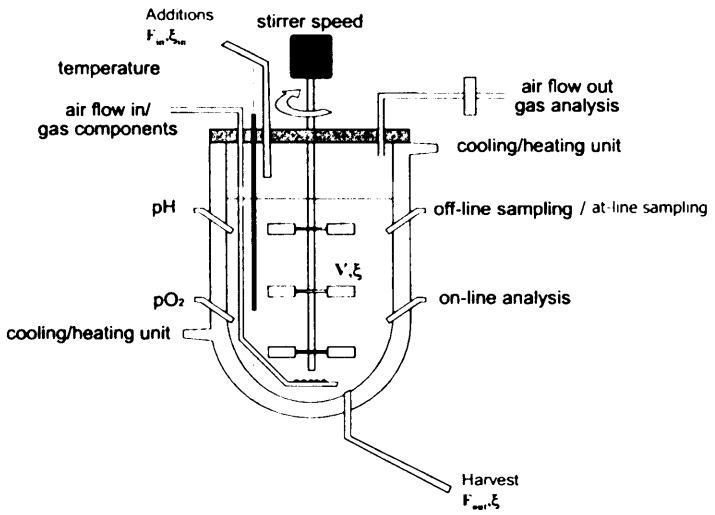


Figure 1. Diagram of a stirred tank bioreactor.

1.1.2 Operating modes.

In batch mode, the bioreactor volume V is constant ($\underbrace{F_{in}}_{additions} = \underbrace{F_{out}}_{harvest} = 0$).Classically the process dynamics of batch processes are characterised by 4 physiological stages (Figure 2): a lag phase in which the cell adapts to new media conditions; a growth phase in which biomass increases exponentially; a stationary phase which is entered when substrates become depleted and the growth rate declines. Finally the death phase is entered where cells die due to lack of nutrients and the build up of harmful by products.

In fed batch processes feed is added during operation ($F_{in} \neq 0, F_{out} = 0$) the volume increases and after product has accumulated the full reactor contents are harvested. Controlled feeding can be used to extend the productive period and increase the maximum biomass concentration. This strategy is particularly effective where high substrate concentrations are detrimental to performance for example where high glucose concentrations cause the production of acetate due to overflow metabolism.

[2] A key driver for increased on line measurement and modelling will be the FDA's process analytical technology initiative. (See Appendix A1.)

In continuous operation feed is added and product is removed continuously ($F_{in} \neq 0, F_{out} \neq 0$). This allows greater control for example; in chemostat operation the concentration of a limiting substrate is used to determine the specific growth rate. Since substrate concentrations are maintained and harmful by-products are removed production time can be greatly extended. However, continuous process operation is relatively rare due to problems of contamination risk and strain instability leading to production strain being out competed by fast growing mutants.

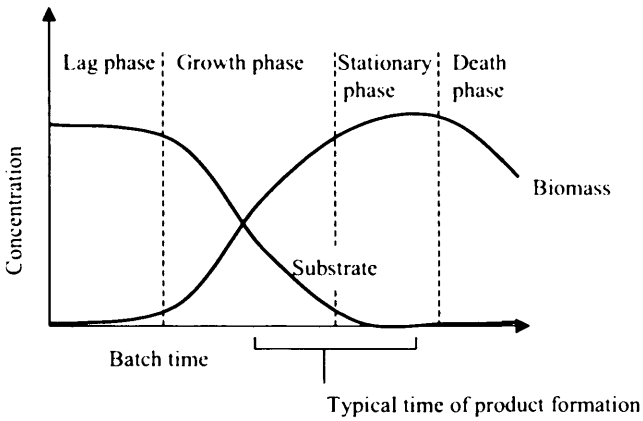


Figure 2. Classical batch profile.

1.2 Bioprocess modelling

1.2.1 General issues.

Bioprocesses are deterministic and, in principle, if the current state of the system and if the laws governing its behaviours were fully known then the future state of the system could be predicted.

Classically this can be represented as a system of differential equations relating the change in the system state vector ξ composed of the concentrations of each species in the bioreactor to some function of the current[3] state, environmental conditions v and time varying control actions u such as feeding.

[3] Extending this so that the dynamics depend on the system history during the last n time steps rather than just the current state leads to a more general system of delay differential equations (DDE's) $\frac{d\xi}{dt} = f(\bigcup_{\tau=(t-n\Delta t)}^{t} \xi(\tau), v(\tau), u(\tau))$ These are rarely used to model bioprocesses although they are interesting for population age balances (Bortz and Nelson(2004)). There is no theoretical reason why the techniques developed in this thesis cannot be extended to DDE's although the numerical integration is somewhat more problematic (Yang et al(2005)).

$$\frac{d\xi}{dt} = f(\xi, v, u) \tag{1.1}$$

The problem is that in reality biological systems are hugely complex and are not fully understood. Analysing the cell systems to determine the true underlying mechanisms may be impossible because many metabolic mechanisms are still unknown or cannot determined from experimental measurements. Therefore to paraphrase Bailey(1998) any model must necessarily be a modest approximation.

Even if a perfect model could be formulated there would still be serious issues. As Palsson(2002) states *"evolution changes the numerical values of kinetic constants over time. In addition, in an out bred population we could have a perfect in silico model for one individual but it would not apply to other individuals due to the polymorphism in the genes and therefore non-identical kinetic parameters"*. Detailed mechanistic modelling is not possible or even desirable; rather models should use the available data and knowledge to capture key features of the system and facilitate decision-making.

1.2.2 Approaches to modelling.

Modelling building methodologies can be divided into three categories; *'white-box'*, *'black-box'*, and *'hybrid/grey-box'* strategies, which differ with regard to roles played by roles played by mechanistic knowledge and data. The three approaches are briefly described below. Chapter three will review these in more detail.

White box modelling.

The white box approach typically begins by deriving a dynamic mass balance model of a perfectly stirred tank (although more detailed physical models using computational fluid dynamics to incorporate the effect of imperfect mixing have also been proposed Royce(1993)). Stoichiometric constraints $K \in \Re^{N_c \times N_r}$ are then introduced on the basis of information about the stoichiometry of the reaction system. The specific mechanism of each reaction $r(\xi, v) \in \Re^{N_r}$ is then determined and an appropriate mathematical description of the reaction kinetics selected from literature. Finally the parameters[4] of the kinetic models are determined on the basis of experimental data.

[4] For the model to be mechanistic rather than semi empirical the parameters should have physical meaning.

General state space representation

The majority of white box models found in literature can be described by a model of the following general form capable of representing most published white box models (Chen et al(2000); Galvanauskas et al(1997)).

$$\frac{d\xi}{dt} = \underbrace{Kr(\xi,\nu)}_{kinetics} - \underbrace{D\xi - g(\xi) + F\xi_{in}}_{transport} \tag{1.2}$$

This is simply a system of differential equations where state $\xi = (\xi_1, \xi_2, ..., \xi_n)^T \in \Re^{N_c}$ is the state vector of species concentrations in the reactor, (assuming perfect mixing). $\nu \in \Re^{N_{env}}$ is a vector of directly controlled or uncontrollable environmental variables such as temperature.

The first term $Kr(\xi)$ represents the biological and biochemical conversions taking place in the reactor. This term further breaks down into a set of linear constraints due to conservation relations written in matrix form $K \in \Re^{N_c \times N_r}$ (which we will refer to as the constraint matrix), and a vector of kinetic functions, which determine the reaction rates, $r(\xi,\nu) = (r_1(\xi,\nu), r_2(\xi,\nu), ..., r_{N_{reac}}(\xi,\nu))^T \in \Re^{N_r}$. The reaction rate $r(\xi,\nu)$ is sometimes written as; $r(\xi,\nu) = C_x f(\xi,\nu)$, where C_x is the biomass concentration, to reflect the prior knowledge that the system breaks down into biotic and non-biotic phases.

The second term represents transport of material across the reactor boundary. $D \in \Re$ is the overall dilution rate due to both outflow and changes in volume and is given by $\frac{F_{in}}{V}$; the ratio of incoming volumetric flow to reactor volume. $g(\xi) \in \Re^{N_c}$ is a vector of the gaseous outflow rates per unit volume and is a complex function of equilibrium and mass transfer effects. $F \in \Re^{N_c}$ is a vector representing additions to the reactor on a per unit volume basis.

For brevity we sometimes write the net effect of feed and gaseous outflow compactly as a single transport term $u = (u_1, u_2 \ldots u_{N_c}) \in \Re^{N_c}$ where each u represents the balance between inflows/outflows for a particular species.

The above representation provides a natural way of combining *a priori* and data driven components into a parallel hybrid model since the reaction kinetics can be modelled by either data driven or mechanistic models.

Data driven modelling

In the black box approach the model structure is determined so as to fit experimental data. No *a priori* restrictions are placed on the form of the model or range of the parameters. Examples of black box representations include neural networks and polynomials. Typically the model is constructed by choosing parameter values, which minimise the difference between the model prediction and some data. The data used for this purpose in known as *training data*.

Hybrid modelling

The hybrid approach combines black box and white box components within an overall model. Two different types of hybrid model can be distinguished. In the serial approach equation (1.3) the model prediction is a weighed sum of the predictions of black box and white box models.

$$\frac{d\xi}{dt} = \left(1 - \underbrace{w}_{\substack{weighing \\ factor}}\right)\left(\underbrace{Kr(\xi,v) - D\xi - g(\xi) + F\xi_{in}}_{white\ box\ model\ prediction}\right) + w\left(\underbrace{f(\xi,v,u)}_{\substack{bb \\ black\ box\ model\ prediction}}\right) \qquad (1.3)$$

In the parallel hybrid modelling approach *a priori* and data driven components are combined into an overall model as shown in Figure 3. The transport terms are assumed to be known. The constraint matrix $K \in \Re^{N_c \times N_r}$ is normally obtained *a priori* from elemental balances or knowledge of the metabolic network while the reaction kinetics can be determined by either kinetic functions from literature or by fitting black box components to the available data.

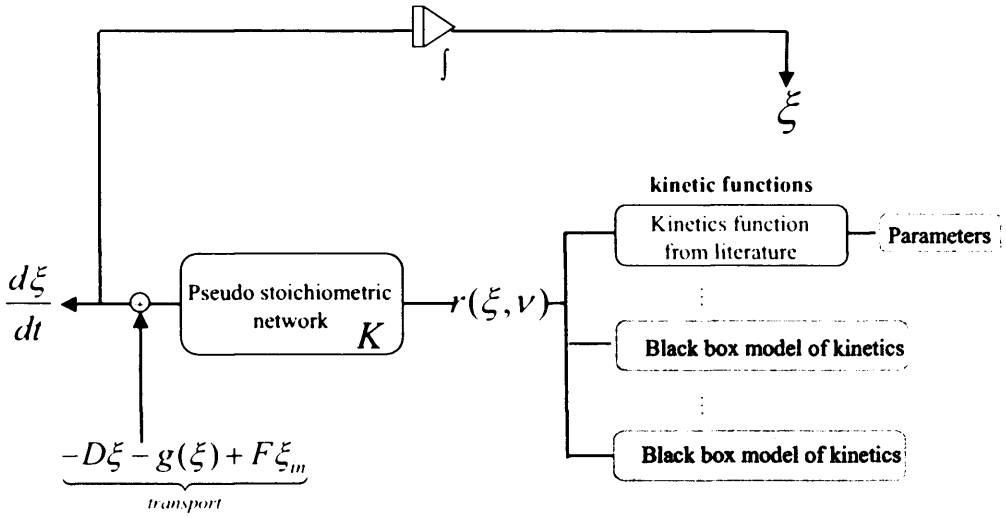


Figure 3. Diagram of hybrid model representation.

1.3 Scope of thesis:

1.3.1 Research Aims.

Model based process supervision, supervision, optimisation and control has only been applied to a small number of industrial bioprocesses. The adoption of model based strategies has been hampered by a lack of accurate models and the long development times associated with model building.

The general aim of this thesis is to speed up model development though improved methodologies for data driven modelling within the *serial hybrid-modelling framework*. The focus is on developing an improved general methodology rather than on applying existing methods to a specific process. This work splits naturally into two parts:

- In the first part a fast methodology is developed for quickly building black box components of hybrid models. Data driven methods for inferring the stoichiometric constraints are considered. Then support vector machines are proposed for kinetic modelling.
- In the second part Bayesian framework is proposed for jointly inferring the parameters of both mechanistic and black box components from incomplete data sets. The Bayesian approach provides a theoretical basis for coping with

uncertainty, dealing with noisy data, using data with missing measurements and incorporating information in the form of uncertain beliefs into model.

1.4 Structure of thesis.

In chapter two the contribution of models to bioprocess operation and development is reviewed. Three types of models are distinguished:

- **Response surface models**, which are simple non-recursive statistical models. These relate a dependant variable of interest such as yield to independent variables such as initial conditions. Response surface models are a useful tool for optimising environmental conditions and the composition of initial growth media.

- **Inferential sensors,** which are models, which estimate the value of variables that cannot be easily measured from variables that are measured online. By providing estimates of key variables inferential sensors allow feedback control and are hence are useful for bioprocess operation.

- **Dynamic models**, which are capable of predicting the behaviour of the system as a function of the current state and control actions. These models can be used for both optimisation and inferential estimation.

Chapter three, considers existing bioprocess modelling methodologies. Approaches to white box modelling are outlined. Black box modelling representations: feed forward neural networks; radial basis neural networks and genetic programming are reviewed. The problem of overfitting and poor generalisation performance is then highlighted. Finally the parallel and serial approaches to hybrid modelling are discussed.

As a result of these initial chapters it is proposed that:

- Development of mechanistic models of bioprocesses is time consuming and in many cases it may not be possible to determine the stoichiometric constraints and reaction kinetics *a priori*. The ability to quickly develop dynamic models would therefore significantly enhance bioprocess engineering.

- The serial hybrid-modelling framework is more appropriate for rapid model development than the parallel hybrid-modelling framework since the latter requires a complete mechanistic model to be developed.

- Development of hybrid or black box models is hampered by the need to select the architecture of data driven components such as neural networks so as to minimise overfitting.

With these objectives in mind a methodology for building the data driven components of serial hybrid models is developed in chapters three to seven.

Chapter four considers the problem of determining the system constraints. The use of elemental balances and metabolic network analysis is reviewed. Then two methods for inferring the constraints from data are detailed:

- The first method uses regression to determine unknown pseudo-stoichiometric coefficients of a reaction network.

- The second method involves the use of principal component analysis (PCA) to infer constraints of a hybrid model of a bioprocess without any prior knowledge of the stoichiometry or reaction network.

In chapter five a methodology is proposed for inferring the reaction kinetics using support vector machines(SVM's). The SVM method is then demonstrated on a simulated hybridoma culture. Support vector machines were selected since SVM's do not suffer from the problem of local optimum and the architecture of SVM's is determined automatically from data so as to minimise overfitting. Difficult decisions about the architecture of neural networks are therefore avoided and the process of data driven modelling is considerably simplified.

In chapter six the SVM approach is used to model three experimental systems: a Murine hybridoma cell culture; a *Saccharopolyspora erythraea NRRL2338* shake flask cultivation and a 42L *Streptomyces clavulingerus* batch cultivation.

In chapter seven the relative performance of models built using three techniques are compared. These techniques were: support vector machines; multilayer perceptrons and genetic programming. The comparison involves testing completely automatic methodologies on a large number of randomly created simulated systems.

Development of an advanced methodology: chapters eight and nine.

A key weakness with the above methods (and many similar approaches) is that the data driven components are trained to predict reaction rates as a function of the system state at a specific time. Taking derivatives in order to obtain these reaction rates magnifies measurement noise and requires the state vector to be interpolated.

In chapter eight the hybrid-modelling problem is cast in terms of Bayesian inference. Firstly a Bayesian approach to white box modelling is detailed. It is then shown how neural networks can be used within the Bayesian approach. Finally it is demonstrated that the Bayesian approach can cope with missing data. This leads to a modelling framework with the following advantages:

- No derivatives need to be taken since data is directly used.
- Data can be used even if measurements of entire series are completely missing.
- Mechanistic and black box components can be mixed in a flexible manner and knowledge about the values of parameters introduced through 'Bayesian priors'
- The framework is developed further in chapter nine where it is shown that the Bayesian approach to hybrid modelling can be used to discriminate between different mechanistic sub models on the basis of data. Unfortunately the Bayesian approach is very computationally intensive and therefore represents a promising direction for future research rather than a complete and immediately applicable methodology.

2 Bioprocess operation and development.

The objective of bioprocess engineering, in an industrial context, is to maximise profitability. Profitability is a complex function incorporating not only the obvious factors such as: growth rate; product formation rate; yield; batch duration, which define the costs of running the process but also, labour costs, time to market, and variability.

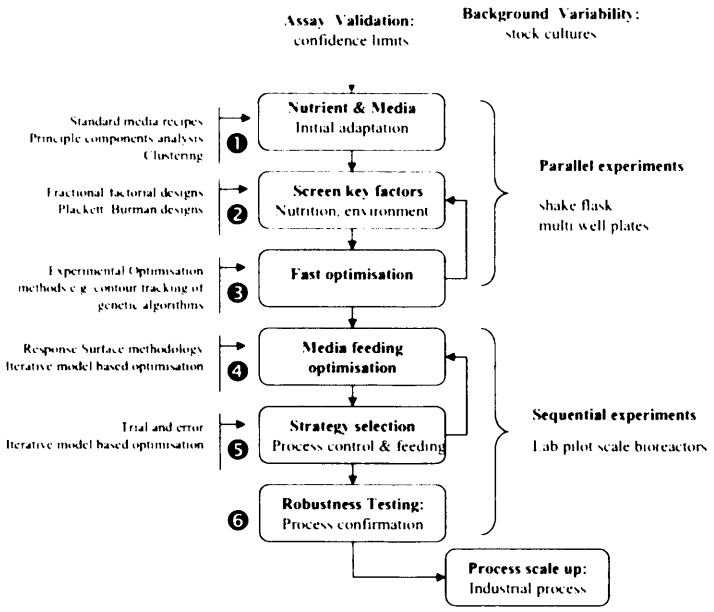
The engineering decisions regarding a bioprocess can be broadly classified into two distinct stages: a *process development stage* where decisions on the process design and operating strategies need to be made about a largely unknown process within a limited time frame; and an *industrial operation stage*, where a defined the process is operated and controlled so as to maintain quality and maximise profitability. In this section the contribution that modelling can make at each stage is considered.

2.1 Bioprocess development.

2.1.1 Overview of the development process.

During the development stage the aim is to understand the behaviour of the organism and optimise process parameters such as media composition, operating conditions and feeding strategies. Typically little is known about the process and, because the period of exclusivity of a new drug is very short, decisions need to be made within a very tight time frame.

The general approach to bioprocess development is one of gradual scale up and fixing of process conditions. As the flow diagram in Figure 4 shows development progresses from small and inexpensive experiments in 96 well plates (or shake flasks) to large scale fermentations. Small scale experiments are aimed at finding a suitable liquid media in which the cell line will grow. Bench top fermentations are used to determine key nutritional, environmental factors and control strategies.



❶ Assessing basic requirements and translating cell line to liquid media (adaptation to defined/serum free media would also be at this stage).

❷ Determining which key nutritional and environmental factors affect growth rate and product yield using statistical designs

❸ Crude experimental optimisation in shake flask/multi well plates of key factors using gradient descent/ 'trial and error' methods

❹&❺ Optimising the lab scale bioreactor process including feeding and control strategies using either a combination of response surface models and "trial and error" or model based optimisation

Figure 4. Bioprocess development pipeline. (Adapted from a presentation by Cambridge Bioprocess Management Ltd.)

(Stage 1) It is normally desirable for cells to be grown in liquid media as a suspended culture since this allows for improved mass transfer and material handling. For many mammalian cells an additional challenge may be the need to adapt the cell line to serum free media in order to comply with regulations concerning the use of animal derived serums in therapeutics. It is therefore necessary to determine the basic conditions under which the cell line will grow as a suspended and possibly serum free culture. Standard media recipes are often used, however additional components may need to be added. These required components can be identified by systematically screening different media to identify recipes that ensure satisfactory rates of growth and product formation.

(Stages 2 and 3) The next objective is to determine and optimise the key factors, which affect the expected profitability of the process. Systematic optimisation methods can be employed to directly optimise the process by performing sequential

experiments. Weuster-Botz(2000) used a genetic algorithm[5] (Goldberg(1989)) to optimise medium components for a formate dehydrogenase production process. Cockshott and Hartman(2001) used particle swarm optimisation[6] to improve the medium composition of a Echinocandin B production process. More typically optimisation is based on the informal expert knowledge of experienced process development people rather than systematic methods.

One problem with directly optimising a process is that experiments must be performed in sequence. An alternative option (known as the response surface methodology) is to build a model of the process, from the results of experiments performed in parallel, and then select operating conditions corresponding to the model optimum.

Response surface models of the general form shown in (2.1) predict the value of a dependant variable of interest $y \in \Re$ (such as yield or growth rate) as a function of n independent variables $x \in \Re^n$ (such as temperature or initial glucose concentration).

$$\begin{aligned} y &= f(x) \\ y &\in \Re, x \in \Re^n \end{aligned} \tag{2.1}$$

Typically regression is used to fit a simple linear or polynomial function to experimental data. For example if the objective is to maximise growth rate parallel experiments would be performed to measure the growth rate achieved at different values of independent variables. A function would then be fitted to relate grow rate to the values of these independent variables.

Due to the large number of independent variables which may be involved statistical procedures have been proposed to design the experiments required to build response surface models (Montgomery(1991)), and after a slow start are increasingly used. For example Kalil et al(2000) used a Plackett–Burman design, (Plackett and Burman(1946)) for initial screening and then a Factorial design in order to generate response surfaces and hence optimise the significant variables of a multi stage alcoholic fermentation process. Abdel-Fatta et al(2004) applied a similar method to enhance the production of uricase by *Pseudomonas aeruginosa*. Sen and

[5] Genetic algorithms are a search method inspired by evolution. (See chapter 3.)
[6] Particle swarm' methods (Eberhart et al(2001)) work by each particle in the population remembering its own best previous location and the best previous locations of its nearest neighbours[6] and then accelerating(at random) towards these locations.

Swaminathan(2004) applied the Response surface methodology to the effects of inoculum age and size on surfactin production by *Bacillus subtilis*. In shake flask fermentations with *Rhodotorula gracilis* Kennedy and Spooner(1996) found that simple neural networks and fuzzy logic models could produce a saving of 63% in the number of experiments required for media optimisation compared to factorial designs. However it is hard to see how this fundamentally differed from the response surface methodology except an ANN was substituted for a polynomial.

(Stages 4 and 5) It is difficult to precisely control the conditions of shake flask and microwell cultivations. High oxygen transfer rates cannot be achieved simply by agitation and so cultures cannot reach high cell densities. To continue to optimise the process it is therefore necessary to perform experiments in CSTR or airlift bioreactors. In these later stages experiments are relatively large scale and due to equipment limitations it is simply not possible to perform large numbers of parallel experiments. The final stages of process development are therefore a bottleneck where the cost of experimentation limits purely experiment-based optimisation.

Model based dynamic optimisation.

The output of the initial stages is usually a set of optimum static operating conditions such as initial media concentrations and the values of constant environmental variables. The purpose of the later stages is to determine time varying operating conditions. This includes control of pH, aeration and agitation as well as feeding and induction strategies. Experimental optimisation of these can be a time consuming process of trial and error.

Model based optimisation aims to choose time varying operating conditions $\xi(0), v(t), u(t) \quad t \in [t_0, t_f]$ (subject to physical and model validity constraints) which the model predicts will maximise profit as defined by some utility function:-

$$U(\xi(t), v(t), u(t)) \; t \in [t_0, t_f] \tag{2.2}$$

where the state vector at any time is predicted by simulating a model of the system in the form of differential equations.

$$\xi^P(t) = \int_0^t f(\xi(\tau), v(\tau), u(\tau)) d\tau$$
$$\text{where } \frac{d\xi}{dt} = f(\xi, v, u) \tag{2.3}$$

The control profiles can be represented in a number of ways, such as piecewise linear functions (Carrasco and Banga(1997)), smooth piece-wise control curves or general functions such as wavelets (Binder et al(2000)).

Various techniques have been used to solve the resulting large optimisation problem, such as, dynamic programming and stochastic search methods. The former is a method based on quantising the optimisation parameters to a discrete grid, (Mekarapiruk and Luus(2000)). Stochastic search methods such as; Genetic algorithms, particle swarms, simulated annealing[7], are becoming increasingly popular with Tremblay et al(1993) applying GAs to feed control of a hybridoma cell culture. Moriyama and Shimizu(2005) applied GAs to determine optimal temperature trajectories of a *Saccharomyces cerevisia* fermentation and Roubos et al(1999) used GAs to determine feed trajectories of a *Streptomyces clavulingerus* fed batch process.

If a correct model of the process is available then considerable time can be saved in process optimisation by employing the above approach. The success of model based optimisation is limited by the accuracy of the available model. However the model can be improved as part of an iterative procedure of modelling and experimentation (Figure 5) using model based optimisation to suggest operating conditions and then using experimental data obtained under these conditions to refine the model.

[7] Simulated annealing utilises an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure and the search for a minimum in a more general system. Initially samples (produced by a Markov process) explore large areas of parameter space but as the temperature cools the sampling settles down a smaller range of states. (Kirkpatrick et al(1983))

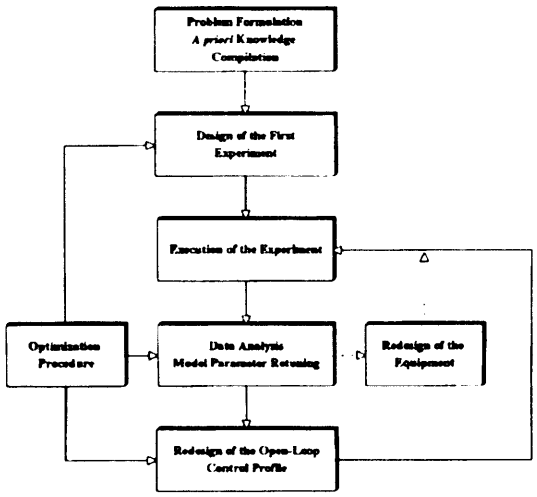


Figure 5. A general structure of model based biochemical process design procedure, taken from (Galvanauskas et al(1998)).

2.1.2 Summary of the role of models in bioprocess development

If sufficiently accurate models exist then they can be used to predict the output variables of interest as a function of controllable variables. This can speed up process optimisation since the time required to perform such *'in silico experiments'* is significantly less than the time required to perform real experiments.

During the initial stages of bioprocess development the objective is to determine the optimal values of static variables. Static models, which directly predict a variable of interest as a function of static operating conditions, are therefore sufficient. Such models can be built by fitting a flexible model to experimental data consisting of input output pairs ($y \in \Re, x \in \Re^n$) obtained from parallel experiments.

During the later stages the objective is to optimise both initial conditions and time varying control profiles. This requires recursive models capable of predicting how a bioprocess will behave over time. Such models can be built from *a priori* knowledge of the underlying mechanisms and/or from measurements of how the state evolves over time. Suitable experimental data typically consists of samples taken regularly from the cell culture. The resulting measurements can be presented as a matrix $\left[\xi_0, \xi_{t_s^0}, \ldots \xi_{t_s^{\langle N_{samples}-1\rangle}}\right]$ consisting of the state vector ξ measured at discrete sample times $t_s = \left[t_0, t_s^0, \ldots t_s^{\langle N_{samples}-1\rangle}\right]$.

2.2 Bioprocess operation.

The objective of bioprocess operation is to ensure that the process remains economically optimal and that it operates within validated bounds. Since industrial bioprocesses are essentially fixed through: validation; legal and financial restrictions, the scope for changing process conditions is limited[8].

2.2.1 Feedback control.

For many bioprocesses feed back control maintains constant pH, temperature and DOT. However in contrast to classical chemical engineering (where feedback control of most key variables is the norm) for bioprocesses the feeding strategy and time of induction[9] is usually predetermined according to a set trajectory such as: constant linear feed; exponential feed or dump feeding[10].

Productivity of many bioprocesses could be increased by the use of feed back control based feeding strategies. For example Cannizzaro et al(2004) increased productivity of a *Saccharomyces cerevisiae* fed-batch fermentation by regulating the external ethanol concentration in the bioreactor. This allowed growth rate to be maximised without ethanol production due to the overflow metabolism inhibiting further growth.

Unfortunately online measurements of some critical process parameters are not normally available due to the lack of cheap and accurate online devices and therefore feedback control cannot be directly implemented. There may however be direct relationships between key process variables such as biomass or product concentrations and readily available online parameters (Offgas CO_2 O_2, feed of acid/base, air flow and stirrer speed).

For example, when there is negligible product formation, the oxygen uptake rate (OUR) is related to the growth rate (μ) and the existing biomass concentration (C_x) through yield on growth Y_{x/O_2} and maintenance Y_{m/O_2}.

$$\frac{\mu C_x}{Y_{x/O_2}} + \frac{r_{mx} C_x}{Y_{m/O_2}} \approx OUR \qquad (2.4)$$

[8] Consequently the scope for productivity improvement is less that the scope during the process development stage. Also the 'information content' of operating batches is relatively low since the operating conditions do not tend to vary much.

[9] Time of induction refers to the point during a cell culture when an addition is made or an environmental variable changes so as to 'switch on' product formation.

[10] Dump feeding refers to the sudden addition of feed at set times during the process operation

The rate of change of dissolved oxygen concentration in the liquid $C^{l}_{O_2}$ is equal to the difference between the oxygen uptake rate and the oxygen transfer rate

$$\frac{dC_{O_2}}{dt} = OTR - OUR \tag{2.5}$$

The oxygen transfer rate OTR is determined by the interfacial concentration gradient between the equilibrium concentration $C^{*}_{O_2}$ and the concentration in the liquid $C^{l}_{O_2}$.

$$OTR \approx k_L a(C^{*}_{O_2} - C^{l}_{O_2}) \tag{2.6}$$

The oxygen transfer coefficient $k_L a$ can be modelled as a function of impeller speed V_s and aeration flow rate P_g. (V is the culture volume and c, x, y are experimentally determined constants).

$$OTR \approx k_L a(C^{*}_{O_2} - C^{l}_{O_2})$$
where (2.7)
$$k_L a \approx c\left(\frac{P_g}{V}\right)^{x}(V_s)^{y},$$

Thus a variable (growth rate) which is not measurable online can be related to online variables. Other relations exist, for example REDOX balances can be used to relate the measured pH and the volume of acid/base added to growth or product formation rates. It therefore should in principle be possible to infer at least some immeasurable variables from online variables.

Steady state calibration models.

The first approach is to directly develop a model which predicts the variable of interest (Biomass in the discussion) as a function of online measurements. These models are in the general form shown in equation (2.8) where $Y \in \Re^{N_{online}}$ is the vector of online measurements and $\xi_t \in \Re$ is the variable of interest.

$$\xi_{t,t-k} = f(Y_{t-k}) \tag{2.8}$$

There are two problems with this approach: The first is that since the state estimate is a direct function of the online measurements the state estimate is corrupted by measurement noise. The second is that there is often no unique relationship between the state and the observations. For example in the above discussion the oxygen uptake rate is the sum of the oxygen demand due to both growth and maintenance (equation

(2.4)). If the only information is the oxygen uptake rate this equation does not have a unique solution for μ and C_s.

An alternative approach is to use black box modelling techniques to infer correlations between key process variables and online measurements. An example of such an inferential sensor is shown in Figure 6 and Figure 7. These show the inferential estimation of the nitrogen source concentration from feed flow rates and off gas analysis in an industrial fermentation. The model was built using feed forward neural network software developed in industrial work related to this thesis.

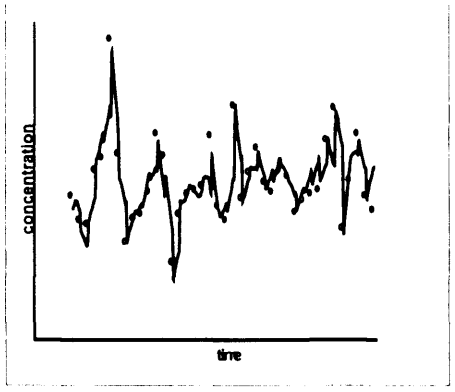


Figure 6. 'Soft sensor' accuracy on testing batch 1. (Details and units are not provided for confidentiality reasons.)

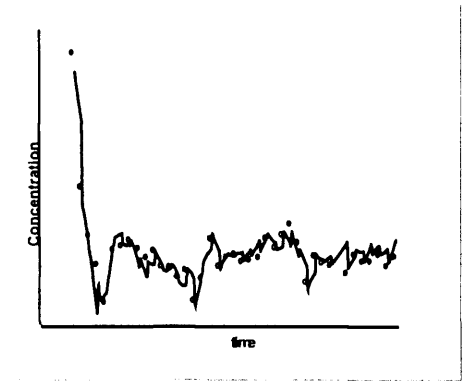


Figure 7. 'Soft sensor' accuracy testing batch 2. (Details and units are not provided for confidentiality reasons.)

Recursive Bayesian estimation (Kalman filters).

An alternative approach is to determine the *most likely* value of internal states from available measurements. Recursive Bayesian estimation provides a systematic method for determining the most likely values given a model of the process.

The theory of observers for linear systems dates back to the early years of control theory (Kalman(1960); Luenberger(1966)) and has been extended to non-linear systems. An overview of observers in a bioprocess context can be found in Bogaerts and Wouwer(2004) and of the extended Kalman filter in Wilson et al(1998).

The goal of filtering is to estimate the system state from noisy and incomplete measurements. Kalman filters use a measurement model $Y_{t-k} = h(\xi_{t-k}, w)$ relating the observed measurements Y to the internal state of the system and a process model

$\xi_{t-k} = f(\xi_{t-k-1}, w)$ which predicts one time step ahead to obtain a maximum likelihood estimate of the system state as new measurements become available (Figure 8).

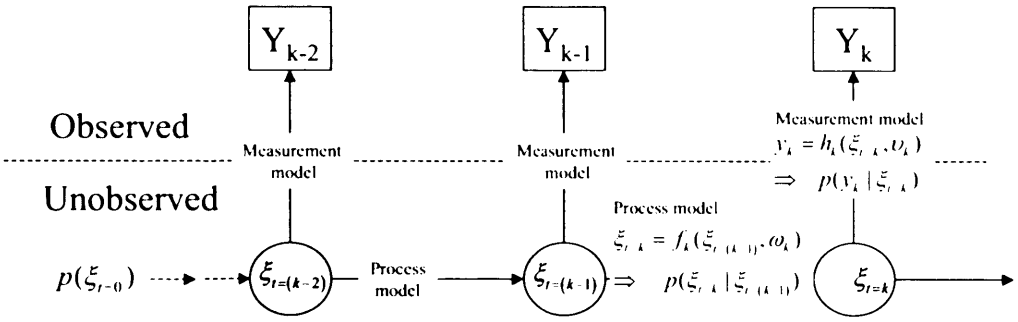


Figure 8. Kalman filter.

The filter is initialised with a belief about the previous state in the form of a probability density function $p(\xi_{t-(k-1)})$. The condition probability density function of the current state $p(\xi_k | p(\xi_{k-1}))$ can be obtained from model of the system $\xi_{t=k} = f\left(\xi_{t=(k-1)}, w\right)$. This prediction is then corrected in the light of the new measurement $Y^*_{t=k}$ using the measurement likelihood function $p(Y_{t-k} | \xi_{t=k})$ obtained from $Y_{t-k} = h(\xi_{t=k}, w)$.

This optimum recursive Bayesian estimate is given by:

$$\underbrace{p\left(\xi_{t=k} | Y_{t=k}\right)}_{posterior} = \frac{\overbrace{p\left(Y^*_{t=k} | \xi_{t=k}\right)}^{likelihood} \overbrace{p\left(\xi_{t=k} | Y_{t=k-1}\right)}^{prior}}{\underbrace{p\left(Y_{t=k} | Y_{t-k-1}\right)}_{evidence}} \tag{2.9}$$

where

$$p\left(\xi_{t=k} | Y_{t=k-1}\right) = \int p\left(\xi_{t-k} | \xi_{t-k-1}\right) p\left(\xi_{t=k} | Y_{t-k-1}\right) d\xi_{t=k-1}$$

$$p\left(Y_{t=k} | Y_{t=k-1}\right) = \int p\left(Y_{t=k} | \xi_{t=k}\right) p\left(\xi_{t-k} | Y_{t-k-1}\right) d\xi_{t=k-1}$$

With the estimated state being the expectation of $p(\xi_{t-k})$;-

$$E\{\xi_{t-k}\} = \int \xi_{t-k} p\left(\xi_{t-k} | Y_{t-k}\right) d\xi_{t-k} \tag{2.10}$$

Calculating these multidimensional integrals is very difficult and so the approach of the Kalman filter is to assume the probability distribution is Gaussian and approximate it by the mean and variance of this Gaussian. This means that no

integrals need to be calculated and only the mean and variance propagated through the model.

For non-linear systems this leads to the extended Kalman filter which linearises the model around the current estimated state by calculating the Jacobian[11] (Figure 9). For example, Zorzetto and Wilson(1996) used a hybrid model in the form specified in the introduction as the process model of a EKF to monitor a *Saccharomyces cerevisiae* process. Unfortunately the EKF suffers from linearisation errors. This issue is addressed by the Unscented Kalman filter (UKF). The idea of the Unscented Kalman filter is that *"it is easier to approximate a probability distribution than it is to approximate an arbitrary non-linear function or transformation"* (Julier et al(2000)). The UKF works by propagating carefully chosen samples known as 'sigma' points though the unmodified non-linear model instead of linearising the model (Figure 10) and can be seen as a computationally efficient alternative to importance sampling (Figure 11).

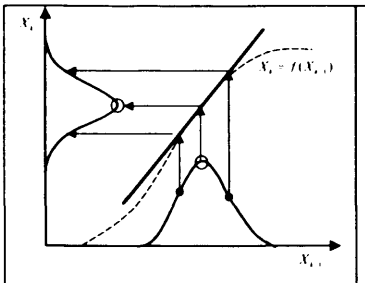


Figure 9. Extended Kalman filter.

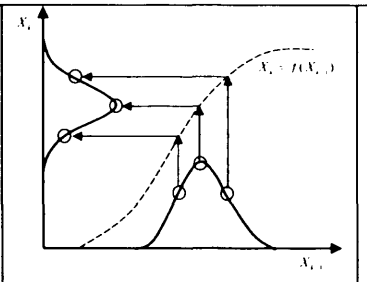


Figure 10. Unscented Kalman filter. Adapted from Orderud(2005)

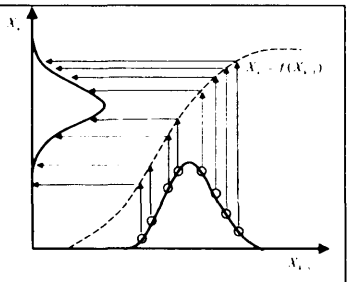


Figure 11. Sequential importance sampling.

With the exception of Romanenko and Castro(2004) the unscented Kalman filter has not been applied in a chemical engineering or bioprocess context, which is somewhat surprising given that the performance of the filter surpasses that of the extended Kalman filter method while actually being simpler to apply. Given a dynamic model these filtering techniques can accurately estimate measured and unmeasured states from noisy measurements and hence enable feedback control.

[11] Matrix of first partial derivatives with respect to the state variables.

2.2.2 Model predictive control.

MPC is the online application of model-based optimisation to recalculate control actions. In the usual MPC framework, the following steps are performed to calculate future control signals (Garcia et al(1989), Gawthrop et al(2003)):

1. The future outputs $\xi(k+i)$ for the variables of interest between determined horizons $H_c < H_p$, called the control and prediction horizons, are predicted using the process model. These predicted outputs for $i = 1 \ldots H_p$ depend on the current state of the system and on the future control signals $u(k+i)\ i = 1 \ldots H_c$.

2. The set of future control signals is calculated by optimising an objective function in order to keep the process as close as possible to a pre determined reference trajectory $ref(k+i)$. This criterion usually takes the form of a quadratic function of the errors between the predicted output signal $\xi(t)$ and the desired reference trajectory $ref(t)$. The control effort $\Delta u(t) = u(t) - u(t-1)$ is included in the objective function in most cases.

$$\arg\min_{u(t)} \left(\sum_{i=0}^{N_p-1} \left| ref(k+i) - \xi(k+i) \right| P + \sum_{i=0}^{N_c-1} \left| \Delta u(k+i-1) \right| Q \right) \qquad (2.11)$$

Where are Q,P weights for multiple variables of interest they are matrices determining the trade off between the various objectives.

3. The control signal $u(t)$ is sent to the process while the next control signals calculated are rejected, since at the next sampling instant $\xi(t+1)$ is known. Step 1 is repeated using this new value and all the sequences are brought up to date (receding horizon strategy).

Preuss et al(2000) and Hodge and Karim(2002) applied MPC to fed batch yeast growth on an industrial scale. However MPC is not commonly used in bioprocesses due to: the difficulty in obtaining accurate models; a lack of online measurements and the difficulty of solving the optimal control problem for non linear models in a timely manner.

2.2.3 Summary of the role of models in bioprocess operation

For most bioprocesses feed back control is currently only implemented at a low level (for temperature, pressure, pO2, pH) while feed forward control is generally not used.

For feed back control to be applied to more advanced objectives key process parameters, such as substrates and biomass concentrations, need to be measured online. The accuracy of most of the on-line measurements is low. Therefore, observers or filters are necessary to estimate the unknown variables.

Two approaches to state estimation were distinguished. The first 'soft sensor' approach directly estimates the variable of interest $\xi_t \in \Re$ as a function of measurable variables $Y \in \Re^{N_{online}}$.

$$\xi_{t,t=k} = f\left(Y_{t=k}\right) \tag{2.12}$$

In the second 'Bayesian filtering' approach models the estimate of the state is chosen so as to be consistent with the observed measurements and the previous estimate of the state. This approach requires both a measurement model which predicts $Y \in \Re^{N_{online}}$ as a function $Y_{t=k} = h(\xi_{t=k})$ of the state $\xi_{t=k}$ and a process model which predicts the state $\xi_{t=k}$ as a function $\xi_{t=k} = f(\xi_{t=k-1}, w)$ of its previous value $\xi_{t=k-1}$.

For model predictive control to be implemented both predictive models and measurement models are necessary.

2.3 Overall conclusion

The potential contribution of models to bioprocess operation and development was considered. Three types of models were distinguished:

- Response surface models capable of modelling a variable of interest as a function of independent parameters.
- Inferential sensors capable of inferring a variable or variables of interest from online measurements.
- Dynamic models capable of modelling the evolution of the system with respect to time as a function of initial conditions and time varying control profiles.

Table 1 The usefulness different types of model in the different stages of bioprocess operation and development.

Stage		Response surface models	Dynamic models	Inferential sensors
Development	Initial media optimisation	X		
	Optimisation of feeding strategies		X	
Operation	Feed back control		X	X
	Model based optimal control.		X	

The potential contribution of each model type is summarised in Table 1. While response surface models are useful during the early stages of bioprocess development they cannot easily be used to optimise dynamic time varying control profiles. Inferential sensors are useful for providing continuous estimates of variables of interest and hence implementing feedback control. The greatest potential contribution is from dynamic models. These models can be used to optimise static operating conditions and time varying profiles in both a process development and a process operation context. Additionally if used as part of an state observer such as a Kalman filter they can be used for state estimation and hence feedback control. Unfortunately dynamic models are by far the most complex category of models and the most difficult to build.

3 Modelling Approaches.

In this chapter the various approaches to modelling bioprocesses are reviewed. There are three main categories of models. Fully mechanistic models are termed *'white box'* although in reality many relations used in these models such as the logistic growth equation are simply widely applied empirical models. Fully empirical models, inferred from data are termed *'black box models'*. Models which include a mixture of data driven and mechanistic elements are termed *'hybrid models'*.

3.1 White box:

3.1.1 An overview of white box models.

The term *'white box model'* refers to first principle or knowledge-based models, derived from tested and accepted theories of underlying physics or chemistry. In practice it is not strictly true that white box models are formed entirely from *a priori* knowledge. Parameters of white box models are chosen to fit observations and many kinetic models such the 'logistic growth' equation are empirical. Generally we shall use the term *'white box'* to refer to models for where the structure (but not necessarily the parameter values) are determined on the basis of physical insight or *a priori* knowledge.

White box models only exist for a few well described micro-organisms producing defined products such as yeast. This poses a problem since many industrial processes involve genetically modified organisms producing unique products or growing on unusual substrates. Building new white box models requires the collection of detailed biological knowledge to determine the underlying mechanisms and is hugely time consuming.

Because of the complexity of the true system it is necessary to make simplifying assumptions. This may mean that certain behaviours are not fully captured since the model is inadequate. Most white box models can be classified on the basis of these assumptions into one of the four categories shown in Figure 12.

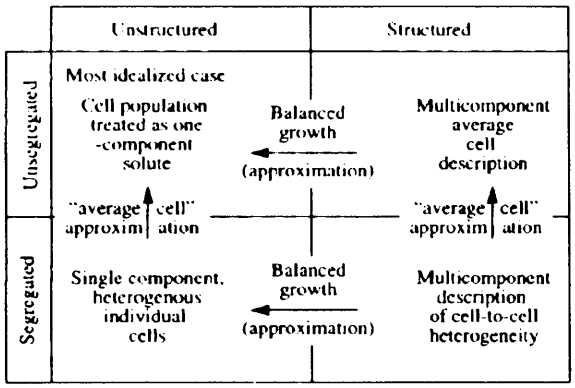


Figure 12. Classifications of models taken from Bailey, J. E.(1998).

Segregated models treat the population of cells as consisting of several classes with different behavioural characteristics rather than as a single homogeneous population. Altýntas, et al(2001) described a model of *Saccharomyces cerevisia* where the population is segregated into 3 classes: Cells containing plasmid and expressing the gene product; Cells containing plasmid but not expressing the gene product and Cells without plasmids. The proportion of cells in each class is explicitly modelled.

Structured models can be compartmental models such as the model of *Escherichia coli.* described in Nielsen et al(1990). In this model the four 'compartments' are: A-ribosomes; mRNA; tRNA; P-plasmid DNA; E-plasmid product; G-genome DNA and structural material. Glucose is the single limiting substrate. An even more detailed class of structured models are metabolic models[12]. These models incorporate detailed information about the network of internal reactions within the cell. (Bellgardt, 1991; Bellgardt and Yuan, 1991; Nielsen and Villadsen, 1992; Schmidt and Isaacs, 1995; Shioya et al., 1995; Guo et al., 1995; Dochain and Perrier,1997). The main benefit of structured models are they are capable of describing a lag phase or transient phase behaviour. However in many cases these effects can be neglected if the response time of the cell to changes in the environment is either negligibly small or very long compared to the duration of the cultivation process. (Tsuchiya et al(2005)) The majority of literature models are unstructured and unsegregated.

3.1.2 Model identification.

If a white box model is available for a similar micro organism to the one of interest it cannot necessarily be assumed that the model parameters or structure will be identical.

[12] Metabolic modelling is described briefly in the next chapter

The strain may have been modified to express a new product, to grow on a specific substrate or to be induced through a mechanism. Even if the micro-organism is exactly the same differences between bioreactors may cause behavioural differences not accountable for within the existing model. It is therefore necessary to identify the 'correct' model from experimental data.

Parameter identification.

If the model structure is known then least squares or maximum likelihood regression can be used determine the parameters of the model which minimise the discrepancy between the model prediction and the measured state at each time point $\left[\xi_0, \xi_{t_1}, \ldots \xi_{t_1(N_{meas}-1)}\right]$.

$$\arg\min_{w}\left(\sum_{i=0}^{N_{samples}-1}\sum_{j=0}^{N_{meas}}\left(\xi^{I}_{t_i^j}-\xi^{P}(t_i^j)^I\right)^2\right)$$
$$\text{where } \xi^P(t)=\int_0^{t_i^j} H(\xi_{t_i^0}, w)dt \tag{3.1}$$

Various optimisation techniques can be used to solve the regression problem. However there is no guarantee that a unique set of best fit parameter values will be identifiable[13]. Even for simple kinetic functions such as the Monod model, the identification is rather difficult, requiring a careful experiments to be carefully planned (Baltes et al(2005)).

This opens up the interesting question: *"how should experiments be designed if the purpose of gathering data is to identify the parameters of a model of known structure?"*

The locally[14] optimal[15] design for parameter identification can be found by means of D–optimal designs that maximise determinant of the expected Fisher information matrix (Federov(1972)). Technically this is given by the second derivative of the negative log-likelihood function as defined in Chapter 8. A conceptual understanding

[13] This issue will be explored in chapter 8, which provides an overview of a Bayesian method for parameter identification.

[14] D optimal designs are only strictly optimal if the parameters are correct. In practice this means requires an initial estimate of the parameters must be available before a design can be constructed..

[15] D-optimality seeks to reduce parameter uncertainty regardless of the relative importance of each parameter, when the model is used for decision-making. See Chapter 8. for an alternative optimal but computationally demanding *'decision theoretic'* formulation.

of D–optimal designs can be obtained by noting that D–optimal designs recommend that experiments are designed so as to obtain data:

(a) Where the resulting measurements are highly sensitive to model parameters,

(b) Where measurement error is small.

(c) Where data has not already been gathered.

Takors et al(1998); Versyck et al(1998) applied the equivalent 'Modified *E*-criterion', the ratio of the largest to the smallest eigenvalue of the Fisher information matrix, in a bioprocess modelling context. The above methodology is highly effective but requires considerable biological and mathematical expertise and has therefore not been widely applied in an industrial context.

Discriminating between model structures

If the model structure is unknown then the optimal design should allow competing models to be discriminated. One obvious criterion for such an experimental design is that data should be gathered where competing models make very different predictions (taking into account parameter uncertainty and measurement error).

Formally an optimal design should maximise the expected relative entropy between the predictive distributions of competing models. With relative entropy defined by the Kullback-Leibler distance[16] (Kullback and Leibler(1951)) between competing models H_1 and H_2.

$$I(p(y|x,H_1),p(y|x,H_2)) = \int p(y|x,H_1)\log\left(\frac{p(y|x,H_1)}{p(y|x,H_2)}\right)dy \qquad (3.2)$$

That is the relative entropy defines the difference between the probability distributions $p(y|x,H_1), p(y|x,H_2)$ for the output variable y predicted by each model at, for example, each sample point given some model inputs x such as initial conditions.

Cooney and McDonald(1995) used a computationally simpler method based on maximising the minimum absolute difference between model predictions, for discriminating between bioreactor model structures.

[16] Model discrimination and parameter estimation are actually equivalent problems. Since parameter estimation can be viewed as discriminating two models $f(w)$ and $f(w-\Delta w)$.

$$\arg\max_{\lambda}\left(\arg\min_{w}\left(\left|H_1(x,w)-H_2(x,w)\right|\right)\right) \tag{3.3}$$

3.1.3 Comment on white box modelling.

There exists both a well developed set of mechanistic models and a methodology for building them. The obvious advantage of the white box approach to modelling is that less experimental data is required during the model building process since certain expected behaviours are already encoded in the form of the model. Unfortunately the application of white box models to industrial fermentations is restricted since.

- The composition of biomass, substrates and products may not be known. Therefore stoichiometric constraints cannot be derived.
- The form of the kinetics of substrate uptake, growth rate and product formation may not be known *a priori* and cannot necessarily be assumed to follow classical kinetic equations such as Monod or diaxic kinetics. Therefore the reaction kinetics cannot be defined.
- Classical models of the kinetics rarely incorporate the impact of environmental conditions.
- Finally development time is a critical issue for industrial processes and white box modelling is very time consuming since detailed research is required to identify the underlying mechanisms. Furthermore complex experimental designs may be required to identify the parameters of mechanistic models.

3.2 Black box modelling:

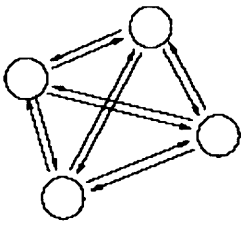
'Black box' models are so called because knowledge of the mechanics of the underlying process is not used when developing the model. We shall use this term to refer to models where the structure of a model is determined on the basis available data rather than from *a priori* knowledge. Because the structure of the model is *a priori* unknown the model representation must be flexible enough to approximate any function

The central idea of black box modelling is to fit a flexible representation to a subset of the available data (known as the *training data*). Some data (known as *validation data)* is reserved to determine the complexity of the model. Three popular black box modelling representations are considered here:

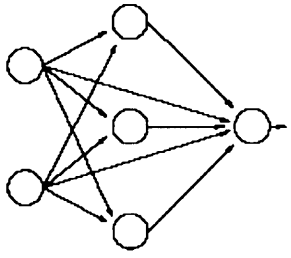
- Feed forward neural networks
- Radial basis neural networks
- Genetic programs

3.2.1 Neural Networks.

Neural networks are systems of simple signal processors designed to mimic a nervous system. They are composed of interconnected signal processors called neurons. There are many types of neural network, however they can be broadly categorised into two types:



Mutually connected networks such as Hopfield networks and Boltzmann machines. These are structured so that every neuron is connected in a bi-directional manner to every other neuron. Hopfield networks (Hopfield(1982)) can be used as associative memories and solving for optimisation problems. A "Boltzmann machine" (Hinton and Sejnowski(1986)) is a stochastic version of Hopfield network which can learn and simulate the probabilities of states of the environment.



Layered networks such as radial basis function networks and multilayer perceptrons. These have a layered structure of neurons with layers ordered from the input layer to the output layer. A neuron in a layer is only connected to the neurons in the next higher layer. Layered networks can be trained on data to perform regression and classification.

Only layered networks will be considered in this thesis since mutually connected networks are not appropriate for function approximation. Multilayer perceptrons (also known as feed forward neural networks (FFNN)) will be discussed. Then radial basis function networks (RBF) will be described.

Feed forward networks (Multilayer perceptrons)

Figure 13 shows the architecture of a Multilayer perceptron. The network consists of an input layer, at least one hidden layer and an output layer. Multilayer perceptrons can approximate any mathematical function as proven by the universal approximation theorem; (Haykin(1999)). This makes neural networks a natural choice for finding unknown complex empirical correlations. One hidden layer, with an arbitrarily large number of units, suffices for the 'universal approximation' property.

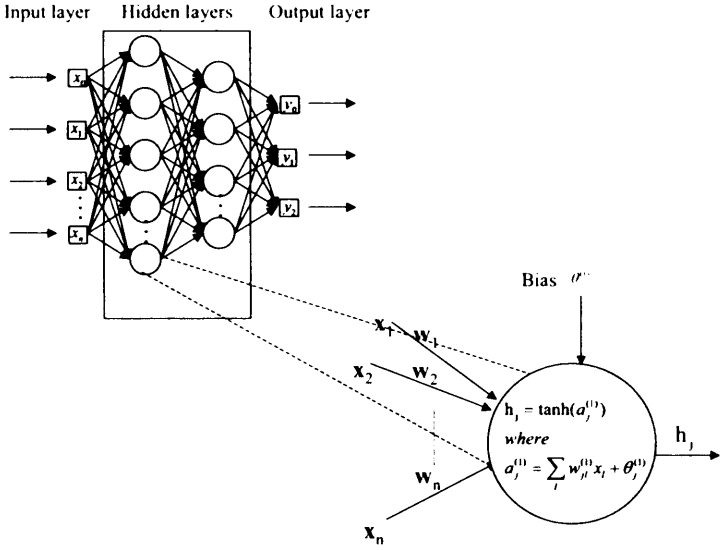


Figure 13. Feed forward neural network architecture.

Each neuron performs a weighted summation of the inputs, which then passes though a one-dimensional non-linear monotonic differentiable *activation function* (ϕ). The most commonly used activation functions are sigmoid or tan functions: $\phi(x) = \frac{1}{1+e^{-x}}$ or $\phi(x) = \tanh(x)$.

The output x_i^l of i^{th} neuron the l^{th} layer can be written as a function of the output of the previous layer x^{l-1} as;

$$x^{l}_{i} = \phi\left(\theta^{(l)}_{i} + \sum_{j}^{N_{inputs}^{l-1}} w_{i,j}^{(l)} x^{l-1}_{j}\right) \tag{3.4}$$

Where $w_{i,j}^{(l)}$ indicates the weight connecting i^{th} neuron the l^{th} layer to the j^{th} neuron the l-1^{th} layer. The network thus defines a non-linear mapping between the input and the output parameterised by the weights of the connections between each layer.

Training feed forward neural networks.

For simplicity we will only consider networks with a single output value where training data consists of N_{pairs} of data: an input vector $x^{(u)} \in \Re^n$ and the corresponding desired output value $y^{(u)} \in \Re$

The network is trained by setting the weights connecting the neurons in each layer so as to minimise some function of the network output. Typically for function approximation this would be the root mean squared error between the neural network output produced when input values are propagated through the network and the desired output for all training examples.

$$\begin{aligned} &\arg\min_{w}\left(R_{emp}(w)\right) \\ &R_{emp}[w] = \frac{1}{N_{pairs}} \sum_{u=0}^{N_{pairs}-1} \left(y^{(u)} - f_{NN}(x^{(u)}, w)\right)^2 \end{aligned} \tag{3.5}$$

Gradient descent can be applied to minimise (3.5) by changing the weights in accordance with the derivative of the error function.

$$\Delta w = \underbrace{-\alpha \frac{\partial R_{emp}[w]}{\partial w}}_{\substack{\text{sensitivity of} \\ \text{error to weight}}} x + \underbrace{\beta \Delta w}_{\substack{\text{momentum} \\ \text{term}}} \tag{3.6}$$

α is called the 'learning constant' and determines the rate of convergence of the algorithm. A momentum term β is sometimes included to decrease the probability of the algorithm becoming trapped in local optimum. Equation (3.6) can be written as the sum of all training examples as

$$\Delta w = -\alpha \sum_{u=0}^{N_{pairs}-1} \left(\frac{\partial f_{NN}(x^{(u)}, w)}{\partial w} \left(y^{(u)} - f_{NN}(x^{(u)}, w)\right) x^{(u)} \right) + \underbrace{\beta \Delta w}_{\substack{\text{momentum} \\ \text{term}}} \tag{3.7}$$

The Δw for weights in hidden layers can be computed efficiently using the back propagation algorithm (Rumelhart et al(1986) Haykin, S.(1999)). However because the optimisation is non linear there is no guarantee that the globally optimum set of weights can be found therefore the training process is usual repeated several times starting from different random initial weights. The back propagation algorithm is as follows:

For each training pair $\left(y^{(u)}, x^{(u)}\right)$

1. **Forward pass.** The input vector $x^{(u)}$ is propagated through the network to predict the output vector $y^{(u)}$ by evaluating equation (3.4) iteratively to obtain the output of the last layer.

$$\underset{NN}{f}(x^u, w) = f(x^{L-1}) = \begin{cases} f(x^{l-1}) = \phi\left(\theta_i^{(l)} + \sum_{j}^{N_{inputs}} w_{i,j}^{(l)} x^{l-1}_j\right) \\ for\ l = 1\ to\ L \end{cases} \quad (3.8)$$

2. **Evaluate the error signal of the output units.** The difference between the desired output $y^{(u)}$ and the network output $f(x^{L-1})$ is calculated and multiplied by the derivative of the output.

$$\delta_i^L = f'(x^{L-1})\left(d_i - \underset{NN}{f}(x^u, w)_i\right) \quad (3.9)$$

3. **Backwards pass.** The error signal at the output units is propagated backwards through the entire network, by evaluating

$$\delta_i^{l-1} = f'(x^{l-1}) \sum_{i=1}^{L} \delta_i^l W_{ij}^l \quad (3.10)$$
$$for\ L\ to\ l = 1$$

4. Weight update. The weights and biases are updated using the error signal obtained from the backwards pass and the network signal obtained from the forward pass.

$$\Delta W_{ij}^l = \eta \delta_i^{l-1} x^{l-1}$$
$$\Delta \theta_i^{(l)} = \eta \delta_i^l \quad (3.11)$$
$$for\ L\ to\ l = 1$$

The algorithm is repeated until convergence or the training process is terminated.

Radial Basis function networks.

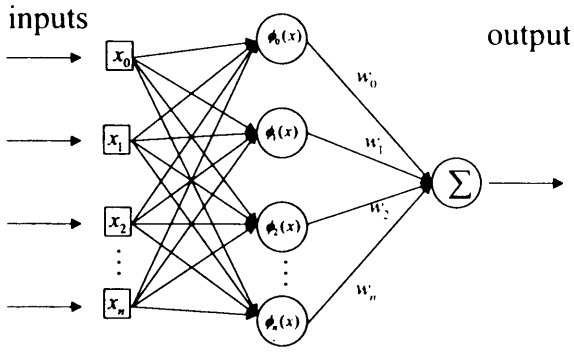


Figure 14. Radial basis function network architecture.

A radial basis function network (Haykin, S.(1999); Poggio and Girosi(1989)) shown in Figure 14 has one hidden layer consisting of radial functions. Radial functions are a special class of functions where the response of the function decreases/increase monotonically with distance from a central point (c_h).

$$\phi_h\left(\left|x-c_h\right|\right) \tag{3.12}$$

The most commonly used basis function is a multidimensional Gaussian

$$\phi_h(x)=\exp\left(-\frac{(x-c_h)^2}{\sigma^2}\right) \tag{3.13}$$

The weights connecting the inputs to this layer are fixed. Only the weights connecting this layer to the output can be changed. The RBF network is non-linear if the basis functions can change size or the centres of the basis functions move. However if the basis functions are fixed then the network approximates a non-linear function as a linear combination of the non-linear features.

$$y(x,w)=\sum_{h=1}^{H} w_h\phi_h\left(\left|x-c_h\right|\right) \tag{3.14}$$

Since the mapping is linear the weight vector $w\in\Re^{N_{basisfunctions}}$ can be easily determined as a straightforward linear regression avoiding the need to perform computationally expensive non-linear optimisation.

$$\begin{aligned}&\arg\min_{w}\left(R_{emp}(w)\right)\\&R_{emp}[w]=\frac{1}{\ell}\sum_{i=0}^{i-1}\left(y^{(i)}-\sum_{h=1}^{H} w_h\phi_h(x)\right)^2\end{aligned} \tag{3.15}$$

This problem is significantly easier to solve than the non-linear optimisation required to determine the parameters of a multilayer perception. An additional advantage is that confidence limits can be easily calculated for the network output.

One key characteristic of Radial networks is that they are inherently incapable of extrapolation. As the input case gets further from the data points represented by the centres of the radial basis functions the activation of the radial units decays. Therefore an input case located far from the training data will generate a zero output from all hidden units. If the bias of the network is set to the sample mean, the RBF will always output the mean if asked to extrapolate. This can be viewed as a positive feature or a limitation depending on your point of view.

Determining the position of the basis functions.

In the initial formulation (Moody and Darken(1989)) RBFs were centred over every point in the training set. This is a very simple technique but results in large networks, which do not perform well on testing data. It is therefore advisable to design a network with a reduced number of basis functions. Unfortunately this leaves the difficult problem of selecting the position of the basis functions if they are not centred over every data point. While several methods exist for determining the position of basis functions there is no universally accepted method and this remains a difficult issue in its own right.

The centres of the radial basis functions can be distributed uniformly within the region of the input space for which there is data or centred over a randomly selected subset of the training data (Broomhead and Lowe(1988)). However, in both of these cases unless the number of radial units is large, the radial units are unlikely to be a good representation of the underlying data Haykin, S.(1999).

Non-linear optimisation techniques can be used to optimise all the RBF network parameters simultaneously including the location of the centres. Kassam and Cha(1995) used stochastic gradient training algorithm whereas Whitehead and Choate(1994) proposed an evolutionary training algorithm. Unfortunately these approaches tend to increase overfitting since the position and width of the basis

functions are determined by the fit to training data as if they were simply additional parameters .

The most commonly used method for determining the centres and widths of the radial basis functions is to use k-means clustering to position the RBFs so that they capture the distribution of the data(Poggio and Girosi(1990)).. The K-means algorithm is a simple algorithm for putting N data points into K clusters as follows.

Initialisation: Set K means to random values

Assignment: each data point n is assigned to the nearest mean.

Update: the means are updated to the mean of the data points assigned to them.

Steps 2 and 3 are repeated until convergence is achieved then basis functions are centred over each cluster.

The K-means algorithm takes account only of the distance between the means and the data points; it has no representation of the size of each cluster. Data points which actually belong to a large cluster can therefore be wrongly assigned to a small cluster. Since the centres of each cluster are determined solely by the data points this can result in sub optimal positioning of the basis functions.

The use of neural networks in bioprocessing.

There been very few reported[17] applications of black box neural network modelling to the production of fully recursive dynamic bioprocess models capable of predicting the state vector at any time point given time varying control profiles $u(t)$ and initial conditions ξ.

$$\xi(t) = \underset{nn}{f}(\xi(t-1), v, u) \; or \; \frac{d\xi}{dt} = \underset{nn}{f}(\xi, v, u) \tag{3.16}$$

Instead work has tended to focus on the use of neural networks to produce black box calibration models for use as inferential sensors or *'one step ahead'* models of the form.

$$\xi_i(t) = \underset{nn}{f}(y^{(t-1)}, t) \tag{3.17}$$

[17] General overviews can be found in Baughman and Liu(1995), Willis et al(1992) and Basheer and Hajmeer(2000).

where $\xi_i(t)$ is a single state of interest such as biomass concentration and $y^{(t-1)}$ is a vector of process states measured online such as DOT. James et al(2002) used a feed forward neural network to the infer the biomass concentration of a *Alcaligenes eutrophus* fed batch fermentation as a function of online variables. While Warnes et al(1998) compared the performance of inferential sensors based round radial basis and feed forward networks on a *Escherichia coli* fermentation process. Galvanauskas, V. et al(1998) compared biomass estimates produced by a feed forward neural network to laser turbidometer signals and found the estimates to be comparable to an accuracy of ± 10%.The 'over-fitting' problem

While the achieved fits can be impressive some of these models include time as an input variable. King and Budenbender(1997) note that many batch fermentations have very similar time varying profiles. It is therefore easy to approximate batch profiles with a simple polynomial function of time. This suggests that the predictive capacity of such model is questionable. It could be the case that the model simply recalls the typical profile of an average batch.

The over-fitting problem.

Both feed forward and radial basis neural networks are powerful black box modelling techniques and are capable of approximating arbitrary functions. However because of this powerful flexibility they suffer from a feature known as *'over-fitting'*. This phenomenon is illustrated in Figure 15 where it can be seen that as the complexity of the model increases, the model is able to fit the training data more closely. However, too complex a model will not be able to generalise and make accurate predictions on new data. Controlling the complexity of the model so as to produce models capable of generalisation is a non-trivial problem.

For multilayer perceptrons the complexity of the function is determined by three factors: the number of hidden layers, the number of neurons in each layer and the characteristic magnitude of the weights Neal(1996a).

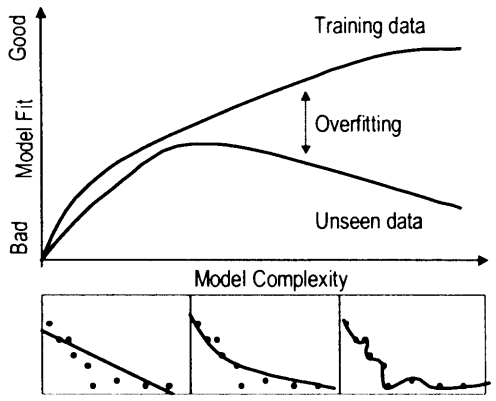


Figure 15. An illustration of the relationship between goodness of fit and generalisability as a function of model complexity. Adapted from Pitt and Myung(2002).

Over simple networks may be unable to capture the underlying complexity of the system. Excessively complex networks with large numbers of neurons and hidden layers tend to 'over-fit' training data. The number of hidden layers and number of neurons should therefore be determined so as to minimise the error on validation data. This can be a time consuming process requiring multiple networks with different architectures to be trained and their performance on validation data compared. Alternatively the size of the network can be reduced by *'pruning'* connections which have no significant effect on the networks performance Karnin(1990).

The magnitude of the weights tends to increase during training and so two techniques are employed to reduce the selection of large weight values and hence over-fitting. The first method is to stop the weight optimisation process before over fitting occurs, by monitoring the error on validation data. The second is to add a regularisation term to the objective function, which penalises large weights

$$R_{reg}[w] = R_{emp}[w] + \frac{1}{2}\sum_{j}\sum_{i} w_{ij} \tag{3.18}$$

For radial basis function networks the complexity of the model is determined by the number and size of basis functions as well as the characteristic magnitude of the weights. Determining the position of the basis functions is therefore a important but difficult task.

An additional cause of over fitting is the so-called *'curse of dimensionality'*. This refers to the fact that as the dimension of the input space increases the number of data

points required to uniquely specify a function increases exponentially. This is a particular problem for RBF networks. RBF networks cannot ignore irrelevant inputs and so large numbers of basis functions are required to cover the entire high dimensional space of the input data. Multilayer perceptrons by contrast tend to concentrate on a lower-dimensional section of the high-dimensional space and can ignore irrelevant inputs by setting the weights from those inputs to zero.

3.2.2 Genetic programming.

Mathematical functions can be represented in tree form (Figure 16). A tree is a data structure widely used in programming and in linguistics for representing context free grammars. Each node in the tree is either a basic function or a terminal. If a node is a function then its sub-nodes are the function's arguments. If a node is a terminal it has no arguments but rather contains a pointer to a constant or a model variable. The representation is computationally efficient since the function can be directly evaluated by calling the root node. In contrast to neural networks GP trees can be read by humans in the form of standard equations.

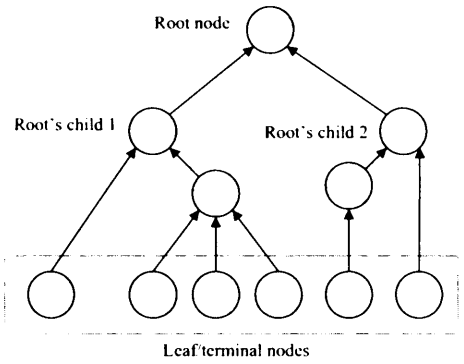


Figure 16. Tree structure.

Provided the function set contains a linear function and a Tauber-Wiener function[18], then a tree structure can approximate any mathematical function with arbitrary precision (Xin(1999)).

Searching over the space of trees can be accomplished by an evolutionary computation technique known as 'Genetic programming' (GP). Genetic programming

[18] For a continuous function to be a Tauber-Wiener function a necessary and sufficient condition is for it not to be a polynomial. Examples of Tauber-Wiener functions include sigmoid, exponential and trigonomic functions.

was popularised by Koza(1992) but despite his patent on the idea it is really a simple extension of 'Genetic Algorithms' (Goldberg, D. E.(1989)). There is prior art going back to Cramer(1985); Dickmanns et al(1987).

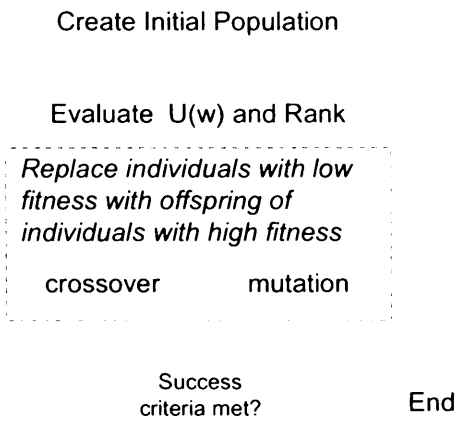


Figure 17. The genetic programming algorithm.

The algorithm (Figure 17) takes its inspiration from evolution, (or rather selective breeding), and so the value of the objective function is referred to as 'fitness' and attempted solutions are referred to as 'individuals'.

An initial population of individuals, representing possible solutions, is generated. The fitness function is then evaluated for each individual. A number of individuals with low fitness are then removed from the population and replaced by new individuals derived from the surviving population.

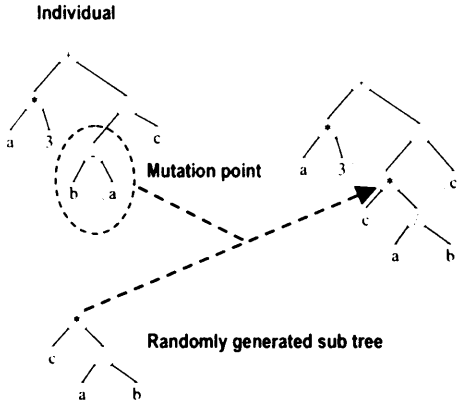


Figure 18. Sub tree mutation.

1. A node is selected at random from the parent.
2. The node and its sub tree are then replaced by a randomly generated sub tree.

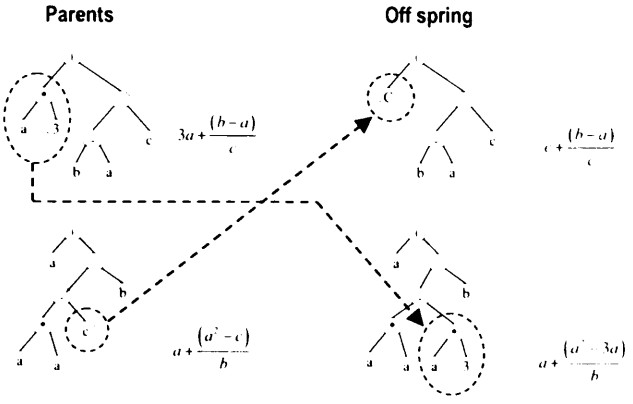


Figure 19. Sub tree crossover.

1. A node is selected at random from both parents,
2. Offspring 1 is given a copy of Parent1's tree but with the selected sub tree exchanged for the selected sub tree on Parent 2.
3. Offspring 2 is given a copy of Parent2's tree but with the selected sub tree exchanged for the selected sub tree on Parent 1,

New individuals are generated by either mutation (Figure 18) or crossover (Figure 19). Mutation consists of changing a sub tree an individual to a randomly generated sub-tree. Crossover consists of recombining sub trees of multiple individuals.

The concept of crossover in theory allows the exchange of useful information between individuals. However this theory has been challenged in recent years and it is not clear that crossover is anymore effective than mutation (Luke and Spector(1997); Muehlenbein(1991)). Genetic programming is therefore best viewed simply as a stochastic search method capable of finding a global optimum given sufficient time.

Genetic programming can be applied to find models from data in a similar manner to neural networks. For regression the fitness function is a normalised version of the error function between the model prediction $\underset{ind}{f}(x^{(u)})$ of the individual being evaluated and the training data.

$$fitness = \frac{1}{1 + R_{emp}}$$
$$where\ R_{emp} = \frac{1}{N_{pairs}} \sum_{u=0}^{N_{pairs}} \left(y^{(u)} - \underset{ind}{f}(x^{(u)}) \right)^2 \qquad (3.19)$$

GP trees tend to become increasingly large as training progresses. This is caused by two factors: Firstly longer trees are more complex and will tend to fit the training data closely and therefore have higher fitness. Secondly the mutation and crossover operations tend (on average) to increase tree size. This not only results in over-fitting problems similar to those seen in neural networks but also slows down the evolutionary search, since large individuals take longer to evaluate than small individuals. In order to prevent such bloat the fitness function can be adjusted to penalise large individuals.

$$fitness = \frac{1}{1 + CR_{cmp} + N_{nodes}} \tag{3.20}$$

N_{nodes} is the number of nodes in the individual and C is a user defined 'hyperparameter' determining the trade off between complexity and fit. The error on validation data can also be monitored and training stopped when over fitting occurs.

Comment on the use of genetic programming in bioprocess modelling.

Genetic programming has recently become a popular method for inferring *'input-output'* models of chemical process models of the form.

$$y = \underset{GP}{f}(x)\ y \in \Re, x \in \Re^n \tag{3.21}$$

For example McKay et al(1997) and Hinchliffe and Willis(2003) used a GP for inferential estimation in a vacuum distillation column and for modelling of a twin screw cooking extruder. Grosman and Lewin(2004) applied GP to determine a kinetic expression for the hydro demethylation of toluene.

In a bio-processing context genetic programming was successfully used for inferential estimation of process states from online measurements by Marenbach(1998)). In this application GP was used to build models in the form of control block diagrams rather than directly as a conventional equations. While they did not apply GP to industrial bioprocesses Cao et al(1999) and Ando et al(2002) showed that GP could be used to infer models of system dynamics in the form of systems of ordinary differential equations.

$$\frac{d\xi}{dt} = \underset{GP}{f}(\xi, v) \tag{3.22}$$

3.3 Hybrid models:

The terms *'Grey box'* or *'hybrid model'*, refer to models which use a mixture of white box and black box methodologies. There are essentially two approaches: *'parallel'*; and *'serial'*.

3.3.1 Parallel hybrid modelling.

In the parallel formulation (Figure 20) a mechanistic model of the whole system is used to describe the main dynamics of the system. A data driven part is then used to compensate for the difference between the white box model prediction and the measured data (Thompson and Kramer(1994)).

$$\frac{d\xi}{dt} = \left(Kr(\xi, v) - D\xi - g(\xi) + F\xi_{in}\right) + \underset{bb}{f}(\xi, v, u) \tag{3.23}$$

If a RBF network is used for the black box component the network will increase the accuracy of the model within the bounds of the training data by modelling the residual. Outside of the range of the training data the network output will decay to zero and the hybrid model will have the same performance as the *a priori* model.

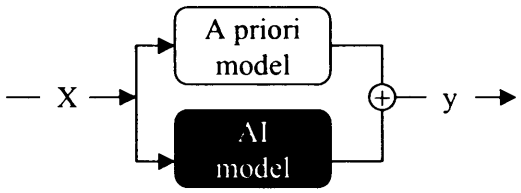


Figure 20. *'Parallel'* hybrid model

If both a mechanistic model and the data required to build a black box model exist the parallel approach can be very successful. For example Lee et al(2002) compared mechanistic, black box, serial hybrid and parallel hybrid approaches to modelling coke plant wastewater treatment plant. They found that the parallel approach gave the closest fit to data of these approaches while giving reasonable responses to process upset. The disadvantage of this approach is that a both a full black box model and a full mechanistic model must be developed. The serial hybrid modelling approach is therefore appropriate where accuracy is more important than speed of development or where a full mechanistic model is already available but insufficiently accurate.

One interesting extension of this parallel grey box modelling approach is the *'operating regimes'* formulation shown in Figure 21 (Foss et al(1995); Johansen and

Foss(1998)) the whole input space is covered by a patchwork of locally valid mechanistic or black box models sub models. A *'meta model'* is then used to decide on the relative validity of sub models and thus output a weighed sum of the sub model predictions.

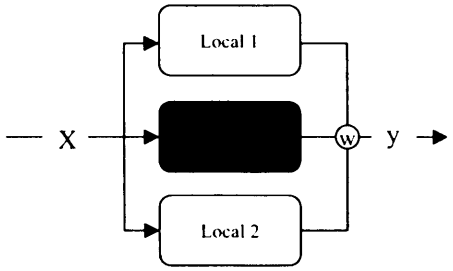


Figure 21. The 'operating regimes approach to hybrid modelling.

The general scheme is a weighed sum of the predictions of different models as shown in equation below.

$$\frac{d\xi}{dt} = \sum_{i=0}^{N_{models}-1} w_i\left(f_i(\xi,v,u)\right) \qquad \text{where} \sum_{i=0}^{N_{models}-1} w_i = 1 \tag{3.24}$$

Where $f_i(\xi,v,u)$ is the output of the i^{th} model which can be either black box or mechanistic. The weight w_i attributed to each model can either be based on the relative confidence levels of each sub model (for example by Oliveira(1998) who used the approach for state estimation) or encoded *a priori* as a set of fuzzy logic[19] rules for example by Schubert et al(1994).

3.3.2 Serial hybrid modelling.

In the serial approach shown in Figure 22 data driven components are used to predict variables which are meaningful within the context of a white box framework. Typically *a priori* knowledge can be used to construct mass and elemental balances and which impose constraints on the data driven kinetic functions.

[19] In fuzzy logic (Zadeh(1965) values are not true/false but rather the degree to which an entity belongs to a class is represented by a real valued parameter $\mu \in \Re \quad 0 \le \mu \le 1$ known as it's membership.

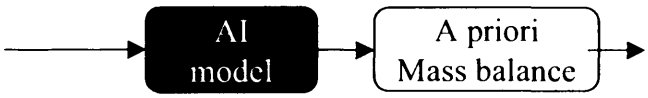


Figure 22. The *'serial approach'* to hybrid modelling.

The hybrid model thus has the following form:

$$\frac{d\xi}{dt} = \underbrace{Kr}_{kinetics} \underbrace{- D\xi - g(\xi) + F\xi_{in}}_{transport} \tag{3.25}$$

The physical transport effects are modelled *a priori* by the last term. The matrix $K \in \Re^{N_c \times N_r}$, which imposes stoichiometric constraints, is usually determined *a priori* although it can be determined from data as will be shown in the next chapter. The reaction rates r are determined by either white box models of the kinetics or black box models of the kinetics. We distinguish two different forms of serial hybrid model: The first form is where a black box component is used to estimate a reaction rate from online data Y such as OUR.

$$r_i = f(Y) \tag{3.26}$$

Psichogios and Ungar (1991) used this serial approach to model a fed batch bioreactor. The network used off gas data to estimate the specific growth rate, which was then input into a white box model of the component mass balances. Hybrid models of this first type are suited to process control applications where the objective is to estimate internal states or to predict one time step ahead. They are not suited to a development context since the hybrid model requires online data to work and therefore cannot be used recursively to predict the behaviour of new fermentations. Since such models cannot be used to perform *in silico experiments* they cannot be used to find the optimal initial conditions or time varying control profiles.

The second form of serial hybrid model is where black box components are used to model the reaction kinetics as a function of the current state vector and controlled environmental variables.

$$r_i = \underset{bb}{f}(\xi, v) \tag{3.27}$$

The form of the model is exactly the same as a white box model but where a data driven component has been used to determine the kinetics of one or more of the reactions. Chen, L. et al(2000) used radial basis functions to determine the reaction kinetics of a antibiotic production process in this way. While Oliveira(2004) used a

combination of feed forward neural network and mechanistic models of the kinetics. Roubos et al(2001) compared mechanistic models, feed forward neural networks and fuzzy logic for modelling the kinetics of a *Streptomyces clavulingerus* batch cultivation. Feyo de Azevedo et al(1997) compared a serial hybrid approach using back propagation neural networks to an unstructured sliding window approach and Galvanauskas et al(2004) applied feed forward neural networks in conjunction with Monod type equations to model the kinetics of an animal cell culture production process. Since the kinetics are not determined by online measurements these models are fully recursive. This second approach is therefore useful for rapidly building dynamic models from data and as such it is suited to a bioprocess development where there is a need to optimise the time varying control profiles but insufficient time available to build a fully mechanistic model. With the addition of a measurement model this type of hybrid model can be used for a state estimation following the Bayesian filtering approach.

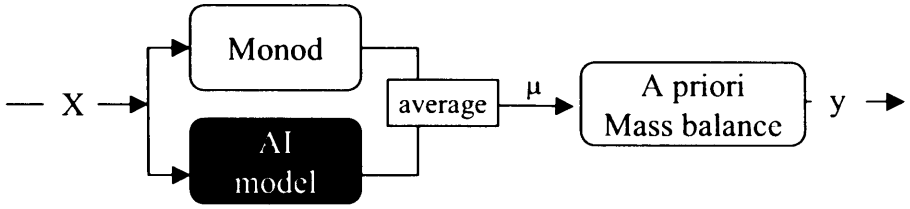


Figure 23. *'Parallel'* and *'serial'* elements within the same model.

The serial and parallel approaches are not mutually exclusive in that sub models within the serial approach can themselves be hybrid models. For example as Figure 23 shows a sub model predicting growth rate could be a *'parallel hybrid'* model composed of weighed average of a Monod model and neural network predictions.

3.4 Summary

Mechanistic models of many bioprocesses can be found in literature. Such models are generally considered to valid over a wide range of operating conditions. Development time is a critical issue for industrial processes and therefore the detailed research required to identify the underlying mechanisms presents a barrier to the development of white box models of industrial processes involving new or modified micro-organisms. Moreover since mechanistic models are necessarily simplifications of reality their accuracy may be limited. In particular they may only capture the response

of the system to the key variables that are included in literature models of enzyme kinetics and may ignore the impact of other conditions.

The alternative data driven approach to model development involves training 'universal function approximators' such as neural networks on experimental data. This method can be effective at fitting training data and can result in more accurate fits that those achieved by mechanistic models. However predictions of data driven models under new conditions can be highly inaccurate. The main problem is that excessively complex black box models can over fit the training data at the expense of their ability to generalise.

In the parallel hybrid modelling approach white box and black box techniques are combined to produce a model where the prediction is the sum of a mechanistic model and a black box model (most effectively an radial basis function network). This technique improves the accuracy of mechanistic models within the range of the training data while maintaining the extrapolation abilities of the mechanistic model. However the parallel hybrid modelling approach cannot be used if a mechanistic model of the whole process does not exist.

The serial approach to hybrid modelling is a general framework where models are built in the form shown in equation (3.28) .

$$\frac{d\xi}{dt} = \underbrace{Kr(\xi,v)}_{kinetics} - \underbrace{D\xi - g(\xi) + F\xi_{in}}_{transport} r_i = \tag{3.28}$$

In a mechanistic model the kinetics $r(\xi,v)$ are all determined from knowledge of the underlying mechanisms. In serial hybrid models some or all of these expressions are black box models inferred from data. The serial approach is therefore a flexible framework for building models ranging from where all the kinetics are inferred from data (completely black box) models to completely white box models where all the kinetics and constraints determined from mechanistic insight.

- If no mechanistic model of the kinetics a particular reaction (or indeed all the reactions) is available this approach allows models to be built quickly from experimental data.

- If a mechanistic model of the kinetics a particular reaction is available either the mechanistic or black box component can be used or both combined in parallel.

3.4.1 Direction of research.

The serial hybrid formulation is the most flexible framework for bioprocess modelling. However it requires prior knowledge of the stoichiometric constraints. In many cases this knowledge will not be available therefore there is a need to develop methods for inferring the constraint matrix from data. This would allow entirely data driven models to be quickly developed but within the parallel framework and therefore in a format, which allows mechanistic knowledge to be incorporated at a later date.

As stated previously existing methods for black box modelling such as multilayer perceptrons and RBF networks suffer from overfitting. They therefore represent a major source of inaccuracy in hybrid models. Overfitting can be reduced by careful choosing the neural network architecture. For multilayer perceptrons this means choosing the number of layers and nodes per layer. For RBF networks the number and location of basis functions must be determined. An additional problem with such techniques is that training algorithms can become trapped in local optimum. Despite significant work in this area there is no generally accepted method for dealing with these problems. Therefore in chapter five Support vector machines Vapnik(1995) are proposed for modelling the kinetics of hybrid models. This relatively new technique is explicitly designed to avoid local optima and overfitting but has not yet been applied in this context.

4 The constraint matrix.

Recall the following model representation.

$$\frac{d\xi}{dt} = \overset{constraints}{\widetilde{K}} \times \underbrace{r(\xi,t)}_{kinetics} - \underbrace{D(t)\xi(t) + u(t)}_{transport} \tag{4.1}$$

This chapter considers the problem of determining the constraint matrix K so that it is consistent with the available measurements of the state variables ξ, the known external control actions, as well as any prior knowledge of the system.

The use of elemental balancing and knowledge of metabolic networks to derive the constraint matrix is reviewed. However, as stated in chapter two, for some industrial processes the composition of biomass, substrates and products may not be known. Therefore stoichiometric constraints cannot be derived from *a priori* knowledge. This issue motivates the use of data driven methods for reducing the number of degrees of freedom and consequently the number of kinetic functions that must be specified. Two such methods are described:

- A simple regression based method of estimating the unknown coefficients of a pre-specified reaction network.
- The use of principal component analysis to infer the number of degrees of freedom of the system and corresponding constraints.

4.1 Elemental balances

A bioprocess can be viewed in general form (Nielsen(2001); Stephanopoulos et al(1998)) as the conversion by biomass of some substrates $S = [S_0, S_1 \ldots S_n] \subset \xi$ to some products $P = [P_0, P_1 \ldots P_n] \subset \xi$ and more biomass X where $\overset{X}{\rightarrow}$ indicates conversion if and only if biomass is present.

$$\sum S \overset{X}{\rightarrow} X + \sum P \tag{4.2}$$

If the media is defined then this allows the calculation of elemental balances for carbon, hydrogen, oxygen, nitrogen, phosphate and sulphur for the system from the

chemical composition of the various species involved. In some cases it is possible to extend this by defining a generalised degree of reduction balance and a charge balance for the system (Roels(1983)). Consider the general reaction scheme shown in Figure 24.

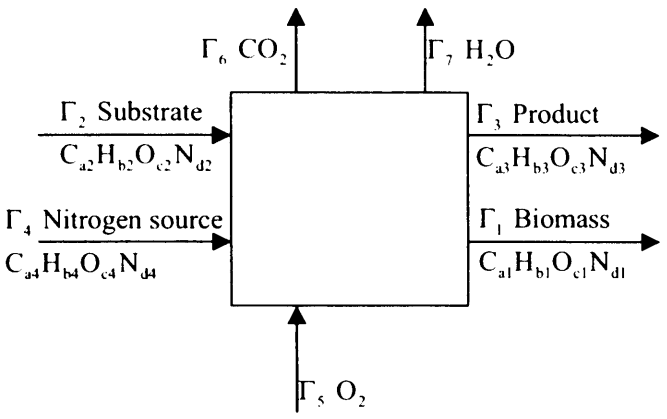


Figure 24. Generalised Elemental balance.

These elemental balances for C,N,H,O can be conveniently summarised by the following matrix equation involving the consumption/production rates and a elemental composition matrix. The columns represent each species and the rows the carbon, hydrogen, oxygen and nitrogen content. Where N_e is the number of elements and $\Gamma(t)$ is the molar flux due to reactions.

$$E \cdot \Gamma = 0$$

$$E \in \Re^{N_e \times N_s}$$

where

$$E = \begin{pmatrix} \underbrace{biomass}_{a_1} & \underbrace{substrate}_{a_2} & \underbrace{product}_{a_3} & \underbrace{Nitrogen\ source}_{a_4} & \underbrace{oxygen}_{0} & \underbrace{carbon\ dioxide}_{1} & \underbrace{water}_{0} & \leftarrow (C) \\ b_1 & b_2 & b_3 & b_4 & 0 & 0 & 2 & \leftarrow (H) \\ c_1 & c_2 & c_3 & c_4 & 2 & 2 & 1 & \leftarrow (O) \\ d_1 & d_2 & d_3 & d_4 & 0 & 0 & 0 & \leftarrow (N) \end{pmatrix} \quad (4.3)$$

Partitioning this into free rates Γ_{free} and calculated rates Γ_{calc} (4.3) can be written as:-

$$E\Gamma = E_{calc}\Gamma_{calc} + E_{free}\Gamma_{free} = 0 \quad (4.4)$$

Rearranging, the calculated rates are given by:-

$$\Gamma_{calc} = -E_{calc}^{-1} E_{free} \Gamma_{free} \tag{4.5}$$

As an illustration of the principle of elemental balancing consider the following black box model of the aerobic growth of Saccharomyces cerevisiae on a defined medium described in Stephanopoulos, G. N. et al(1998)).

The yeast grows aerobically with glucose as the carbon source and ammonia as the nitrogen source. The elemental composition of biomass for *S. cerevisiae* grown under glucose limited conditions (on a Cmol[20] basis) is typically $CH_{1.83}O_{0.56}N_{0.17}$

The overall conversion can be represented in Cmols as.

$$\underbrace{CH_2O}_{glucose} + \underbrace{NH_3}_{ammonium} + \underbrace{O_2}_{oxygen} \rightarrow \underbrace{CH_{1.83}O_{0.56}N_{0.17}}_{biomass} + \underbrace{CO_2}_{carbon\ dioxide} + \underbrace{H_2O}_{water} \tag{4.6}$$

and the rates of change of each species are specified by the following rate vector:

$$\Gamma = [-r_s, -r_o, r_c, \mu, -r_N, r_w]^T \tag{4.7}$$

The elemental composition matrix is:-

$$E = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 2 & 0 & 0 & 1.83 & 3 & 2 \\ 1 & 2 & 2 & 0.56 & 0 & 1 \\ 0 & 0 & 0 & 0.17 & 1 & 0 \end{pmatrix} \tag{4.8}$$

Hence by (4.5) the calculated rates can be found from the free rates as:-

$$\begin{pmatrix} -r_o \\ r_c \\ -r_N \\ r_w \end{pmatrix} = \begin{pmatrix} -1 & -1.05 \\ 1 & 1 \\ 0 & 0.17 \\ 1 & 0.66 \end{pmatrix} \begin{pmatrix} \mu \\ -r_s \end{pmatrix} \tag{4.9}$$

and the model becomes:-

$$\begin{pmatrix} r_{C_1} \\ r_{C_{m2}} \\ r_{C_{m2}} \\ r_{C_1} \\ r_{C_N} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & -1.05 \\ 1 & 1 \\ 1 & 0 \\ 0 & 0.17 \end{pmatrix} \begin{pmatrix} \mu \\ -r_s \end{pmatrix} \tag{4.10}$$

[20] By convention the elemental composition is normalised with respect to carbon content.

4.2 Metabolic network modelling.

In metabolic modelling a detailed list of the reaction stoichiometry describing how substrates are converted to metabolic products and biomass constituents or macromolecular pools is compiled. Evidence for the likely reaction network can be obtained from detailed literature surveys and phylogenetic analysis.

It is then assumed that the turn over of internal metabolites is very fast and therefore that with respect to larger time scales there is no metabolite accumulation in the intracellular pools.

$$0 = r_{met} - \mu X_{met} \tag{4.11}$$

The dilution term μX_{met} can be neglected therefore the total rate of production and consumption of each metabolite is equal.

$$\begin{aligned} &X_0 \xrightarrow[v_0]{} X_1 \xrightarrow[v_1]{} X_2 \\ &v_0 = v_1 \end{aligned} \tag{4.12}$$

A necessary consequence of this is that only metabolites at branch points need to be considered, since all reaction rates in a linear sequence of reactions must be equal and can therefore be considered as a single lumped reaction e.g. $X_0 \xrightarrow[v_1]{} X_2$, without altering the degrees of freedom of the system.

The reaction network can be written in matrix form by writing the stoichiometry as $G \in \Re^{N_r \times N_m}$ and the fluxes of each lumped reaction as a vector $\vartheta \in \Re^{N_m}$ implying a system of linear constraints

$$G\vartheta = r_{met} = 0 \tag{4.13}$$

The consistency of the network can be checked by applying the elemental composition matrix:-

$$GE = 0 \tag{4.14}$$

As an example of flux balance analysis consider citric acid fermentation by *Candida lipolytica*. The reaction network for this process as described by Aiba and Matsuoka (1970) and Stephanopolous 1998 is shown in Figure 25.

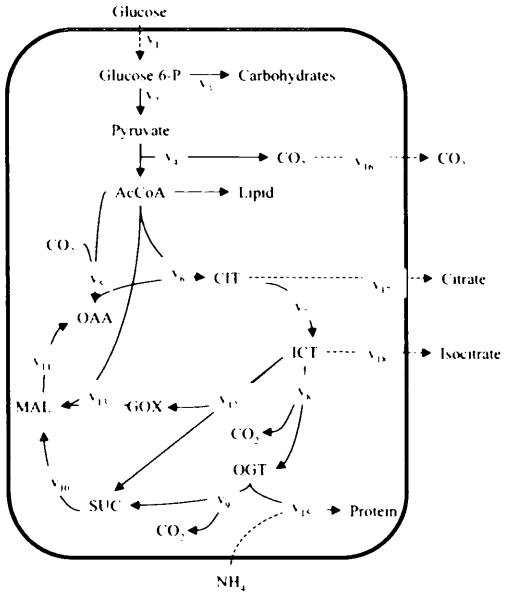


Figure 25. Simplified reaction network for *C. lipolytica*. Taken from Stephanopoulos, G. N. et al(1998).

In matrix form the reaction network shown in Figure 25 can be written as:-

$$\begin{array}{l} \text{Glucose}_{(\text{external})} \\ \text{Glucose6-P} \\ \text{Pyruvate} \\ \text{AcCoA} \\ \text{OAA} \\ \text{CIT} \\ \text{ICT} \\ \text{OGT} \\ \text{SUC} \\ \text{MAL} \\ \text{CO}_2\ (\text{external}) \end{array} \begin{pmatrix} 1 & -0.5 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \vartheta_1 \\ \vartheta_2 \\ \vartheta_3 \\ \vartheta_4 \\ \vartheta_5 \\ \vartheta_6 \\ \vartheta_7 \\ \vartheta_8 \\ \vartheta_9 \\ \vartheta_{10} \\ \vartheta_{11} \\ \vartheta_{12} \\ \vartheta_{13} \\ \vartheta_{14} \\ \vartheta_{15} \\ \vartheta_{16} \\ \vartheta_{17} \\ \vartheta_{18} \end{pmatrix}' = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (4.15)$$

Each row of the matrix represents a species and each column represents a reaction.

For example the second column represents reaction ϑ_2. It contains –0.5 in the first row and +1 in the second row showing that it consumes glucose-6-phosphate which and produces Pyruvate in a 1:2 ratio.

$$\text{Glucose6-P} \xrightarrow{\vartheta_2} 2\ \text{Pyruvate} \quad (4.16)$$

To obtain constraints the matrix (4.15) is partitioned into free/measured rates, and calculated rates.

$$0 = G_{free}\vartheta_{free} + G_{calc}\vartheta_{calc} \tag{4.17}$$

and the reaction rates calculated as $\vartheta_{calc} = -(G_{calc})^{-1} G_{free}\vartheta_{free}$ choosing the glucose uptake rate, carbon dioxide production rate, glycoxylate shunt, isocitrate production rate, protein synthesis rate and carbohydrate synthesis rate as the free reactions since these can be readily measured(with the exception of the shunt which is known to be zero under normal conditions) the other rates can be calculated as:-

$$\begin{pmatrix} \vartheta_2 \\ \vartheta_4 \\ \vartheta_5 \\ \vartheta_6 \\ \vartheta_7 \\ \vartheta_8 \\ \vartheta_9 \\ \vartheta_{10} \\ \vartheta_{11} \\ \vartheta_{13} \\ \vartheta_{14} \end{pmatrix} = \begin{pmatrix} 2 & -2 & 0 & 0 & 0 & 0 & 0 \\ 2 & -2 & 1 & -1 & 0 & -1 & -1 \\ 0 & 0 & -1 & 1 & 0 & 1 & 1 \\ -1 & 1 & 0 & 1.5 & 0.5 & 2 & 2 \\ -1 & 1 & 0 & 1.5 & 0.5 & 1 & 2 \\ -1 & 1 & -1 & 1.5 & 0.5 & 1 & 1 \\ -1 & 1 & -1 & 0.5 & 0.5 & 1 & 1 \\ -1 & 1 & 0 & 0.5 & 0.5 & 1 & 1 \\ -1 & 1 & 1 & 0.5 & 0.5 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & -3 & 0 & -2.5 & -0.5 & -3 & -3 \end{pmatrix} \begin{pmatrix} r_{glc} \\ r_{car} \\ r_{shunt} \\ r_{prot} \\ r_{c} \\ r_{cit} \\ r_{act} \end{pmatrix} \tag{4.18}$$

4.3 Theoretical basis for inferring the constraint matrix from data.

4.3.1 Problem statement.

As we have seen from the previous approaches, in many bio-process systems the degrees of freedom of the system under typical conditions can be significantly lower than the number of reactions or indeed measured components. $N_\xi > N_r = \text{rank}(K)$.

Firstly this suggests that given certain restrictions it should be, in principle, possible to identify K from data. Secondly, it suggests that by so doing the overall problem of modelling the system, including kinetics, can be simplified.

The rate of change of a component due to reaction effects, as distinct from flow effects, is denoted as $\Gamma(t)$ defined in (4.19).

$$\Gamma(t) = \underbrace{Kr(\xi,t)}_{kinetics} = \frac{d\xi(t)}{dt} + \underbrace{D(t)\xi - u(t)}_{transport} \tag{4.19}$$

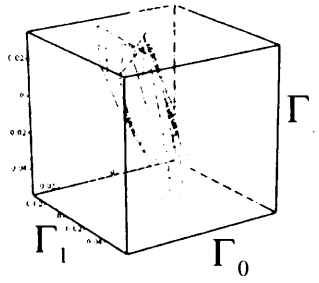


Figure 26. Illustration of system behaviour restricted to a 2 dimensional manifold in 3 dimensional space.

Viewing $\Gamma(t)$ as a coordinate vector the existence of matrix $K \in \Re^{N\xi \times Nr}$ implies that all allowable coordinates, and therefore all members of Γ, lie on some linear manifold of dimension equal to the number of free reactions. Figure 26 illustrates this for a simple case of 3 measured states and 2 degrees of freedom. The challenge is to constrain the inferred model to the same manifold as the true system.

4.3.2 Removing transport effects.

If a continuous signal $\hat{\xi}^*(t)$ for the state vector is available by way of measurement, state observer or interpolation, then an estimate of $\Gamma(t)$ can be found with a finite difference approximation such as:-

$$\hat{\Gamma}^*(t) = \frac{\hat{\xi}^*(t+\Delta t) - \hat{\xi}^*(t)}{\Delta t} + \frac{D(t)\hat{\xi}^*(t) - u(t)}{2} + \frac{D(t+\Delta t)\hat{\xi}^*(t+\Delta t) - u(t+\Delta t)}{2}$$
$$\hat{\Gamma}^*(t) \cong Kr(\xi(t), t) \tag{4.20}$$

In practice the state vector will only be known at discrete sample intervals, represented by a row vector of sample times $t_s = \left[t_0, t_s^0, \ldots t_s^{\langle N_{samples}-1 \rangle} \right]$. In that case it is better to consider the cumulative consumption/production of the measured species at each sample point by integrating equation (4.19) between t_0 and each sample point $t_s^{\langle i \rangle}\ i = 1, \ldots \left(N_{samples} - 1 \right)$.

$$\eta^*_{t=t_s^{\langle i \rangle}} = \underbrace{\xi^*_{t=t_s^{\langle i \rangle}}}_{\text{concentrations at time t}} - \underbrace{\xi^*_{t=0}}_{\text{concentrations at time 0}} + \underbrace{\int_0^{t_s^{\langle i \rangle}} D(\tau)\xi(\tau) - u(\tau)d\tau}_{\text{mass changes due to externals}} \tag{4.21}$$

Initially we seem to be no further forward since it appears that $\xi(t)$ needs to be known continuously in order to evaluate equation (4.21). However integration using $e^{\int D(k)dk}$ as an integrating factor leads to the following result.

$$\eta^{\bullet}_{t=t_s^{\langle i\rangle}} = \xi^{\bullet}_{t=t_s^{\langle i\rangle}} - \xi^{\bullet}_{t=0} e^{\int_0^{t_s^{\langle i\rangle}} D(k)dk} - \left(\int_0^{t_s^{\langle i\rangle}} u(\tau) e^{\int_0^{\tau} D(k)dk} d\tau \right) e^{\int_0^{t_s^{\langle i\rangle}} D(k)dk} \qquad (4.22)$$

Some intuitive understanding[21] of this equation can be gained by considering a stirred tank containing a solution of A at initial concentration $\xi_A(0)$. No reactions occur and the feed is pure water. The well mixed system is described by $\frac{d\xi_A(t)}{dt} = -D(t)\xi_A(t)$ and hence the concentration at time $t_s^{\langle i\rangle}$ is given by $\xi_A(t_s^{\langle i\rangle}) = \xi_A(0) e^{\int_0^{t_s^{\langle i\rangle}} D(\tau)d\tau}$. (Notice that this is the second term of (4.22)).

Suppose that an amount u_A per unit volume of A is added to the reactor at time t_{add}. The concentration at time t_{add} is $\xi_A(t_{add}) = \xi_A(0) e^{-\int_0^{t_{add}} D(\tau)d\tau} + u_A$. This is equivalent to having started with an initial concentration of $\xi_A(0) + u_a e^{\int_0^{t_{add}} D(\tau)d\tau}$ which was then diluted.

When equation (4.22) is applied to a stirred tank reactor, $\eta^{\bullet}_{t=t_s^{\langle i\rangle}}$ can be seen as being calculated from the difference between the measured concentration $\xi^{\bullet}_{t=t_s^{\langle i\rangle}}$ and the concentration that would have been measured at time $t_s^{\langle i\rangle}$ had no reactions occurred.

4.3.3 Identification of K by regression.

In many cases prior knowledge of the system will allow the matrix 'K' to be described in terms of a plausible reaction network with unknown coefficients. It should be noted that the true reaction network is usually very large and that the term *'plausible network'* refers to a much smaller system of lumped reactions that

[21] Financially minded readers may wish to consider the similarities between this equation and one for net present value at time t, where the continuously compounded interest rate is given by $D(t)$ and cash flow by another function $u(t)$ of time

approximates the normal metabolism of the organism. Haag et al(2005) shows that systems with complex intracellular reaction networks can be represented by macroscopic reactions relating extracellular components only.

Bogaerts et al(2003); Chen and Bastin(1996) outline the following method which can be used in the above situation to identify the unknown coefficients of K, which they term the *'pseudo stoichiometric matrix'*.:

Method:

The proposed reaction network is written in the usual form but with the constraint matrix containing some unknown coefficients.

$$\frac{d\xi}{dt} = Kr(\xi,t) - D(t)\xi(t) + u(t) \tag{4.23}$$

The constraint matrix is then partitioned after some permutation E where $EK^T = [K_a^T, K_b^T]$ to form two matrices $K_a \in \Re^{p\times Nr}$ and $K_b \in \Re^{(N\xi - p)\times Nr}$ where $p = rank(K)$ such that K_a is of full row rank. The corresponding partitions are applied to the state vector ξ and transport terms to give: $E\xi^T = [\xi_a^T, \xi_b^T]$, $Eu^T = [u_a^T, u_b^T]$.

The overall model can then be written as two sub models:

$$\begin{aligned}\frac{d\xi_a(t)}{dt} &= K_a r(\xi,t) - D(t)\xi_a(t) + u_a(t)\\ \frac{d\xi_b(t)}{dt} &= K_b r(\xi,t) - D(t)\xi_b(t) + u_b(t)\end{aligned} \tag{4.24}$$

There exists a unique solution $C \in \Re^{(N_\xi - p)\times p}$ to the matrix equation shown below

$$CK_a + K_b = 0 \tag{4.25}$$

This solution is given by $C = -K_b K_a^{-1}$. Using C we define an auxiliary vector.

$$z = C\xi_a + \xi_b \tag{4.26}$$

The model can be written in terms of this auxiliary vector as:

$$\frac{dz(t)}{dt} = -D(t)z(t) + Cu_a(t) + u_b(t) \tag{4.27}$$

$$\xi_b = z - C\xi_a \tag{4.28}$$

Integrating the equation (4.27) for the dynamics of $z(t)$ gives:-

$$z(t)=\left(Z(0)+\int_0^t\left(Cu_a(\tau)+u_b(\tau)\right)e^{\int_\tau^t D(k)dk}d\tau\right)e^{\int_0^t D(k)dk} \tag{4.29}$$

Thus substituting this result for $z(t)$ into (4.28) the following relation between $\xi_a(t)$ and $\xi_b(t)$ is obtained.

$$\xi_b(t)=\left(Z(0)+\int_0^t\left(Cu_a(\tau)+u_b(\tau)\right)e^{\int_\tau^t D(k)dk}d\tau\right)e^{\int_0^t D(k)dk}+C\xi_a(t) \tag{4.30}$$

The relation should holds if C is correct. This leads to the following linear regression that can be used to identify the coefficients of C.

$$\min_C\left(\sum_{i=0}^{n}\left(\xi_b(t_i^{(j)})-\left(C\xi_a(0)+\xi_b(0)+\int_0^{t_i^{(j)}}\left(Cu_a(\tau)+u_b(\tau)\right)e^{\int_\tau^t D(k)dk}d\tau\right)e^{-\int_0^{t_i^{(j)}} D(k)dk}+C\xi_a(t_i^{(j)})\right)^2\right) \tag{4.31}$$

Having found C the coefficients of K can be found from the definition of C as:-

$$C=-K_bK_a^{-1} \tag{4.32}$$

Since C is known (4.32) defines a system of simultaneous equations. Depending on the structure of the reaction network there may or may not be a unique solution for the elements of K.

Bernard and Bastin(2005); Chen, L. et al(1996) give a detailed analysis of the conditions known as *'C-identifiability'* under which the algebraic relations defining $C=-K_bK_a^{-1}$ can be used to uniquely identify the values of the coefficients of K.

> *"The vector containing all the coefficients to be identified in the j^{th} column of the matrix K will be denoted $k^{(j)}$ and k is the union of all these vectors $k^T=[k^{(1)T}\cdots k^{(Nr)T}]$. The elements of $k^{(j)}$ are defined as C-identifiable if and only if there exists at least one partition $EK^T=[K_a^T,K_b^T]$ which is non singular and where K_a does not contain any element of $k^{(j)}$."*
>
> **-(Chen, L. et al(1996))**

Comment.

Relation (4.30) and the above derivation, while apparently quite complex, simply states that the cumulative consumption of ξ_a and ξ_b are related by the matrix C.

$$\eta_b(t)=-C\eta_a(t) \tag{4.33}$$

Proof:

Substituting (4.26) for $z(t)$ in equation (4.29) gives:

$$C\xi_a(t)+\xi_b(t)=\left(C\xi_a(0)+\xi_b(0)+\int_0^t\left(Cu_a(\tau)+u_b(\tau)\right)e^{\int_\tau^t D(k)dk}d\tau\right)e^{-\int_0^t D(k)dk} \quad (4.34)$$

Expanding the integral and collecting the $C\xi_a$ terms on the left hand side and ξ_b on the right hand side gives:

$$C\left(\xi_a(t)-\left(\xi_a(0)+\int_0^t u_a(\tau)e^{\int_\tau^t D(k)dk}d\tau\right)e^{-\int_0^t D(k)dk}\right)=-\left(\xi_b(t)-\left(\xi_b(0)+\int_0^t u_b(\tau)e^{\int_\tau^t D(k)dk}d\tau\right)e^{-\int_0^t D(k)dk}\right) \quad (4.35)$$

Substituting in $\eta(t)$ as defined in equation (4.22) gives the required result: $\eta_b(t)=-C\eta_a(t)$

Summary of the regression method.

Since $C=-K_bK_a^{-1}$, equation (4.33) can be seen as finding $\eta_b(t)$ by projecting the cumulative consumption $\eta_a(t)$ of ξ_a through the inverse matrix K_a^{-1} to obtain the integral of the free reaction rates $\int_0^t r(\xi,\tau)d\tau=K_a^{-1}\eta_a(t)$, then through the matrix K_b, to obtain the cumulative consumption of the other elements $\eta_b(t)=k_b\int_0^t r(\xi,\tau)d\tau$.

Thus, there are two simple regression[22] methods for identification of the unknown elements of a proposed reaction network.

1. Linear regression of the form:-

$$\min_C\left(\sum_{j=0}^{N_\xi-1}\left(\frac{1}{\sigma_j^{\,2}}\left(\sum_{i=0}^{t-1}\left(\eta_b\Big|_{t=t_x^{\langle i\rangle}}+C\eta_a\Big|_{t=t_x^{\langle i\rangle}}\right)^2\right)_j\right)\right) \quad (4.36)$$

The non zero elements of C are chosen as to minimise $\eta_b(t)+C\eta_a(t)$ evaluated at each sample point. σ_j is the variance due to noise of each

[22] How this fits into the maximum posterior formulation involving priors over the parameters should be clear from the weighed regressions and the definition of Bayesian parameter identification in Chapter 8.

measured series or some other regression weight. The elements K are then identified from the algebraic definition of C (4.32) using standard methods.

2. Direct non-linear regression of the form:-

$$\min_{K}\left(\sum_{t=0}^{N_t-1}\left(\frac{1}{\sigma_t^2}\left(\sum_{i=0}^{t-1}\left(\eta_{b\,t-t_s^{(i)}}^{\bullet}-K_bK_a^{-1}\eta_{a\,t-t_s^{(i)}}^{\bullet}\right)^2\right)_t\right)\right) \tag{4.37}$$

The elements of K are chosen so as to minimise $\eta_b(t)+K_bK_a^{-1}\eta_a(t)$ evaluated at each sample point.

4.3.4 Identification of K by principal component analysis.

A method is now outlined for identification of the constraint matrix that does not require any prior knowledge of the constraint matrix. Consider a scatter plot (Illustrated in Figure 27.) for a system with three measured components and two degrees of freedom where the coordinates of each point are given by $\eta_{t=t_s}^{\bullet}$.

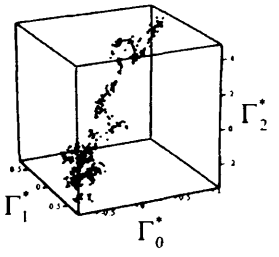


Figure 27. Illustrative scatter plot of data.

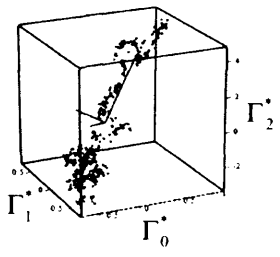


Figure 28. Illustration of principal components.

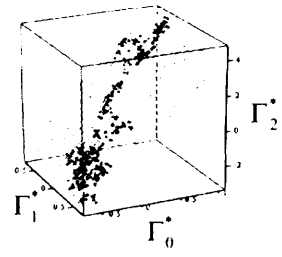


Figure 29. Illustration of linear manifold defined by first 2 principal components.

Recall that the mass balance restrictions restrict $\Gamma(t)$ (or equivalently its integral $\eta(t)$) to some linear manifold. It is therefore the case that in the absence of measurement error the data points would all lie on a 2d surface (Figure 29). The problem of finding K can be visualised as finding this surface from the data and thus restricting the model to this lower dimensional manifold.

Note however that if the reaction network is unknown, the dimension of the manifold is also unknown, although it must be $\leq \dim(N_{\xi})$. Knowing the dimension of the manifold is equivalent to knowing the rank of K and thus the number of free reactions.

If measurement noise is present, and the variance due to noise is less than the variance due to reactions, then the data will be restricted to lie close to the manifold. Finding the manifold reduces to the problem of finding directions where variance is greater than variance due to noise (Figure 28). The standard algorithm for obtaining the directions of maximum variance is *principal component analysis* (see Howard Anton(2000) for a tutorial). The objective of *principal component analysis* (PCA) is to obtain mutually uncorrelated vectors, which explain the variance of the data.

The principal component algorithm is as follows:-

1. Form the data into a single vector η containing all the time points from all the training batches.

$$\eta \in \Re^{(\ell)\times N_c} \tag{4.38}$$

where ℓ is the number of points and N_c is the number of measured components.

2. So as to give equal weighting to all variable the data vectors are normalised as follows by dividing by the standard deviation of each vector and subtracting the column means:-

$$\hat{\eta}^i = \frac{\eta^i - mean(\eta^i)}{\sqrt{\sigma_i}} \tag{4.39}$$

3. Calculate the covariance matrix:-

$$C^{N_c \times N_c} = \left(c_{i,j}, c_{i,j} = \mathrm{cov}(\hat{\eta}^i, \hat{\eta}^j)\right)$$

$$where \quad \mathrm{cov}(X,Y) = \frac{\sum_{i=0}^{\ell-1}\left(X_i - \bar{X}\right)\left(Y_i - \bar{Y}\right)}{\ell} \tag{4.40}$$

4. Calculate the eigen decomposition of the covariance matrix $\lambda V = CV$. The eigen vectors $V \equiv \left[pca_1 \quad pca_1 \quad \ldots \quad pca_n\right]$ are orthogonal unit vectors along which variation occurs. The eigenvalues 'λ' are the magnitude of this variation sorted in order of decreasing magnitude.

5. Retain the smallest number of eigenvectors which explain the required proportion of the normalised variance. The proportion of variance explained is given by (4.41) where k is the sum of all eigenvalues. If the noise level is known this is simply one minus the normalised noise level.

$$\min(n)$$

$$subject\ to\ constraint \sum_{i=0}^{n}\frac{\lambda_i}{k} \geq \left(1 - \sigma_{noise}\right) \tag{4.41}$$

6. Define the matrix of retained vectors as the principal component matrix M.

In the case of a noiseless system the manifold can be characterised by the vectors corresponding to the non zero eigenvalues of the PCA matrix. For a system with noise the manifold is characterised by the eigenvectors corresponding to the n biggest eigenvalues, where n is obtained from (4.41). Having obtained the principal components, the standard model representation (4.1) can be replaced by an equivalent model of the form

$$\frac{d\xi}{dt} = M \times \underbrace{p(\xi, v)}_{PCAs} - \underbrace{D\xi - Q(\xi) + F}_{transport} + \bar{\Gamma}^{\bullet} \tag{4.42}$$

Where $\bar{\Gamma}^{\bullet}$ are the mean measured rates of change of each component. Note that the kinetic components $p(\xi, v)$ are not the "real" reactions $r(\xi, v)$, but if the available data is representative of the variance of the system[23] the manifold of allowable $\Gamma(t)$'s or $\eta(t)$'s will be the same as the true system. The kinetic components $p(\xi, v)$ are simply orthogonal coordinates for defining a point on this manifold.

4.4 Inferring the constraint matrix from data – a simulated example.

4.4.1 Test system.

Three batches of data were generated by choosing random initial conditions for each batch and simulating the following dynamical system for 1000 time steps with 'samples' taken every 100 time steps, to which Gaussian $\sim N(\mu = 0, \sigma = 0.1)$ noise was added.

$$\frac{d\xi(t)}{dt} = \begin{pmatrix} -0.956 & 0 & 0 \\ 1 & -0.462 & 0 \\ 0.539 & 0 & -0.997 \\ 0 & 0 & 0.611 \\ 0 & 1 & 1 \\ 0 & -0.862 & -0.266 \\ 0 & 0.78 & 0.84 \end{pmatrix} \begin{pmatrix} \dfrac{0.94\xi_0}{1.693 + \xi_0} + \varepsilon_0 \\ 0.022\xi_1 + \varepsilon_1 \\ 0.69\xi_2\xi_5 + \varepsilon_2 \end{pmatrix} + D(t)\left(\xi_{in} - \xi(t)\right) \tag{4.43}$$

The dilution rate increases linearly with time $D(t) = 0.1t$ and the feed concentration is the same as the initial media concentration $\xi_{in} = \xi_{t=0}$. To emphasise that the identification procedure requires no knowledge of the reaction kinetics the reaction

[23] If the available data is gathered under restricted conditions such as a reaction rate being zero at all times then the manifold discovered by PCA will be that of the restricted system not the unrestricted system.

rates r_0, r_1, r_2 include a stochastic term $\varepsilon \in \Re^3$ randomly drawn from a three-dimensional uniform distribution $\sim U(0,2)$ at every time step. Initial media concentrations for each batch were drawn from $\sim U(0,5)$. The resulting training data consists of the full state $\xi^*_{t-\tau} \in \Re^7$ at each of the $t = N_{meas} = 30$ sample points and is presented in Figure 30 below. Testing data consisted of three batches drawn at random in the same manner but with noiseless 'measurements' at every 15 time steps rather than every 100.

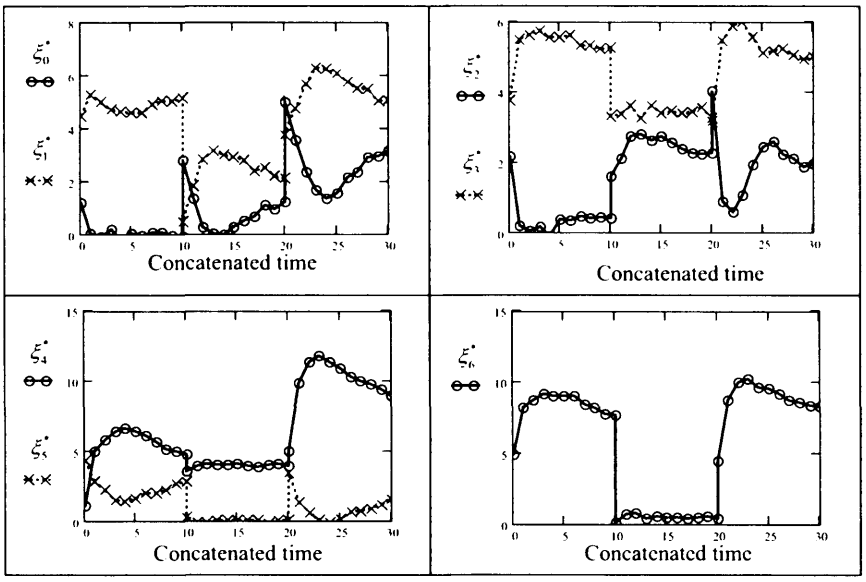


Figure 30. Training data generated by simulated system showing the concentration of each species against time for three concatenated batches.

The y axis give the 'concentration'[24] of each species. The x axis is 'concatenated time', which means that the start time of the second batch is the end time of the first batch. Thus the concatenated time of sample point i from batch b is given by:-

$$concatenated\_time(b,i) = b \times t_s^{\langle N_{meas}-1 \rangle} + t_s^{\langle i \rangle} \qquad (4.44)$$

The use of 'concatenated time' is simply to allow multiple batches of data to be presented on a single graph.

4.4.2 Applying the regression and PCA methods.

Both methods require transport effects to be removed, using (4.22) to obtain the cumulative production for each measured species. This gives the graphs of $\eta(t)$ against time as shown for each species in Figure 31 below.

[24] Dimensionless since its just a simulation.

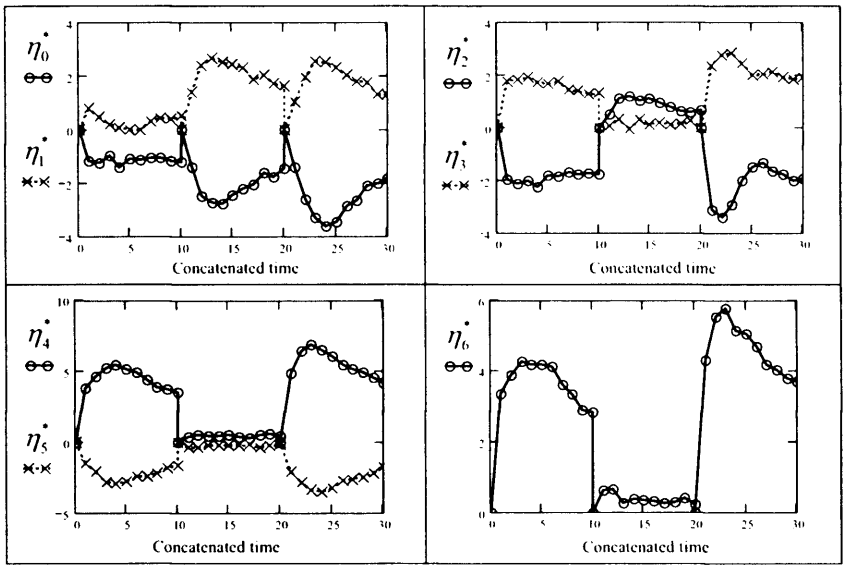


Figure 31. Cumulative change in each species against concatenated time.

Regression technique.

A reaction network is proposed. For this simulated example it is assumed that the network structure is correctly known but not the coefficients.

$$K = \begin{pmatrix} -k_0 & 0 & 0 \\ 1 & -k_2 & 0 \\ k_1 & 0 & -k_5 \\ 0 & 0 & k_6 \\ 0 & 1 & 1 \\ 0 & -k_3 & -k_7 \\ 0 & k_4 & k_8 \end{pmatrix} \tag{4.45}$$

Then partitioned into two groups.

$$K_a = \begin{pmatrix} 1 & -k_2 & 0 \\ 0 & 0 & k_6 \\ 0 & 1 & 1 \end{pmatrix}, K_b = \begin{pmatrix} -k_0 & 0 & 0 \\ k_1 & 0 & -k_5 \\ 0 & -k_3 & -k_7 \\ 0 & k_4 & k_8 \end{pmatrix} \tag{4.46}$$

The C matrix is given by $C = -Kb \cdot Ka^{-1}$ and hence:

$$-K_b K_a^{-1} = \begin{pmatrix} -k_0 & k_0 \dfrac{k_2}{k_6} & -k_0 k_2 \\ k_1 & -k_1 \dfrac{k_2}{k_6} - \dfrac{k_5}{k_6} & k_1 k_2 \\ 0 & \dfrac{k_3}{k_6} - \dfrac{k_7}{k_6} & -k_3 \\ 0 & \dfrac{-k_4}{k_6} + \dfrac{k_8}{k_6} & k_4 \end{pmatrix} \tag{4.47}$$

The following relation is used to identify the non zero elements of the C matrix.

$$\eta_b = \begin{pmatrix} c_1 & c_3 & c_7 \\ c_2 & c_4 & c_8 \\ 0 & c_5 & c_9 \\ 0 & c_6 & c_{10} \end{pmatrix} \eta_a \tag{4.48}$$

leading to:

$$C = \begin{pmatrix} -0.962 & 0.644 & -0.413 \\ 0.483 & -1.826 & 0.183 \\ 0 & 0.669 & -0.753 \\ 0 & 0.468 & 0.652 \end{pmatrix} \tag{4.49}$$

Figure 32 below shows the estimate of η_b from $C\eta_a$ against the true value of η_b using the above best fit values for the coefficients of C.

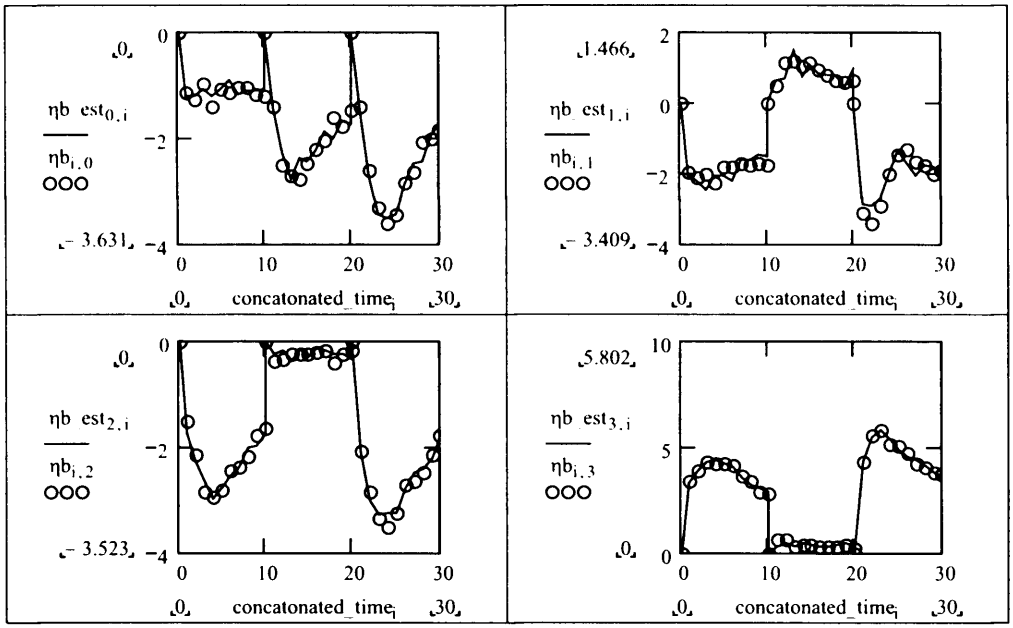


Figure 32. Fit of regression method to training data. Points show η_b, solid line shows regression estimate of η_b from the identified C matrix $\eta_b = C\eta_a$.

Using the algebraic definition of the C matrix given in (4.32) the estimated values of the coefficients of the K matrix can be obtained by elimination.

Estimate K matrix	True K matrix
$\hat{K} = \begin{pmatrix} -0.965 & 0 & 0 \\ 1 & -0.418 & 0 \\ 0.476 & 0 & -0.946 \\ 0 & 0 & 0.627 \\ 0 & 1 & 1 \\ 0 & -0.753 & -0.334 \\ 0 & 0.652 & 0.946 \end{pmatrix}$	$\begin{pmatrix} -0.956 & 0 & 0 \\ 1 & -0.462 & 0 \\ 0.539 & 0 & -0.997 \\ 0 & 0 & 0.611 \\ 0 & 1 & 1 \\ 0 & -0.862 & -0.266 \\ 0 & 0.78 & 0.84 \end{pmatrix}$

The fit to unseen testing data is shown in Figure 33 below. This shows the estimated ξ_b obtained by projecting unseen rates onto a 3 dimensional reaction matrix using $\hat{K}_a^{-1}$ and recovering using $\hat{K}$ compared with the full state vector for the true system. As Figure 33 shows the regression estimates of the state on testing data are almost indistinguishable from the true state.

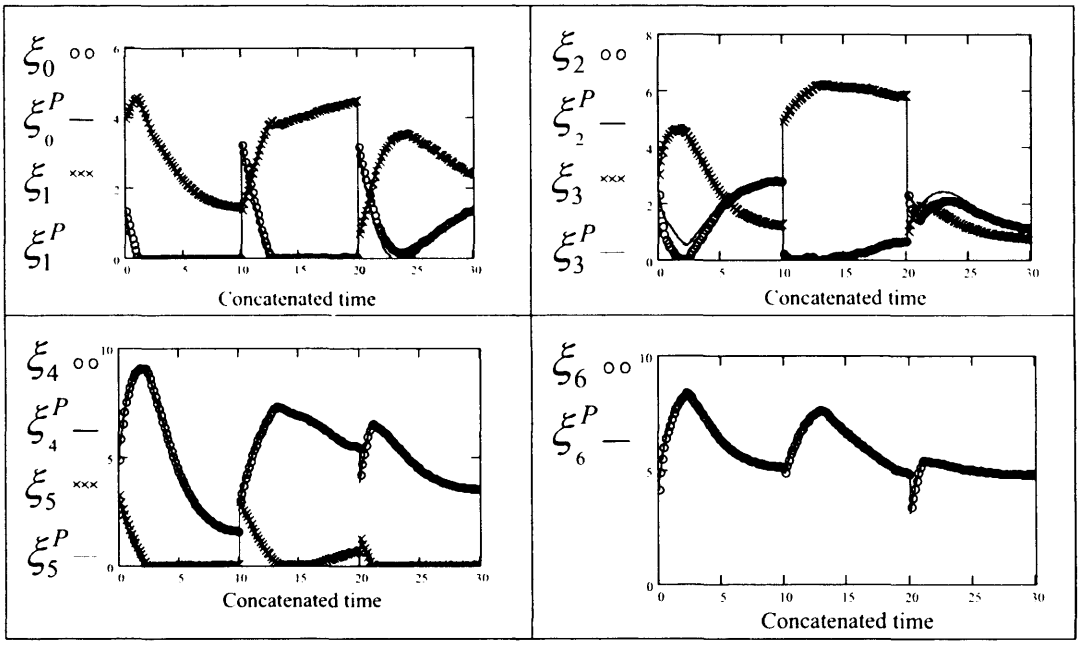


Figure 33. Fit of regression method to testing data. Points show noiseless testing data at 200 points. Solid Lines show regression based estimate of state at these same points.

PCA algorithm.

Applying the PCA algorithm it can be seen from Figure 34 that there are three significant eigenvalues with corresponding eigenvectors given by M.

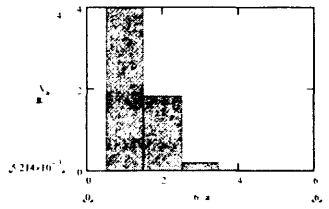


Figure 34. Eigenvalues of principal components against index.

$$K = \begin{pmatrix} -0.031 & -0.663 & -0.076 \\ -0.353 & 0.686 & -0.028 \\ 0.706 & 0.288 & -0.356 \\ -0.308 & 0.033 & 0.25 \\ 0.241 & 0.046 & 0.645 \\ -0.472 & -0.036 & -0.317 \\ 0.019 & 0.042 & 0.537 \end{pmatrix}$$

Figure 35. Eigenvectors for the first 3 principal components.

The fit to training data of the resulting PCA model is shown in Figure 36 and to unseen testing data in Figure 37. Points show the seven dimensional data generated from the true system compared with the same data projected on to a three dimensional PCA matrix $p \in \Re^{l \times 3}$ through the matrix $p = M^T(\eta - \overline{\eta})$ and then recovered from these using $\eta^p = Mp + \overline{\eta}$. On noisy training data the PCA estimate of the state ξ^p can be seen to be a smoothed version of the measured state ξ^*. On testing data the PCA estimates of the state are indistinguishable from the true state.

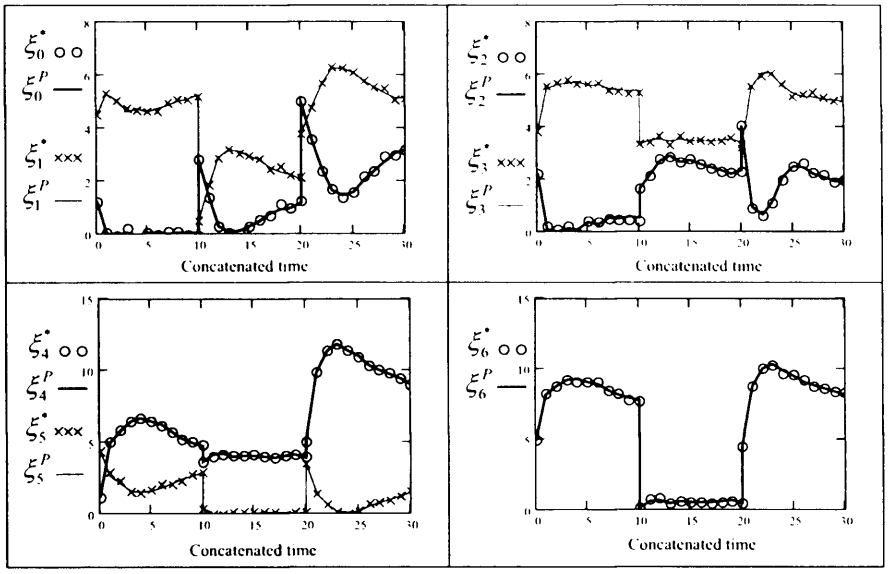


Figure 36. Fit of PCA method to training data. Solid Line shows PCA estimate of state, Points show training data.

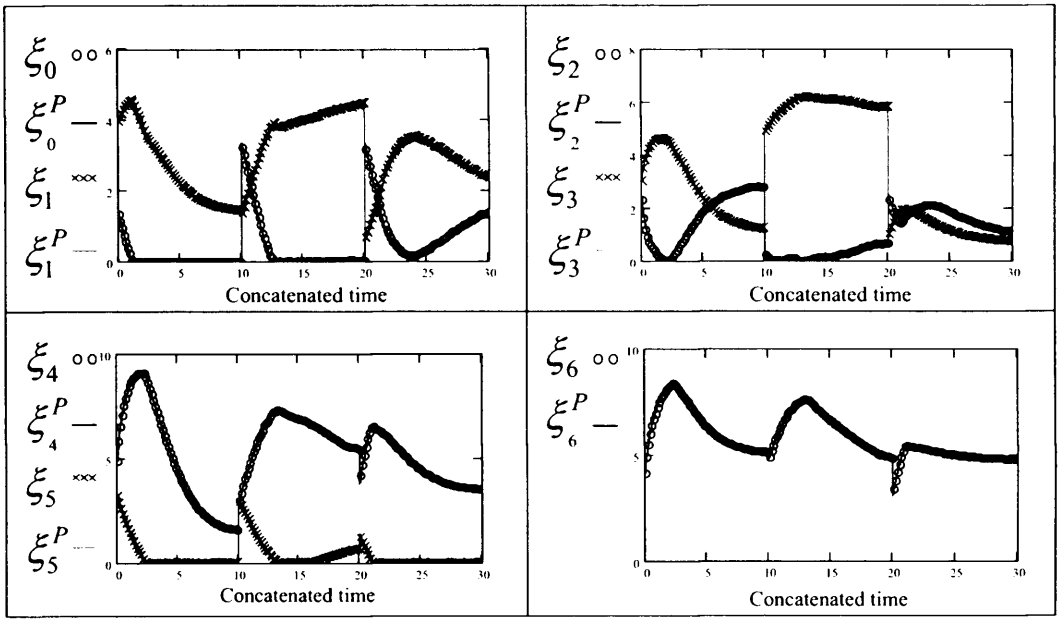


Figure 37. Fit of PCA method to testing data. Points show noiseless testing data at 200 points. Solid Lines show PCA based estimate of state at these same points.

Discussion.

The graphs of seen and unseen data show that both methods perform well on the simulated example. It is interesting to compare the lower dimensional manifolds found using the PCA and regression methods to those of the true system. In order to do this different systems are specified in terms of the same coordinates. The

constraints are transformed[25] so that η_1, η_3, η_4 are the free components and the system is described by $\eta = Z(\eta_1 \quad \eta_3 \quad \eta_4)^T$ where Z is the matrix defined in Table 2 below.

Table 2. Comparison of manifolds found using the two methods.

	True system.	Regression method.	PCA method.
Transform	$K \times \begin{pmatrix} K_{(1)} \\ K_{(3)} \\ K_{(4)} \end{pmatrix}^{-1}$	$\hat{K} \times \begin{pmatrix} \hat{K}_{(1)} \\ \hat{K}_{(3)} \\ \hat{K}_{(4)} \end{pmatrix}^{-1}$	$M \times \begin{pmatrix} M_{(1)} \\ M_{(3)} \\ M_{(4)} \end{pmatrix}^{-1}$
Manifold.	$\begin{pmatrix} -0.956 & 0.722 & -0.442 \\ 1 & 0 & 0 \\ 0.539 & -2.038 & 0.249 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0.975 & -0.862 \\ 0 & 0.099 & 0.78 \end{pmatrix}$	$\begin{pmatrix} -0.956 & 0.643 & -0.403 \\ 1 & 0 & 0 \\ 0.476 & -1.826 & 0.199 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0.668 & -0.753 \\ 0 & 0.469 & 0.652 \end{pmatrix}$	$\begin{pmatrix} -0.974 & 0.838 & -0.485 \\ 1 & 0 & 0 \\ 0.511 & -2.526 & 0.45 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.04 & 0.913 & -0.848 \\ -4.238\times10^{-3} & 0.455 & 0.655 \end{pmatrix}$

The manifolds found by both methods using these transformed coordinates are orientated similarly to the true manifold, although there are clearly differences between the estimated coefficients and the true ones. It is particularly noteworthy that even though no structure was imposed, and principal components in no way reflect actual reactions, the PCA method found almost the same manifold as the regression technique. However, if *a priori* information about the values of constants is available, then the regression technique can make use of this information to further constrain the system whereas the PCA method cannot.

[25] See Appendix C2.

4.5 Conclusion.

The problem of determining the constraint matrix for models of biological systems was considered. The approaches of elemental balancing and metabolic modelling were reviewed and then two methods were demonstrated for inferring the constraints from data. The available approaches are summarised in Table 3 below.

Table 3 Available techniques for determining the stoichiometric constraints.

Level of *a priori* knowledge	**Appropriate method for determining constraints**
Stoichiometric relations known.	Constraints can be derived by matrix manipulations following the method of metabolic flux analysis.
Reaction network is known and elemental composition of species is known	The number of degrees of freedom of the system can be reduced by elemental, generalised degree of reduction and a charge balances for the system.
Reaction network is known but insufficient knowledge is available to constrain the system	Unknown stoichiometric coefficients can be estimated by regression.
No knowledge of the reaction network is available.	Principle component analysis can be used to estimate the lower dimensional manifold to which the system is constrained.

The *"regression method"* involves fitting the unknown constants of a pre-specified reaction network to data. This method provides a flexible structure that can make use of prior knowledge about the coefficients of that network. The second *"PCA method"* requires no *a priori* knowledge and involves directly finding the lower dimensional manifold to which the system is constrained. Unfortunately, the PCA matrix cannot make use of prior knowledge and the kinetic functions corresponding to each eigenvector cannot be easily interpreted in terms of biologically meaningful reactions. Both techniques performed well on the simulated example. However fully kinetic models were not tested. In chapter seven, fully kinetic models built using the PCA technique to infer the constraints were compared with models built knowing the correct constraint matrix.

5 Data-driven Modelling of a simulated mammalian cell culture process using Support Vector Machines[26].

As mentioned in chapter 3 one of the purported advantages of the serial approach to hybrid modelling is that models can be quickly inferred from data thus providing a time saving compared with detailed mechanistic modelling. However with current neural network approaches an involved and time-consuming methodology is required to minimise overfitting. For Radial basis networks the selection of the number of basis functions as well as their location and width need to be determined. For multilayer perceptrons the number of neurons and hidden layers needs to be determined. With both of these networks strategies need to be employed to prevent the selection of large weights and multiple runs are necessary to ensure that the training process has not converged to a local optimum.

This chapter describes the application of Support Vector Machines (SVM); a learning system based on statistical learning theory, to bioprocess modelling. SVMs avoid the above difficulties with local optima and determining the architecture. The architecture is determined implicitly from training data on the basis of two hyper parameters which can be efficiently determined on the basis of validation data. In this chapter as a case study a SVM based system is trained on data simulated by a published model of a hybridoma culture with added Gaussian noise.

5.1. Statement of the problem.

5.1.1 Kinetic modelling.

This section considers the problem of inferring a model of the reaction kinetics from the available data $r(\xi,v)$. Recall the general state space representation of a bio-reactor:-

$$\frac{d\xi}{dt} = Kr(\xi,v) - D\xi + u \tag{5.1}$$

[26] This chapter and chapter 7 are adapted from the paper; "Data-driven Modelling of a simulated mammalian cell culture process Using Support Vector Machines; a comparison with genetic programming and neural networks." Hodgson & Baganz, Computers in Chemical Engineering (Submitted).

The previous sections detailed methods of obtaining the pseudo-stoichiometric matrix K. The reaction network imposes the following *a priori* restriction on the kinetics.

$$r_i(\xi,v)=\begin{cases} 0 & \text{if}\left(\prod_{s=N_-} a(s,i)=0\right) \\ f(\xi,v) & otherwise \end{cases}$$

$$where \qquad (5.2)$$

$$a(S,i)=\begin{cases} \xi_s \text{ if}(K_{s,i}<0) \\ 1 \text{ if}(K_{s,i}\geq 0) \end{cases}$$

Equation (5.2) simply states that each reaction can only occur if all the species consumed by that reaction are present and that its rate, if it does occur, can be modelled by some unknown, possibly non linear, kinetic function of the concentrations in the reactor and environmental conditions. Clearly other such necessary conditions can be imposed based on the knowledge of the engineer.

Denote as $\delta(\xi)$ a function which returns a vector indicating whether each reaction can proceed, thus the model becomes:-

$$\frac{d\xi}{dt}=K\left(\delta(\xi)f(\xi,v)\right)-D\xi+u \qquad (5.3)$$

The approach taken is to infer each kinetic sub model $f(\xi,v)$ separately so each model produces the output implied by the available data. If the constraint matrix K is known and the state vector is continuously known then the reaction rates can be estimated independently at any time point by simple rearranging of equation (5.1).

$$K^{-1}\left(\left.\frac{d\xi(t)}{dt}\right|_{t=\tau}+D(t)\xi(t)-u(\tau)\right)=r(\tau)=f(\xi(\tau),v(\tau)) \qquad (5.4)$$

Where the derivative is evaluated numerically using a finite difference approximation.

$$r(t)=K^{-1}\left(\frac{\xi(t+\Delta t)-\xi(t)}{\Delta t}+\frac{D(t)\xi(t)+D(t)\xi(t)-u(t)}{2}+\frac{D(t+\Delta t)\xi(t+\Delta t)+D(t+\Delta t)\xi(t+\Delta t)-u(t+\Delta t)}{2}\right) \qquad (5.5)$$

However, continuous online measurements of the concentrations are rarely available, so in practice each data series must be interpolated with some smoothing function $\hat{\xi}^*(t)\approx\xi(t)$ such as cubic smoothing splines Flannery et al(1999). In this case a spline was not used. Instead a SVM with time as the single independent variable was chosen as an interpolation function for reasons of code reuse.

By sampling the values and first derivative of this continuous function $\hat{\xi}^{\bullet}(t)$ a series of t pairs[27] relating reaction rates and concentrations at various time points across all training batches can be produced.

$$\begin{aligned} &\textit{Inputs} \quad x = \left[\hat{\xi}^{\bullet}(t_0), \hat{\xi}^{\bullet}(t_0 + \Delta t) \ldots \hat{\xi}^{\bullet}(t_f)\right] \\ &\textit{Outputs} \; y = \left[\hat{r}^{\bullet}(t_0), \hat{r}^{\bullet}(t_0 + \Delta t) \ldots \hat{r}^{\bullet}(t_f)\right] \\ &\text{where} \\ &\hat{\xi}^{\bullet}(\tau) \in \Re^{N_x} \\ &\hat{r}^{\bullet}(\tau) \in \Re^{N_r} \end{aligned} \tag{5.6}$$

The problem of modelling kinetics is then simply to find some non-linear function $r(\tau) = f(\xi(\tau), v(\tau))$ which predicts the interpolated reaction rate as a function of the corresponding full state vector at each time point. If this function $f(\cdot)$ is found correctly then it should be true that $\hat{r}^{\bullet}(\tau) = f(\hat{\xi}^{\bullet}(\tau), v(\tau))$.

5.2. Support vector machines.

Traditional approaches to training neural networks suffer from a tendency to over fit training data at the expense of generalisation. This is a consequence of the optimisation methods used for parameter selection and the statistical measures used to select the 'best' model architecture (number of neurons or basis functions).

Support vector machines (SVM) are a relatively new technique designed for classification and regression problems of high dimensionality and small sample sizes. Rather than minimising the error on training data SVMs employ statistical learning theory to minimise the upper bound on the generalisation error (Cortes and Vapnik(1995)). The SVM technique have the following advantages over traditional approaches to training neural networks.

- When a SVM is trained on data there is a single unique solution, which is guaranteed to be found Schölkopf et al(1999). This is in contrast to existing algorithms for both multilayer perceptrons and radial basis networks where training algorithms may become stuck in local optimum.

---

[27] Since the interpolated signal $\hat{\xi}^{\bullet}(\tau)$ is continuous it can be sampled at any point and not just when the state has been measured and $\xi^{\bullet}_{t-\tau}$ is known.

- The number of basis functions defining the SVM and hence the complexity of the SVM is determined automatically by a few data points known as *'support vectors'*. In practice only two hyper parameters need to be determined from validation data in order to set the architecture and capacity of the network(Chang and Lin(2001)).

- SVM's do not suffer significantly from *'the curse of dimensionality'* since the complexity of the function is independent of the dimensionality of the input or feature space. But rather is determined by the number of support vectors Sanchez(2003).

The technique has been successfully applied in many diverse areas such as time series prediction (Muller et al(1997)) and protein function classification (Cai et al(2003)). Nandi et al(2004) applied support vector regression to modelling and optimisation of a benzene isopropylation catalytic process. At the time of publication there was no application of the technique to bioprocess modelling. The theoretical advantages and reported success in other applications suggest that SVMs may have the capacity to improve hybrid modelling of bioprocesses.

5.2.1. Theory of Support vector machines.

Non-linear regression can be seen as linear regression in non-linear feature space (Figure 38). The input vectors x are mapped onto a high-dimensional feature space z, using some non-linear mapping $\Phi(x)$. A non-linear function fitting the data is then found as a weighed sum of the feature space vectors. The radial basis networks introduced in chapter 2 are a classical example of this. The achievement of the support vector machine formulation is that the function is only *implicitly* found in this high dimensional space. The weights of the mapping shown in Figure 38 are not computed, rather the mapping is specified in terms of the dot products in feature space of a small number of data points (known as support vectors).

While radial basis function networks employ some method such as clustering to determine the position and number of basis functions in SVMs this selection is implicit, with each of the support vectors contributing one local Gaussian function, centred at that data point. (Vapnik, 1995)

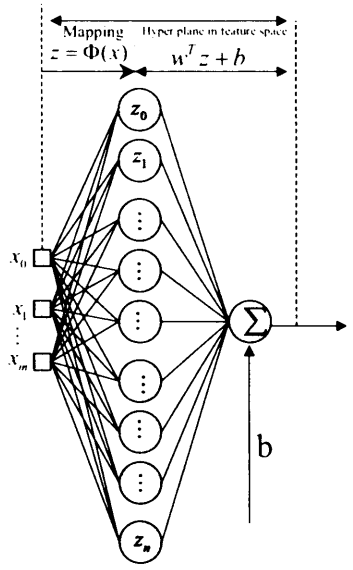


Figure 38. Non linear regression as generalised linear regression.

Consider a generalised linear regression function:

$$f(x) = w^T \cdot \Phi(x) + b. \tag{5.7}$$

Where w is a weight vector, Φ is the function mapping the input data in the feature space and b is the bias. Employing statistical learning theory (V Vapnik(1995)) it can be shown that w is optimally calculated by minimising the 'regularised risk functional':-

$$R_{reg}[f] = C\sum_{i=0}^{l-1} L(f(x^{(i)}) - y^{(i)}) + \frac{1}{2}\|w\|^2 \tag{5.8}$$

Where $\|w\|^2$ is the complexity term, C is a regularisation constant which determines the trade off between flatness and accuracy and $L(\cdot)$ is a loss function. In the SVM case this is usually Vapnik's ε-insensitive loss function shown in equation (5.9) and Figure 39 below .

$$L(f(x)-y) = \begin{cases} |f(x)-y| - \varepsilon. \text{ if } |f(x)-y| > \varepsilon \\ 0 \text{ otherwise} \end{cases} \tag{5.9}$$

If the predicted point lies inside a zone of acceptable error then the loss is zero, while if the predicted point is outside of the 'tube' the loss is proportional to the difference between the point and the edge of the zone.

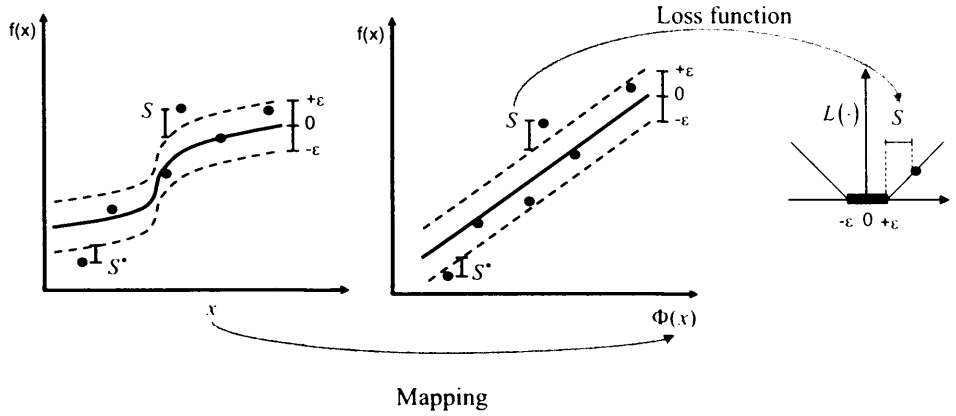


Figure 39. ε-insensitive loss function. Left most figure shows function in input space. Middle figure shows function in feature space. Right most figure shows the ε-insensitive loss function. Points inside the ε-insensitive 'tube' result in no loss. Points outside this 'tube' contribute linearly to the total loss.

Since the regression function is defined by a few points outside of the ε-insensitive zone known as support vectors, denoted S_i if above the 'tube' and S_i^* if below (see Figure 39), this loss function leads to a sparse solution in feature space in terms of these data points.

$$\begin{aligned} &\min_{w} \; C\sum_{i=0}^{\ell-1}\left(S_i^* + S_i\right) + \frac{1}{2}(w^T w) \\ &\text{subject to constraints} \\ &\qquad y^{(i)} - ((w^T z^{(i)}) + b) \le \varepsilon + S_i \\ &\qquad ((w^T z^{(i)}) + b) - y \le \varepsilon + S^*_i \\ &\qquad S_i, S_i^* \ge 0 \\ &\qquad i = 0, \ldots \ell - 1 \end{aligned} \tag{5.10}$$

The above constrained problem can be expressed using Lagrange multipliers $\alpha_i, \alpha_i^*, \gamma_i, \gamma_i^*$ as the following Lagrangian[28].

$$\begin{aligned} \Lambda_{\alpha,\alpha^*,\gamma,\gamma^*} \Bigg\{ &\frac{1}{2}\left(w^T w\right) + C\left(\sum_{i=0}^{\ell-1} S_i^* + \sum_{i=0}^{\ell-1} S_i\right) - \sum_{i=1}^{\ell} \alpha_i^* [y_i - ((w^T z^{(i)}) + b) + \varepsilon + S_i^*] \\ &- \sum_{i=0}^{\ell-1} \alpha_i [((w^T z^{(i)}) + b) - y^{(i)} + \varepsilon + S_i] - \sum_{i=0}^{\ell-1}\left(\gamma_i^* S_i^* + \gamma_i S_i\right) \Bigg\} \end{aligned} \tag{5.11}$$

[28] See Appendix D1 for an introduction to Lagrange multipliers.

V Vapnik(1995) shows by differentiation, with respect to the primal variables w_i, b, S_i, S_i^*, that the saddle points of (5.11) can be found and hence the minimum (5.10) is found by maximising;

$$W(\alpha,\alpha^*) = \sum_{i=0}^{\ell-1}(\alpha_i^* + \alpha_i) + \sum_{i=0}^{\ell-1} y_i(\alpha_i^* - \alpha_i) - \frac{1}{2}\sum_{i,j=0}^{\ell-1}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)\left(\left(z^{(i)}\right)^T \cdot z^{(j)}\right) \quad (5.12)$$

subject to constraints. $0 \le \alpha_i, \alpha_i^* \le C, i = 0,\ldots,\ell-1$ *and* $\sum_{i=0}^{\ell-1}\alpha_i = \sum_{i=0}^{\ell-1}\alpha_i^*$.

Note, that at least one of α_i, α_i^* must be zero since a data point cannot simultaneously be above and below the tube. Additionally, note that by substituting a kernel function $K(x^{(i)}, x) = \Phi(x^{(i)}) \cdot \Phi(x)$, for the function $\Phi(x)$ in the feature space the entire problem can be solved in the input space x rather than the feature space z. i.e:-

$$W(\alpha,\alpha^*) = \sum_{i=0}^{\ell-1}(\alpha_i^* + \alpha_i) + \sum_{i=0}^{\ell-1} y_i(\alpha_i^* - \alpha_i) - \frac{1}{2}\sum_{i,j=0}^{\ell-1}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)K(x^{(i)} \cdot x^{(j)}) \quad (5.13)$$

Table 4 List of Kernel functions taken from Sanchez, D. V.(2003).

Kernel function	Used in
$\tanh(\vec{x} \cdot \vec{y} - \theta)$	Multilayer perceptron (MLP)
$\exp(-\|\vec{x} - \vec{y}\|^2)$	Gaussian RBF Network
$(\|\vec{x} - \vec{y}\|^2 \pm c^2)^{-1/2}$	Direct/inverse multiquadric
$(1 - \vec{x} \cdot \vec{y})^d$	Polynomial of degree d
$\frac{\sin(d + 1/2)(x - y)}{\sin(x - y)/2}$	Trigonometric polynomial of degree d
$\|\vec{x} - \vec{y}\|^{2n+1}$	Thin plate
$\|\vec{x} - \vec{y}\|^{2n} \ln(\|\vec{x} - \vec{y}\|)$	Splines
$B_{2n+1}(x - y)$	B-Splines
$\frac{\sigma}{\pi}\operatorname{sinc}\left[\frac{\sigma}{\pi}(x - y)\right]$	Band-limited Paley Wiener space

Various kernel functions exist as shown in Table 4. In our case the Gaussian kernel was chosen since it provides a smoothly varying feature space capable of universal approximation.

$$K(x^{(i)}, x^{(i)}) = \exp\left(\frac{\left\|x^{(i)} - x^{(j)}\right\|^2}{-2\sigma_k^2}\right) \quad (5.14)$$

The model prediction at a new point x is calculated in terms of the kernel function as:-

$$f(x) = \sum_{i=0}^{m-1}(\alpha_i^* - \alpha_i)K(x^{(i)} \cdot x) + b \quad (5.15)$$

Figure 40 shows the architecture of a support vector regression machine. Conceptually the support vectors $x^{(0)} \cdots x^{(m-1)}$ and input vector $x^{(new)}$ are mapped onto non-linear feature space. The dot product $\Phi(x^{(i)}) \cdot \Phi(x^{(new)})$ is then computed between the mapped vectors thus characterising how similar the new point is to each of the support vectors. In practice the use of a kernel function $K(x^{(i)}, x)$ computes these two layers in a single step. The output is then obtained by a weighed sum of the resulting dot products with the weights having been obtained by maximising equation (5.13).

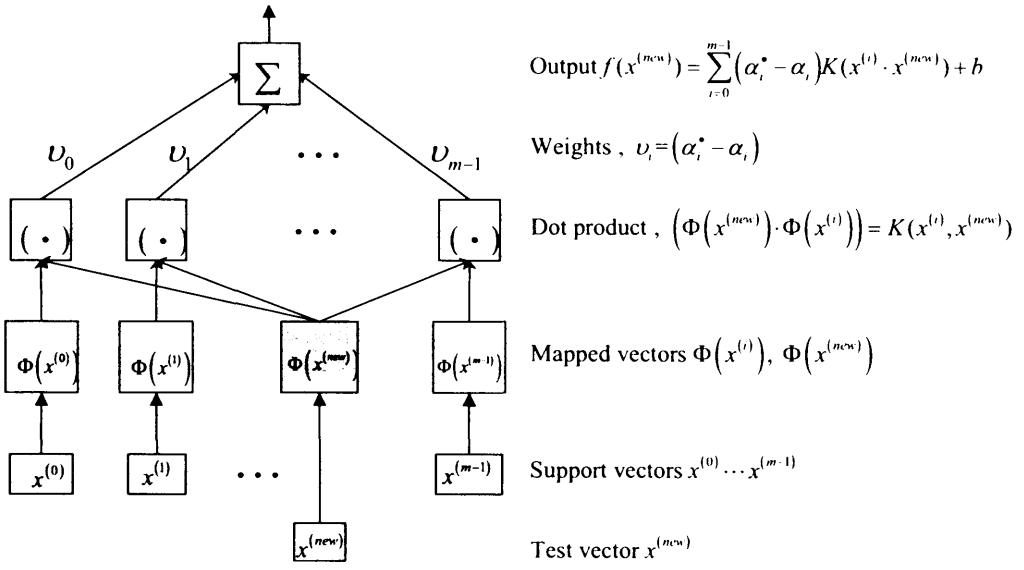


Figure 40. Support Vector Machine architecture. Adapted from Schölkopf, B. et al(1999)

This leads to a learning machine shown in Figure 40 with the following useful properties, (Schölkopf, B. et al(1999)):

1. The problem is convex with a unique global solution.
2. The result is a regularised solution avoiding over training.
3. Computation is efficient due to the usage of inner products and since only a few data points, known as support vectors, characterise the regression function.

4. The complexity of the function is independent of the dimension of either the input space or the feature space; instead it depends on the number of support vectors.

5.2.2. Implementation details. Training.

The C++ class library 'LibSVM', (Chang, C. C. et al(2001)), was used to train the support vector machines using the 'sequential minimal optimisation algorithm', (Platt(1999)). Data was supplied to the training process as a series of ℓ input-output pairs:-

$$\text{Input } x^{(i)} \in \Re^{N_x} \text{ output } y^{(i)} \in \Re$$

where the x data is the interpolated state vector and the y data the relevant reaction rate. The data was normalised to the range [0,1] to prevent scale differences from biasing training.

$$x_i = \frac{\hat{\xi}^*(i\Delta t) - \min_{\tau=0\ldots t_f} \hat{\xi}^*(\tau)}{\left(\max_{\tau=0\ldots t_f} \hat{\xi}^*(\tau) - \min_{\tau=0\ldots t_f} \hat{\xi}^*(\tau)\right)}, y_i = \frac{\hat{r}^*(i\Delta t) - \min_{\tau=0\ldots t_f} \hat{r}^*(\tau)}{\left(\max_{\tau=0\ldots t_f} \hat{r}^*(\tau) - \min_{\tau=0\ldots t_f} \hat{r}^*(\tau)\right)} \tag{5.16}$$

If predictions using the same units as the original data are the desired output then the output of the trained support vector needs to be rescaled:-

$$r = \min_{\tau=0\ldots t_f} \hat{r}^*(\tau) + \left(\max_{\tau=0\ldots t_f} \hat{r}^*(\tau) - \min_{\tau=0\ldots t_f} \hat{r}^*(\tau)\right) y \tag{5.17}$$

The overall model therefore has the following form:

$$\frac{d\xi}{dt} = K\left(\delta(\xi) r(\xi, v)\right) - D\xi + u$$

where (5.18)

$$r_i(\xi, v) = \min_{\tau=0\ldots t_f} \hat{r}^*(\tau) + \left(\max_{\tau=0\ldots t_f} \hat{r}^*(\tau) - \min_{\tau=0\ldots t_f} \hat{r}^*(\tau)\right) f_{SVMi}(\xi, v)$$

with each support vector machine being trained to predict the scaled reaction rate $y \in \Re$ as a function of the scaled states $x \in \Re^n$.

Determining the values of the hyper-parameters.

For each SVM model two hyper-parameters need to be determined these are:

- The regularisation parameter C, which determines the trade off between minimising training errors and minimising model complexity.
- The kernel parameter, which implicitly defines the high dimensional feature space used, in the case of the Gaussian kernel this is the width parameter σ_k.

The influence of these parameters on the complexity of the regression function is illustrated on a one-dimensional problem shown in Figure 41 below.

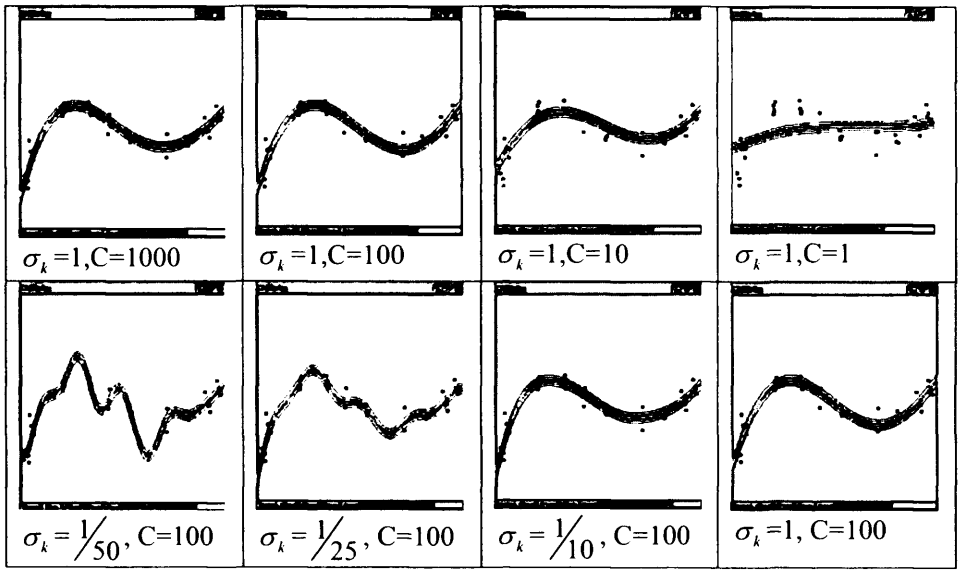


Figure 41. The influence of hyper-parameters on a one dimensional regression function. On the top row σ_k is held constant and C varied. On the bottom row C is held constant and σ_k is varied. C

A small value of the cost parameter C defines a highly regularised function, a small value for the Gaussian kernel width σ_k defines a highly complex feature space.

A cross validation strategy is used to set the values of the hyper-parameters. This strategy is explained in detail in the appendix D2. Briefly the method is to divide the training data into 'training' and 'cross validation' batches and then use the Nelder and Mead simplex method, (Flannery, B. P. et al(1999)), to find the hyper-parameters $\sigma \in \Re^{N_v}, C \in \Re^{N_v}$ which minimise the error the whole hybrid model has on the validation data. For each validation batch this is given as;

$$\min_{\sigma,C}\left(\sum_{j=0}^{N_\xi-1}\left(\frac{1}{R(j)}\sum_{i=0}^{N_{max}}\left(\xi_j(t_s^{\langle i\rangle})-\xi_j^P(t_s^{\langle i\rangle})\right)^2\right)\right)$$

where (5.19)

$$R(j)=Max\left(\xi_j(t_s)\right)-Min\left(\xi_j(t_s)\right)$$

$\xi^P(\tau)$ refers to the prediction of support vector machines trained on the training data with the current hyper-parameter values.

5.3. Simulated example.

5.3.1. Details of Test system.

A model of a fed-batch murine hybridoma cell culture producing monoclonal antibodies by Jang and Barford(2000), was chosen to simulate experimental data. Full details of this model can be found in the appendix B3.

12 batches of data were obtained by running the above model, either in batch mode or in fed batch mode, with different feeding rates (F_{in}, F_{out}), and media concentrations. Data batches consisted of concentration-time profiles for the state vector[29], sampled every 8 hours from $t=0$ to 120 hours.

Of these batches 5 were randomly selected for training, 2 for validation and 5 for testing. Normally distributed noise $\sigma=0.15$ was added to the training and validation batches in order to resemble data obtained from a real cell culture process and to examine how robust the methodology was to experimental error.

[29] For this system the state vector consists of C_{X_v} viable cells, C_{X_d} non-viable cells, C_{AB} antibody, C_{GLC} glucose, C_{GLN} glutamine, C_{AMM} ammonia and C_{LAC} lactate.

5.3.2. Hybrid Model development.

A reaction network where lactose and glucose are carbon sources and glutamine and ammonia are the nitrogen sources was used to formulate a hybrid model of the following form[30].

$$\frac{d}{dx}\begin{pmatrix} C_{X_v} \\ C_{X_d} \\ C_{AB} \\ C_{GLC} \\ C_{GLN} \\ C_{LAC} \\ C_{AMM} \end{pmatrix} = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -y_{GLC/X} & 0 & -y_{GLC/AB} \\ -y_{GLN/X} & 0 & -y_{GLN/AB} \\ Y_{lac/glc}\, y_{GLC/X} & 0 & Y_{lac/glc}\, y_{GLC/AB} \\ Y_{amm/gln}\, y_{GLN/X} & 0 & Y_{amm/gln}\, y_{GLN/AB} \end{pmatrix} \begin{pmatrix} \mu \\ \mu_d \\ Q_{ab} \end{pmatrix} - D(t) \begin{pmatrix} C_{X_v} \\ C_{X_d} \\ C_{AB} \\ C_{GLC} \\ C_{GLN} \\ C_{LAC} \\ C_{AMM} \end{pmatrix} + F_{in} \begin{pmatrix} 0 \\ 0 \\ 0 \\ C^{in}_{GLC} \\ C^{in}_{GLN} \\ 0 \\ 0 \end{pmatrix} \tag{5.20}$$

where the yield factors are those from the true model and the growth rate, death rate, and antibody production rate are determined by SVMs;-

$$\begin{aligned} \mu &= X_v \times \delta_0(\xi) \underset{SVM}{f_0}(\xi, v) \\ \mu_d &= X_v \times \delta_1(\xi) \underset{SVM}{f_1}(\xi, v) \\ Q_{ab} &= \delta_2(\xi) \underset{SVM}{f_2}(\xi, v) \end{aligned} \tag{5.21}$$

As detailed in section 5.2 the series of the training batches were smoothed and interpolated to provide concentrations as a function of time. The known effect of dilution was then subtracted and the remaining rates projected through the mass balance to give the key reaction rates.

---

[30] For conciseness rescaling $r_i(\xi, v) = \min_{\tau=0\ldots t_f} \hat{r}^*(\tau) + \left(\max_{\tau=0\ldots t_f} \hat{r}^*(\tau) - \min_{\tau=0\ldots t_f} \hat{r}^*(\tau) \right) \underset{SVM_i}{f}(\xi, v)$ has been dropped from equation (5.21).

In this way 3 sets of input/output pairs for training were established:

The first SVM was trained to predict growth rate as a function of the state.

$$\underset{SVM}{f_0}(\xi,v)$$

Trained on (5.22)

$$Inputs \quad x=\left[\hat{\xi}^{\bullet}(t_0),\hat{\xi}^{\bullet}(t_0+\Delta t)...\hat{\xi}^{\bullet}(t_f)\right]$$

$$Outputs\ y=\left[\hat{\mu}^{\bullet}(t_0),\hat{\mu}^{\bullet}(t_0+\Delta t)...\hat{\mu}^{\bullet}(t_f)\right]$$

The second SVM was trained to predict death rate as a function of the state.

$$\underset{SVM}{f_1}(\xi,v)$$

Trained on (5.23)

$$Inputs \quad x=\left[\hat{\xi}^{\bullet}(t_0),\hat{\xi}^{\bullet}(t_0+\Delta t)...\hat{\xi}^{\bullet}(t_f)\right]$$

$$Outputs\ y=\left[\hat{\mu}_d^{\bullet}(t_0),\hat{\mu}_d^{\bullet}(t_0+\Delta t)...\hat{\mu}_d^{\bullet}(t_f)\right]$$

The third SVM was trained to predict antibody production rate as a function of the state.

$$\underset{SVM}{f_2}(\xi,v)$$

Trained on (5.24)

$$Inputs \quad x=\left[\hat{\xi}^{\bullet}(t_0),\hat{\xi}^{\bullet}(t_0+\Delta t)...\hat{\xi}^{\bullet}(t_f)\right]$$

$$Outputs\ y=\left[\hat{Q}_{ab}^{\bullet}(t_0),\hat{Q}_{ab}^{\bullet}(t_0+\Delta t)...\hat{Q}_{ab}^{\bullet}(t_f)\right]$$

$$where\ \Delta t=\frac{(t_f-t_0)}{\ell} \tag{5.25}$$

5.3.3. Results of training on hybridoma simulation.

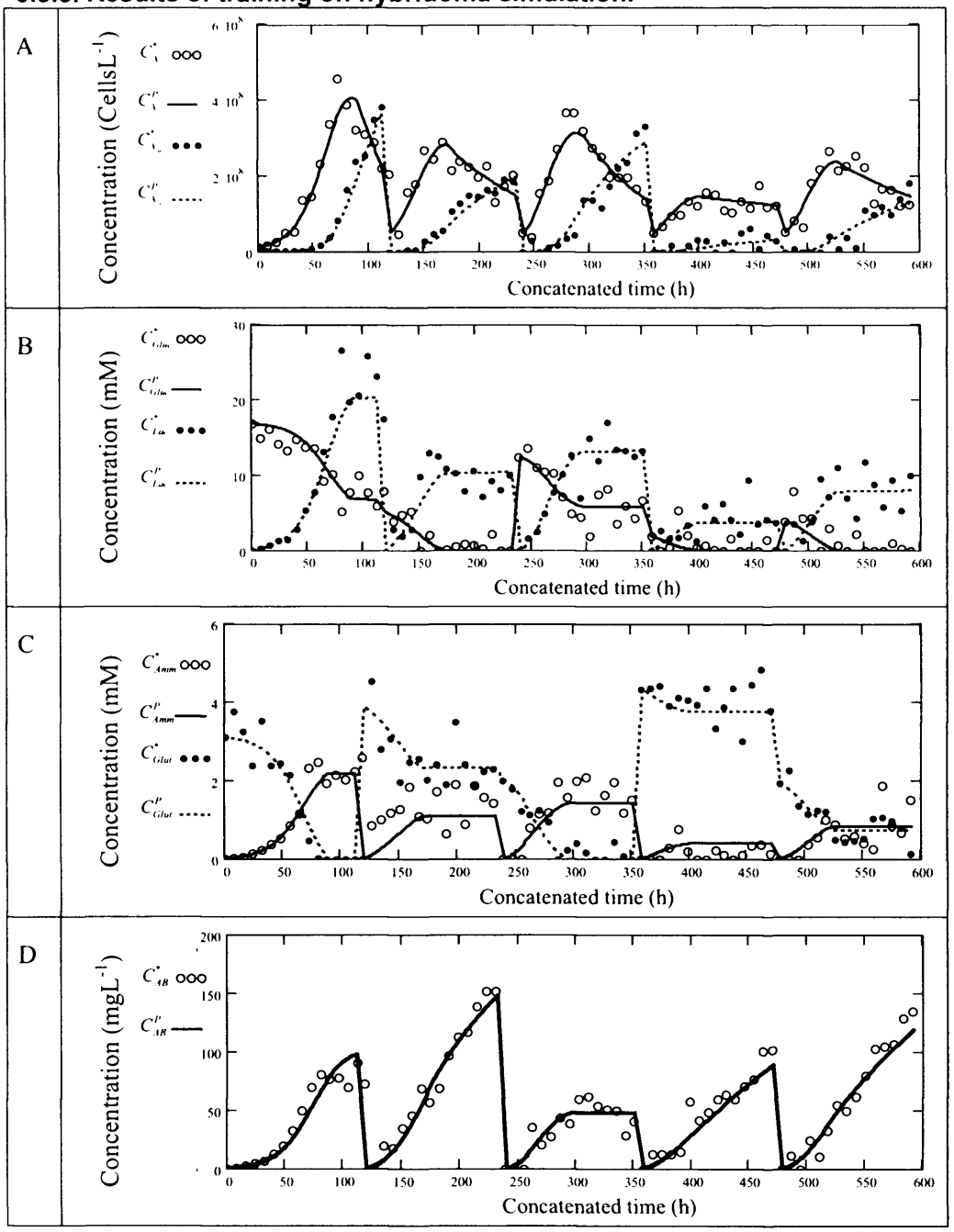


Figure 42. Model prediction vs. training/validation data generated by model of Jang, J. D. et al(2000). Points are the sampled concentrations with added noise used for training. Lines are the prediction of the trained model. From top to bottom; graph A shows the viable and non viable cell concentrations in cells/L, graph B shows glucose and lactate concentrations in mM, graph C shows ammonia and glutamate concentrations(mM) and graph D shows Antibody concentration in mg/L. Time is concatenated to allow all 5 training/validation batches to be displayed on the same graph.

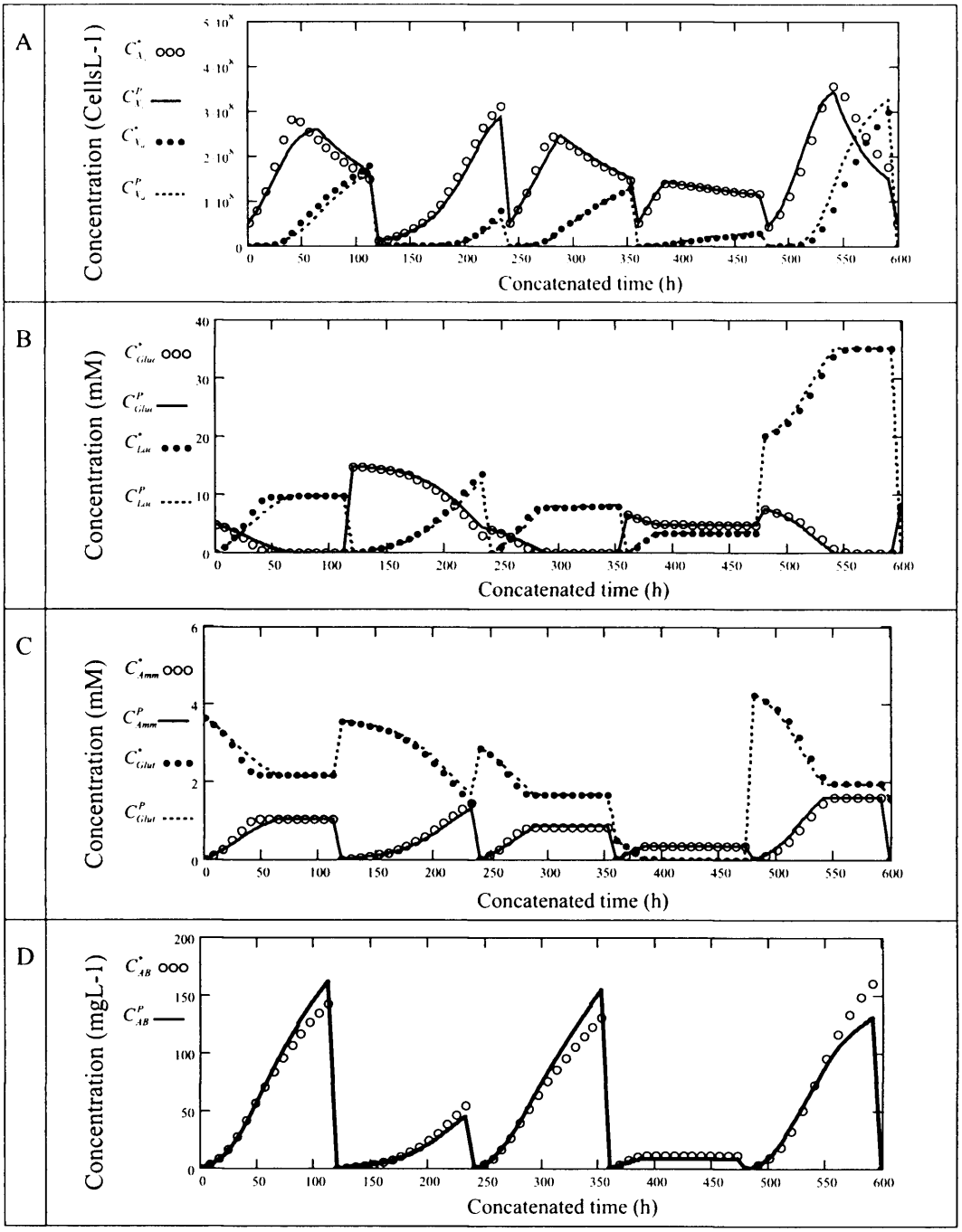


Figure 43. Model prediction vs. testing data generated by model of Jang, J. D. et al(2000).. Points are the true concentrations sampled at regular intervals. Lines are the prediction of the trained model. From top to bottom; graph A shows the viable and non viable cell concentrations in cells/L, graph B shows glucose and lactate concentrations in mM, graph C shows ammonia and glutamate concentrations(mM) and graph D shows Antibody concentration in mg/L. Time is concatenated to allow all 5 testing batches to be displayed on the same graph.

5.3.4 Discussion.

Figure 42 shows an example of the fit to the training set achieved using the SVM methodology. Plot *'a'* shows viable and non-viable biomass. Plot *'b'* shows glucose and lactate concentrations. Plot *'c'* shows glutamine and ammonia. Plot *'d'* shows antibody concentration. Figure 43 shows the model prediction on unseen testing data. It is clear from these predictions on unseen conditions that the modelling methodology has managed to characterise the general dynamics of the system from noisy training data. However the model is not perfect and in common with other recursive systems, errors in the model or the initial conditions accumulate as the system is integrated. This causes the poor prediction of the final biomass and antibody concentrations as seen in Figure 43 *'a'* and *'d'* respectively. However the general accuracy of the prediction is good in the context of the large noise level with the predictions on both training and testing batches being within the range of measurement error.

5.4. Evaluation of potential suitability for model based optimisation.

5.4.1 Details of evaluation.

Plots of the model prediction against testing data are indicative of model accuracy. However, the purpose of modelling within an engineering context is not usually to produce graphs or indeed merely to predict accurately for its own sake. Rather model predictions are used to make decisions.

Consider the use of the hybrid modelling methodology to optimise the concentrations of Glycerol and Glutamine so as to maximise the final antibody concentration. A hybrid-SVM model was trained on 5 batches of noisy data generated as in section 5.4 from the model of Jang, J. D. et al(2000) with randomly chosen initial conditions as follows:-

Table 5. Initial conditions for simulated experiments.

Batch Number	Xv Cells/L	GLN g/L	GLC g/L	LAC g/L
1	4.65×10^7	1.93	7.55	17.56
2	4.95×10^7	3.32	17.07	3.96
3	5.21×10^7	2.31	4.42	0.78
4	5.43×10^7	0.60	17.30	7.39
5	4.85×10^7	0.78	12.54	19.32

The resulting model was then used to generate a surface plot Figure 44a showing the effect of initial glycerol and glutamine concentration on the final antibody concentration. For comparison the correct surface is shown in Figure 44c below and a Gaussian process[31] response surface modelling final titre $C_{AB} = \underset{gaus}{f}\left(C_{GLN}, C_{GLC}\right)$ is shown in Figure 44d.

5.4.2. Results.

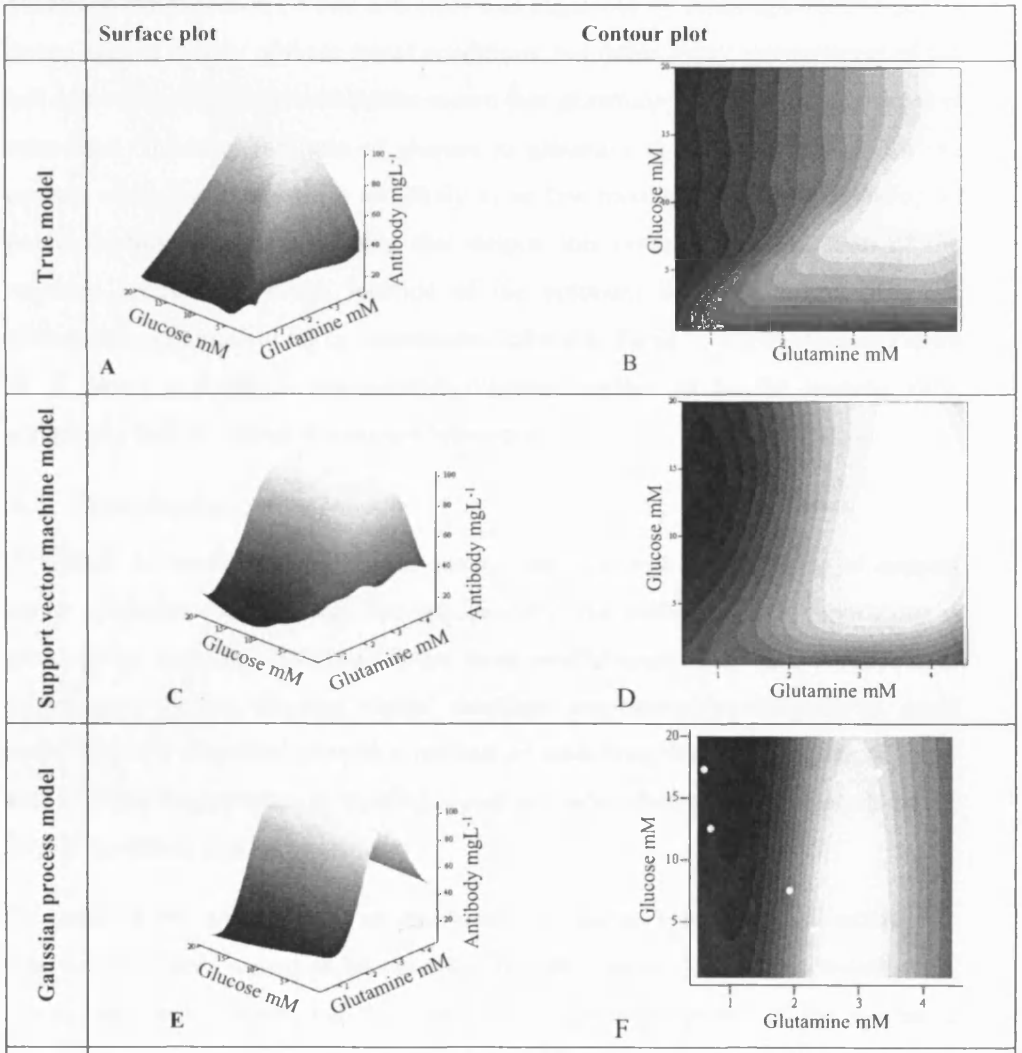


Figure 44 Relationship between final antibody titre (mg/L) and initial glucose and glutamine concentrations (mM). Figures A,C,E show the response surface implied by each model. Figures B,D,F show a contour plots of the same data with lighter shades indicating higher antibody concentration.

[31] Appendix E1 gives MathCAD code for Gaussian process modelling. Here it is sufficient to think of the Gaussian process model as a simple response surface.

The plots of the true system behaviour and inferred model are similar but with 2 discrepancies. The location of the optimum of the inferred model is not exactly that of the true model and the surface predicted is more rounded with the raised ridge leading from coordinates $(C_{GLN}=3.5\ mM, C_{GLC}=10\ mM)$ to $(C_{GLN}=1\ mM, C_{GLC}=2.5\ mM)$ not being fully captured. This may be due to the fact that for most of the training batches the initial media concentration of glucose was significantly more than the initial glutamine concentration (White dots shown in Figure 44 d). Although training points do not consist merely of these initial conditions, but rather every measurement of the cell culture, mass balance constraints ensure that glutamine is consumed as glucose is consumed. Therefore the ratio of glucose to glutamine is likely to be high for the entirety of the batch and there are likely to be few training points corresponding to points on this ridge. It is notable that despite this issue the general form of the response surface and rough location of the optimum has been found from the information contained in the 5 experiments defined in Table 5. By contrast as Figure 44 d shows a standard non-dynamic response surface fit to the training data, completely fails to capture the system behaviour.

5.5. Conclusion.

A method for modelling the kinetic component of hybrid models using of support vector machines was proposed. The application of the method to hybrid modelling is novel in the sense that SVMs have not been used to model the reaction kinetics of bioprocesses before. Support vector machines are explicitly designed to avoid overfitting and therefore provide a method of modelling which is simpler to apply than traditional approaches to training neural networks. This potentially speeds up the hybrid modelling process.

The method was demonstrated on data from a simulated hybridoma cell culture. On this system it was shown to be capable of producing accurate fits to training and testing data and of capturing the general form of the response of the system to changes in initial conditions. However one should be cautious in drawing conclusions about the general performance of a modelling technique from the performance of that technique on a single simulated system. Rather it is necessary to demonstrate the technique on real systems and to compare the technique with others on multiple systems before making any general claims.

6. Application of the SVM methodology to experimental data

In the previous section a hybrid modelling methodology based around support vector machines was developed and demonstrated on a simulated system. In this section the SVM methodology is used to produce data driven models of 3 real experimental systems

- A VPM8 Murine hybridoma cell culture.
- A *Saccharopolyspora erythraea NRRL2338* (red variant wild type) shake flask cultivation.
- A 42L *Streptomyces clavulingerus* batch cultivation.

The last system was used by Roubos(2002) as a test system for his hybrid modelling work. The SVM methodology produces predictions of comparable accuracy to his published results indicating the potential of the technique on real data.

6.1 Demonstration 1: Murine hybridoma shake flask cultivation[32].

6.1.1 Culture details

VPM8 Murine hybridoma cells, (Jones 1985), were acquired from ECACC. VPM8 cells are a Murine hybridoma producing IgG1 directed against the light chain of ovine IgG. The cells were grown in 15ml culture volume of RPMI 1640 medium (Sigma, Poole UK) supplemented with 100mLs/L fetal bovine serum, 2mM L-Glutamine, 1mM Sodium Pyruvate (Sigma, Poole, UK). Culture of cells was carried out in 50ml vent cap shake flasks (Corning, Coventry UK) at 37 °C in a 5% CO_2 Incubator (RS Biotech, Ayrshire, UK) with agitation being provided by a 120 shaker (Heidolph, Schwabach, Germany) at 90rpm.

Analysis.

Throughout, the cultivation samples were taken at regular intervals in order to determine viable/non viable cell counts and concentrations of glucose, glutamine, glutamate, lactate, ammonia and antibody. Cell counts were performed using 100x magnification on an Olympus microscope. Viability was assessed using the trypan

[32] Cell culture work was done in collaboration with Sam Denby of the University College London Biochemical Engineering department.

blue (Sigma, Poole, UK) exclusion method. A high contrast improved Neubauer haemocytometer (Marienfeld) was used to count a minimum of 100 viable cells. Glucose, glutamine, glutamate and lactate concentrations were determined using a YSI 2700 SELECT™ Biochemistry Analyser(YSI, Ohio, USA). Ammonia was determined using a indophenol blue assay read on a saffire plate reader (Tecan, Reading,UK). Antibody concentration was determined using the Enzyme Linked Immunosorbent Assay (ELISA) method.

6.1.2 Modelling.

Five training conditions and two testing conditions consisting of different initial glucose and glutamine concentrations were selected as shown in Figure 45. Two replicates were run at each condition by dividing the adjusted media between shake flasks with all 14 shake flasks being run simultaneously.

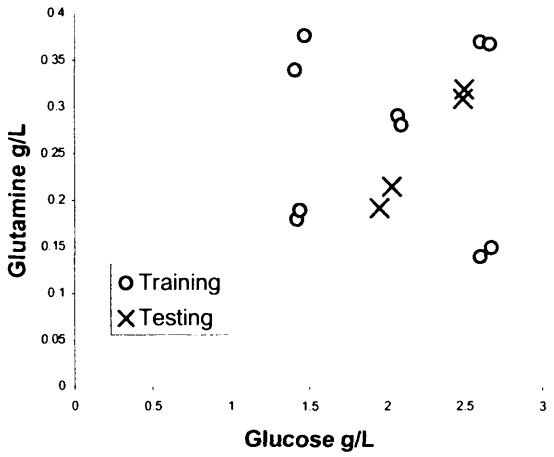


Figure 45. Plot of initial Glutamine and glucose concentrations for 7 Murine hybridoma cell cultures in duplicate.

The training data was scaled between zero and one and interpolated as in the previous section. The measurements for Ammonia, Glucose and Glutamate were too noisy for accurate interpolation and were instead recovered from the other components by mass balancing. Derivatives were then taken of the interpolated profiles of the other components to obtain $\frac{d\xi_i}{dt}$ at each time point and the SVM method developed in chapters 4 and 5 applied. There is no flow into or out of the shake flasks so the resulting model has following form:

$$\frac{d}{dx}\begin{pmatrix} X_{viable} \\ X_{nonviable} \\ C_{lactate} \\ C_{Glucose} \\ C_{Glutamine} \\ C_{Glutamate} \\ C_{Antibody} \\ C_{Ammonia} \end{pmatrix} = M \begin{pmatrix} f_0(\xi) \\ SVM \\ f_1(\xi) \\ SVM \\ f_2(\xi) \\ SVM \end{pmatrix} \tag{6.1}$$

Where the matrix M was determined by principle component analysis and the three support vector machines trained to predict the first, second, and third principle components given by $p(\xi) = M^T \frac{d\xi_i}{dt}$.

6.1.3 Results.

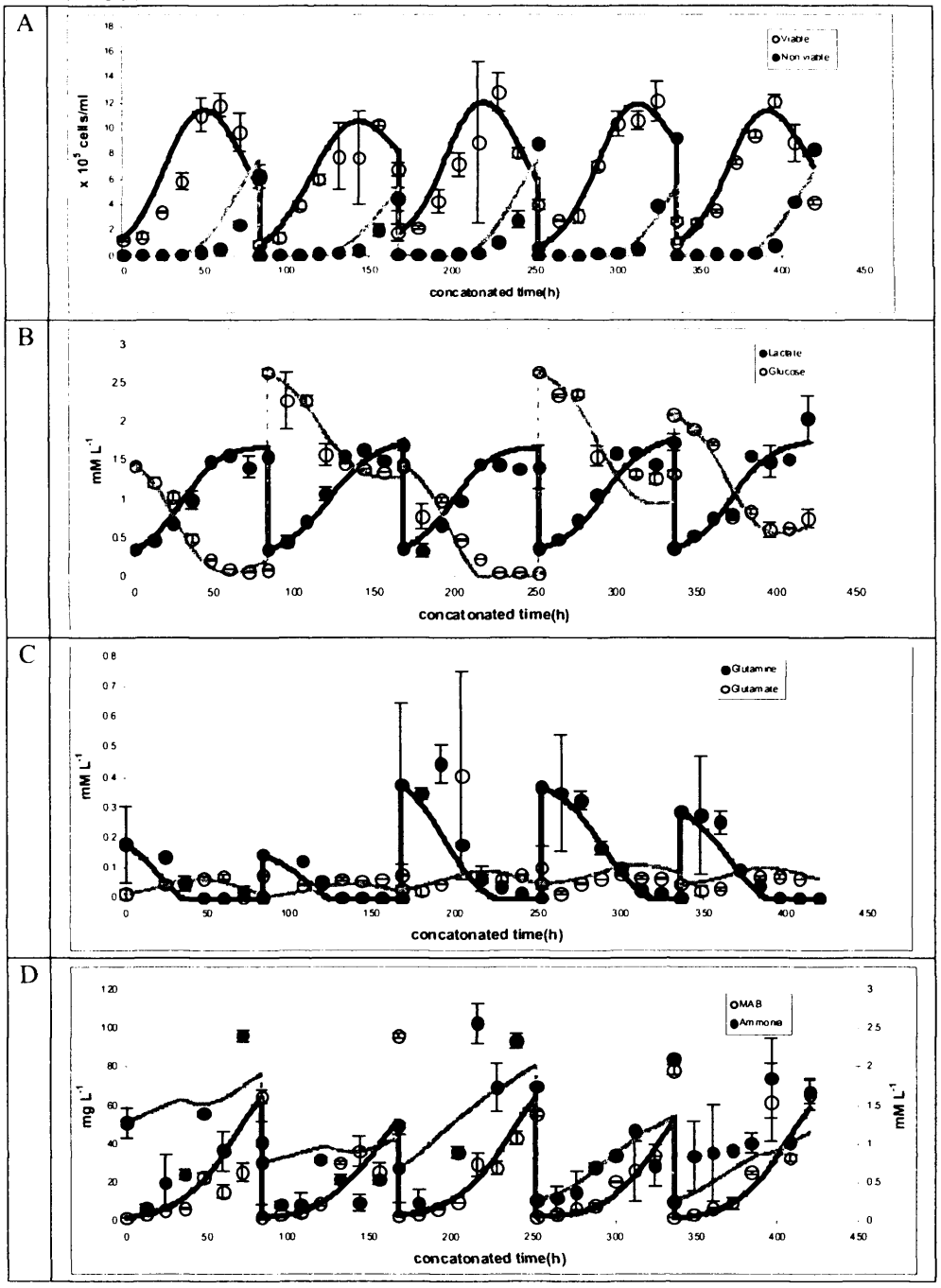


Figure 46. Fits to Murine hybridoma training data. A shows counts for viable and non viable cells. B shows lactate and glucose concentrations. C shows Glutamine and Glutamate concentrations. D shows Monoclonal antibody concentration in (mg/L) as the left most axis, and ammonia concentration(mM) as the right most axis. Lines show model predictions, error bars show maximum and minimum measurements from duplicate batches, points show mean of measured values.

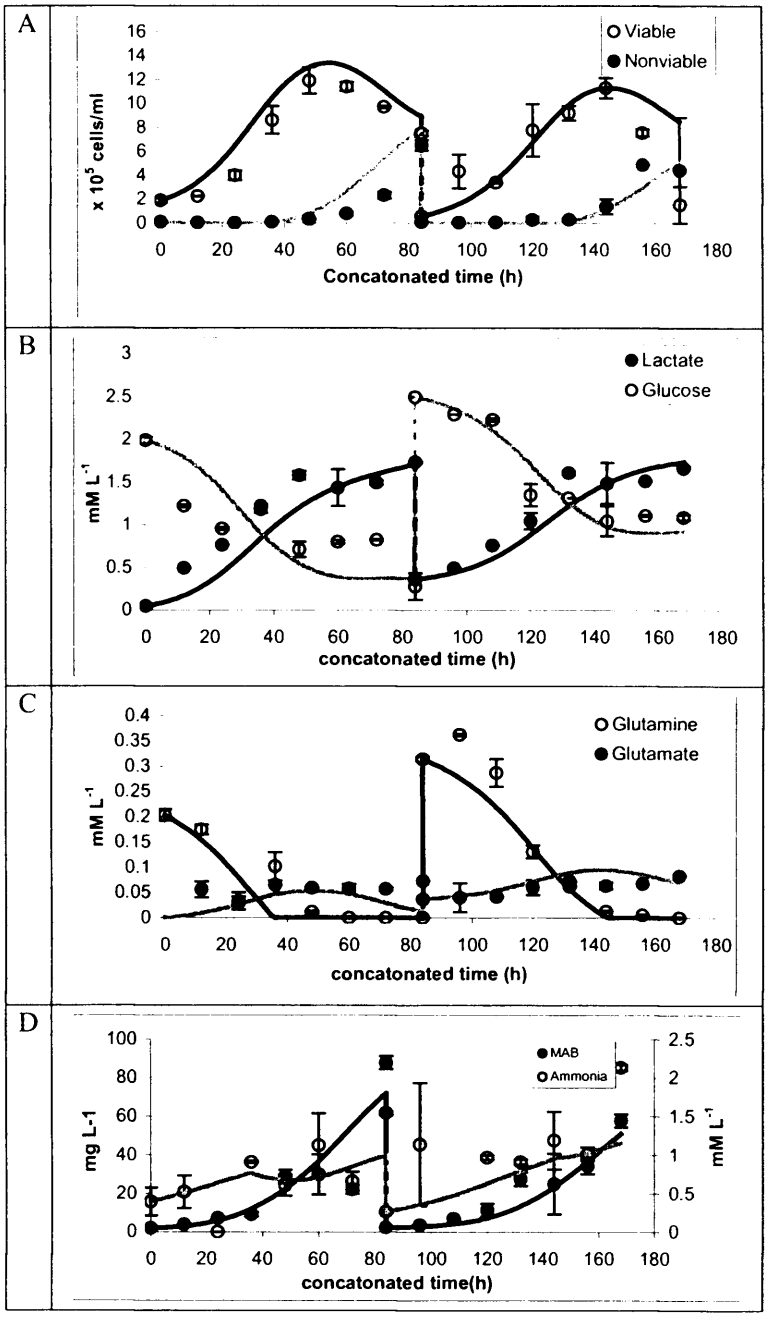


Figure 47. Fits to Murine hybridoma testing data. A shows counts for viable and non viable cells. B shows lactate and glucose concentrations. C shows Glutamine and Glutamate concentrations. D shows Monoclonal antibody concentration in (mg/L) as the left most axis, and ammonia concentration(mM) as the right most axis. Lines show model predictions, error bars show maximum and minimum measurements from duplicate batches, points show mean of measured values

6.1.4 Discussion.

The PCA-SVM model's fit to training data is shown in Figure 46 and the fit to unseen testing data in Figure 47. The x axis is culture time in hours with batches concatenated together to allow all data to be displayed on the same graph. Lines are model predictions. Points are the mean of measured values with error bars showing the difference between the two replicates at each condition.

The model has clearly been able to infer most of the system dynamics. However we have no real basis by which to declare a particular fit to be good or bad. The measurements are very noisy and the dynamics of the experimental system do not seem to vary much as a function of the initial conditions. Even if we were to compare the SVM methodology to another modelling method, because of the difference between replicates, it is not clear that a fit to the measurements would be a desirable basis for comparison. The result should be seen as a demonstration of the SVM methodology but it cannot realistically be used as a basis for judging the method or declaring its usefulness.

6.2. Demonstration 2: Saccharopolyspora erythraea *shake flask cultivation*[33].

6.2.1 Shake flask experiments.

Polyketide producing filamentous *Saccharopolyspora erythraea NRRL2338*, (red variant wild type), was grown in agitated 0.5 L shake flasks containing 50 mL of defined medium with glucose and nitrate as sole C-and N-source, respectively. The organism was grown in batch culture for 120 hours (at 28°C and 200 rpm on a rotary shaking incubator.

Analysis.

Throughout the cultivation samples were taken to measure dry cell weight (DCW), glucose and nitrate concentrations in the supernatant and the concentration of a red pigment that was used as a model product. Details can be found in Ushio(2003) and Hodgson et al(2004).

[33] Shake flask experiments and analysis were performed by Misty Ushio.

6.2.2 Modelling.

The data used in this study were obtained from five experiments in which the C/N ratio was changed by variation of the initial nitrate concentration in the range from 1.76 to 8.77 gL^{-1}.

Table 6. Initial conditions of *S. erythraea* shake flask experiments.
Initial red pigment was 0 g/L in all cases.

Batch Number		Glucose (in sol'n), g/L	Nitrate (in sol'n), g/L	DCW, g/L
batch 1	Training	33.21	1.76	1.37
batch 2	Testing	35.33	2.35	1.19
batch 3	Training	32.78	2.94	1.09
batch 4	Testing	30.05	4.22	1.54
batch 5	Training	29.70	8.77	1.84

The system was chosen since there is great variation in the metabolism as the bacterium shifts from carbon to nitrate limited growth, depending on the initial conditions. In particular the production of red pigment is growth dependent under carbon-limiting conditions, it is produced at the onset of the stationary phase under nitrogen-limited conditions. 3 batches of data were used for training and 2 unseen batches of data used for testing the resulting model.

The training data was scaled between zero and one and interpolated as in the previous section. PCA analysis suggested 2 degrees of freedom would be enough to characterise the system. Growth and red pigment production were chosen as the free rates and modelled using the SVM methodology. The yield factors for glucose and nitrate consumption were calculated by simple regression.

There is no flow into or out of the shake flask so the model thus has the following form:

$$\frac{d}{dx}\begin{pmatrix} C_{Glucose} \\ C_{nitrate} \\ X_{DCW} \\ C_{Redpigment} \end{pmatrix} = \begin{pmatrix} -3.4 & -0.853 \\ -17 & 0.58 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} f_0(\xi) \\ {}_{SVM} \\ f_1(\xi) \\ {}_{SVM} \end{pmatrix} \tag{6.2}$$

where $f_0(\xi)$ predicts the growth rate and $f_1(\xi)$ predicts the red pigment production rate.

6.2.3 Results.

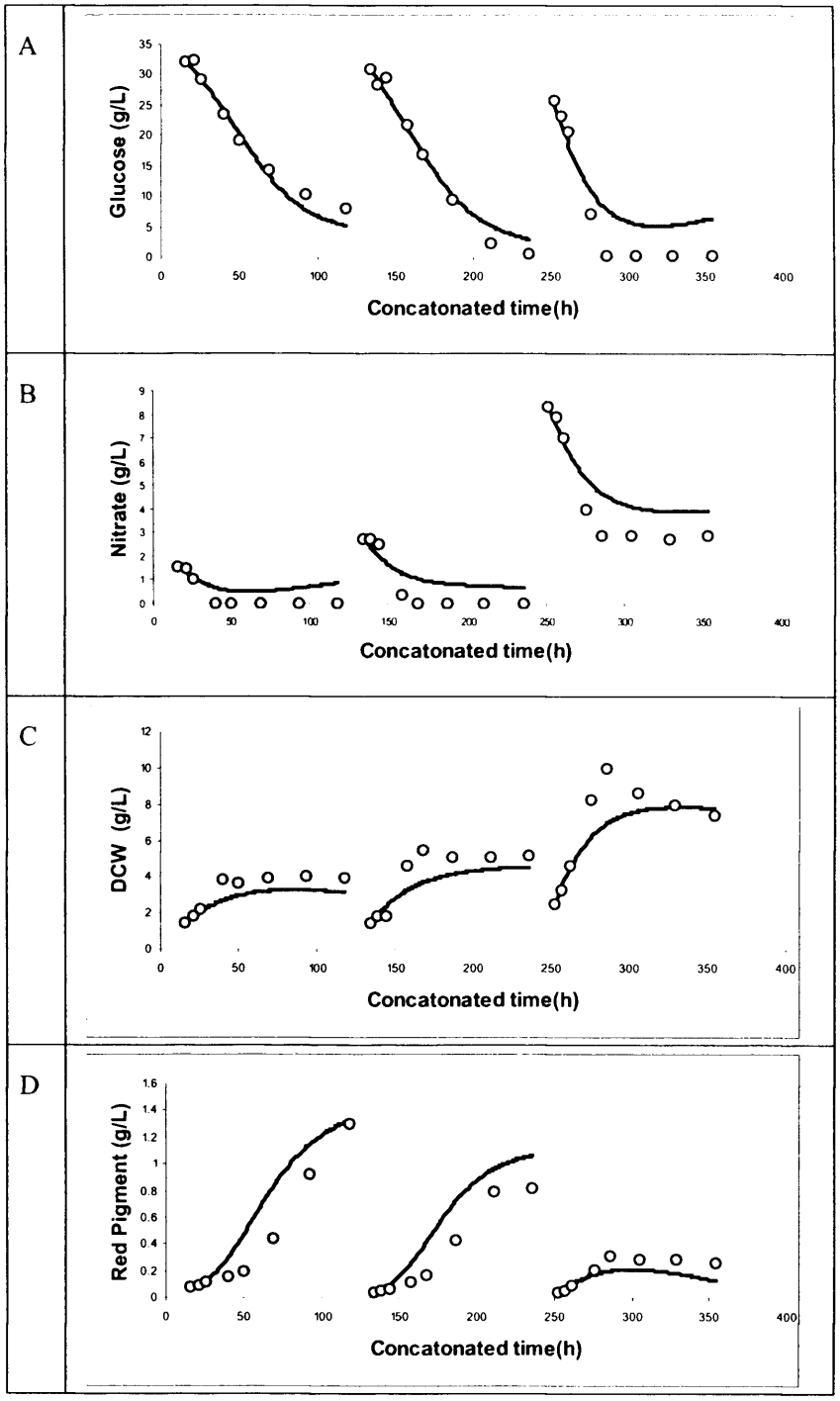


Figure 48. Fit to *S. erythraea* training data. Lines show model predictions, points show measured concentrations (grams per litre). A shows glucose concentration . B shows nitrate concentration . C shows dry cell weight (grams per litre). D shows red pigment concentration.

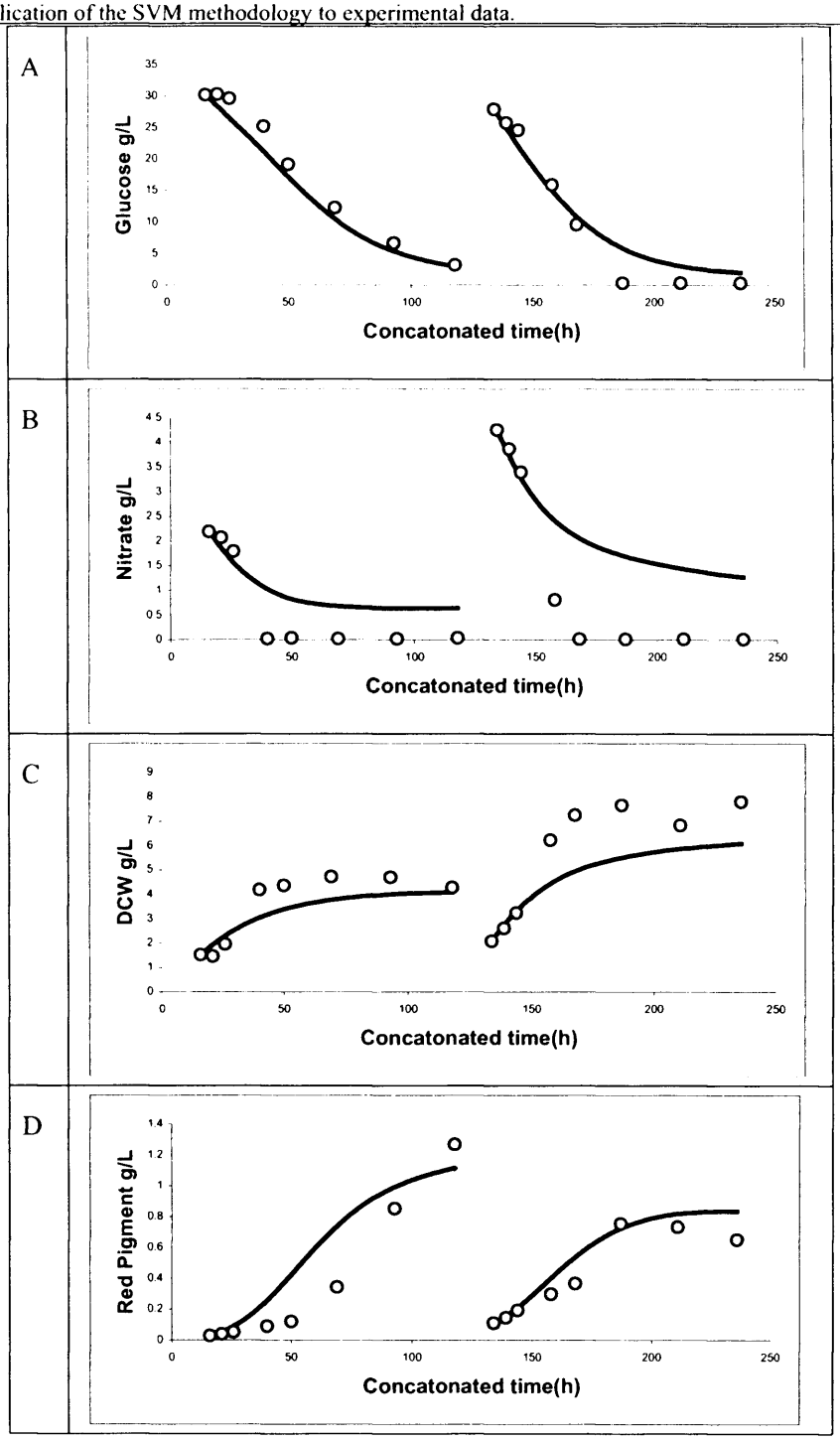


Figure 49. Fit to *S. erythraea* testing data. Lines show model predictions, points show measured concentrations (grams per litre). A shows glucose concentration . B shows nitrate concentration . C shows dry cell weight (grams per litre). D shows red pigment concentration.

6.2.4 Discussion.

The fits to training data are shown in Figure 48. and to testing data in Figure 49. It is immediately apparent that the SVM methodology has failed to capture some significant features of the system. In particular the dry cell weight peaks in training batches 3 and 5, (located at 170 hrs and 270 hrs in Figure 48c respectively), have been ignored when interpolating the data. This results in significant underestimation of the growth kinetics in general which leads to poor prediction of the biomass concentration and as Figure 49 B shows the model fails to predict the complete consumption of nitrate for both testing batches.

6.3. Demonstration 3: Streptomyces Clavuligerus batch process

6.3.1 Introduction:

Seven batches of *S. clavulingerus*[34] batch data produced at Delft University of Technology were modelled by Hans Roubos as part of Roubos, J. A.(2002). In this work a detailed metabolic network was used to determine the constraint matrix. The kinetics were modelled using three methods: NN; fuzzy logic and conventional mechanistic models.

This prior work means the *S. clavulingerus* system can be used as a benchmark for assessing the performance of the SVM methodology. The detailed metabolic modelling approach will not be repeated here. Rather the idea is to test whether almost equivalent accuracy can be obtained on real data with the much faster, but more naïve, PCA technique.

As well as the availability of results for comparison, the system is a relevant benchmark since *Streptomyces* are an industrially important species of filamentous bacteria. They produce two thirds of known antibiotics (Butler et al(2002)). A key advantage is that recombinant proteins are efficiently secreted into the extra-cellular medium and do not form biologically inactive inclusion bodies (Nakashima et al(2005)). An ability to model this system can be viewed as showing the industrial potential of hybrid modelling.

[34] *S. Clavuligerus* produces Clavulanic acid a B-lactamase inhibitor used in combination with B-lactam antibiotics.

6.3.2 Bioreactor experiments[35].

All batch experiments were performed by in a 42-L stirred vessel bioreactor. The whole vessel was placed on a balance to allow the broth weight to be monitored over time. Temperature was regulated at 30°C, pH at 7 by the addition of 4M H_2SO_4 or 4M NaOH and DOT controlled above 50% by stirrer speed and airflow with the initial stirrer speed and air flow being 200 rpm and 15L/min respectively.

The medium in all cases contained: $MgSO_4.7H_2O$ 0.8g/L, $FeSO_4.7H_2O$ 0.2 g/L, Basildon antifoam 0.2 g/L and trace element solution (1.6 g/L). The trace element solution contained H_2SO_4 (96%) 20.4 g/L, citrate.$1H_2O$ 50 g/L, $ZnSO_4.7H_2O$ 16.75 g/L, $CuSO_4.5H_2O$ 2.5 g/L, $MnCL_2.4H_2O$ 15 g/L

Analysis.

Off gas was analysed providing oxygen uptake rate (OUR) and carbon dioxide evolution rate (CER) online. The following measurements were taken regularly from broth samples and measured offline: biomass; glycerol; glutamate; ammonium; phosphate and clavulanic acid concentrations.

6.3.3 Modelling.

The carbon, nitrogen and phosphate sources were varied between each batch according to Table 7.

Table 7. Initial conditions of *S. clavulingerus* cultivations.

Batch Number		Glycerol $C_3O_3H_8$ Cmol/L	Glutamate $NaC_5H_8O_4N$ Cmol/L	Ammonia $(NH_4)_2SO_4$ mol/L	Phosphate KH_2PO_4 mol/L
B1	Training	0.67	0.56	0.044	0.02
B2	Testing	0.64	0.48	0.042	0.022
B3	Training	0.47	0.42	0.031	0.005
B5	Training	1	0.52	0	0.016
B6	Training	0.89	0	0.119	0.020
B7	Testing	0.65	0.55	0.042	0.007
B8	Testing	1.12	0.63	0	0.020

To be consistent with the work of Hans Roubos, batches 1,3,5,6 were used for training and batches 2,7, 8 for testing[36].

The data was scaled between zero and one and interpolated as described in the previous section. Derivatives were then taken of the interpolated profiles to obtain

[35] All bioreactor experiments were performed by Hans Roubos and Prebbn Krabben. Data was kindly released by DSM.

[36] Note Roubos refers to what have been termed 'testing batches' or 'unseen data' in this thesis as 'validation batches'.

$\frac{d\xi_i}{dt}$ at each time point. The model identified using the PCA-SVM methodology remains of the general form

$$\frac{d\xi}{dt} = \underbrace{Mp(\xi,t)}_{kinetics} - \underbrace{D\xi - Q(\xi) + F}_{transport}. \tag{6.3}$$

With the matrix M being the eigenvectors and P being the vector of kinetic functions corresponding to each principal component. As Figure 50 shows four degrees of freedom are sufficient to characterise the dynamics of the 11 measured series[37].

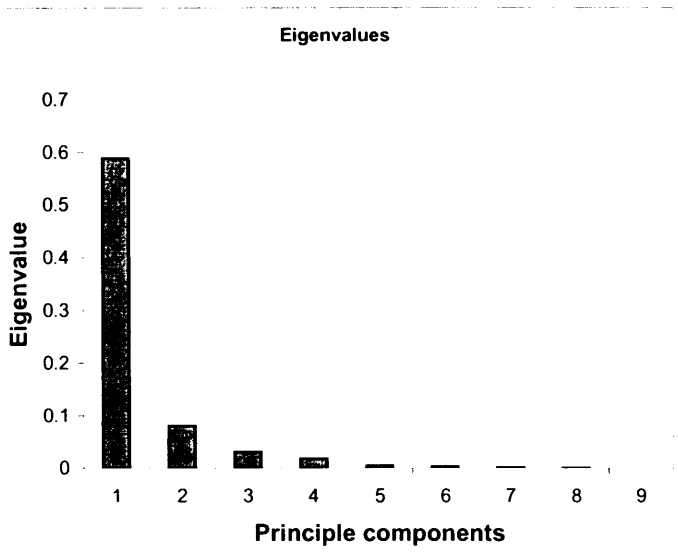


Figure 50. Eigenvalues of system.

However it should be noted that unlike the measurements for most of the variables, which are concentrations, the data for CO_2,O_2, and H+ are cumulative and on a 'total mass' basis rather than a 'per unit volume' basis. These cumulative outflows at any given time is therefore of the form:

$$\eta_{co_2}(\tau) = \int_0^{\tau} Vr_{co_2}. \tag{6.4}$$

With volume of the reactor decreasing each time samples are taken.

[37] The fit of the PCA matrix to training data can be seen in appendix C3.

6.3.4 Results.

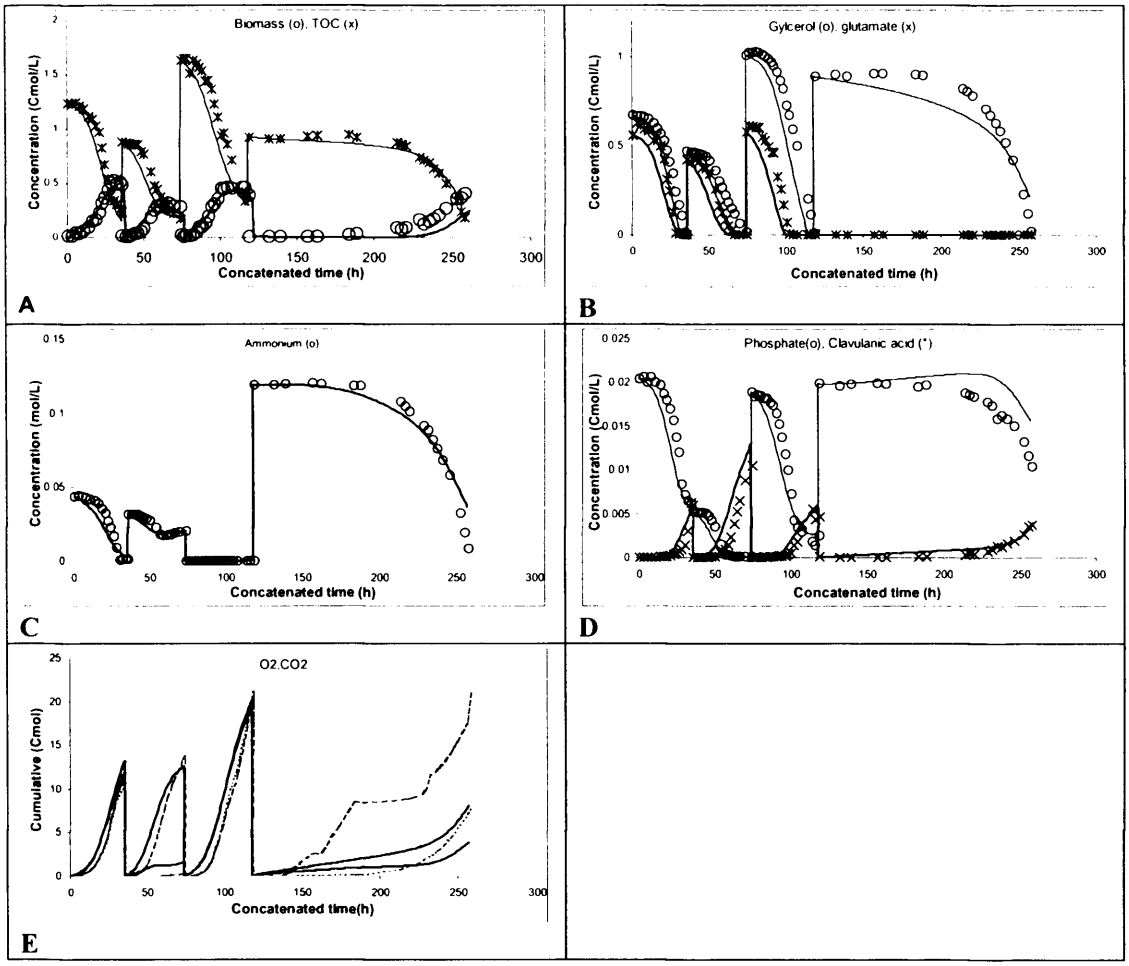


Figure 51. Hybrid SVM model performance on *S. clavuligerus* training batches: 1,3,5,6. Solid lines show model prediction. Points show measured data with (o) (*) as defined in each legend. Graph A shows biomass(o) and Total organic carbon(*) concentrations in Cmols/L. B shows glycerol(o) and glutamate(*) concentrations in Cmols/L. C shows Ammonia concentration(o) in mols/L. D shows phosphate(o) and clavulanic acid(*) concentrations in Cmols/L. E shows total cumulative oxygen consumption(dashed line) and carbon dioxide production(dotted line) in moles. Time is concatenated to enable all 4 batches to be displayed on a single graph.

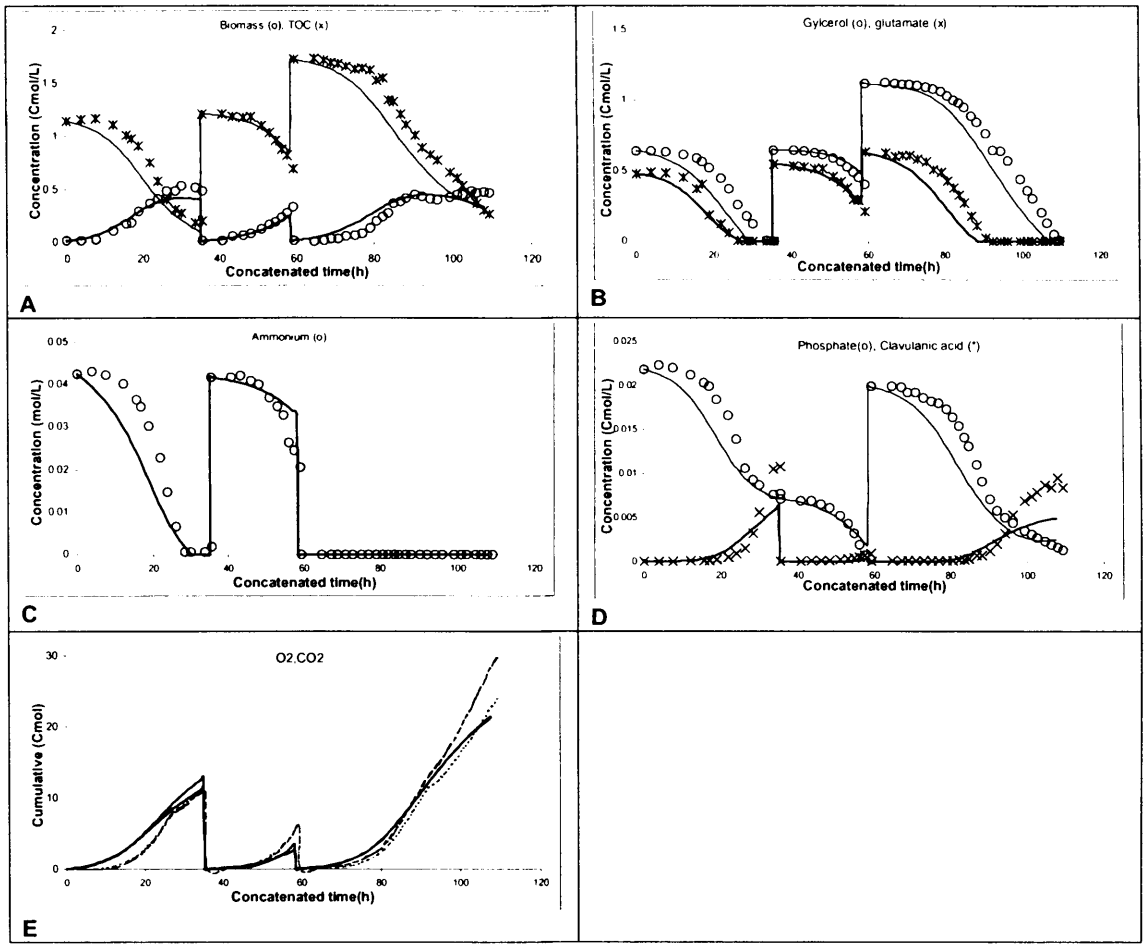
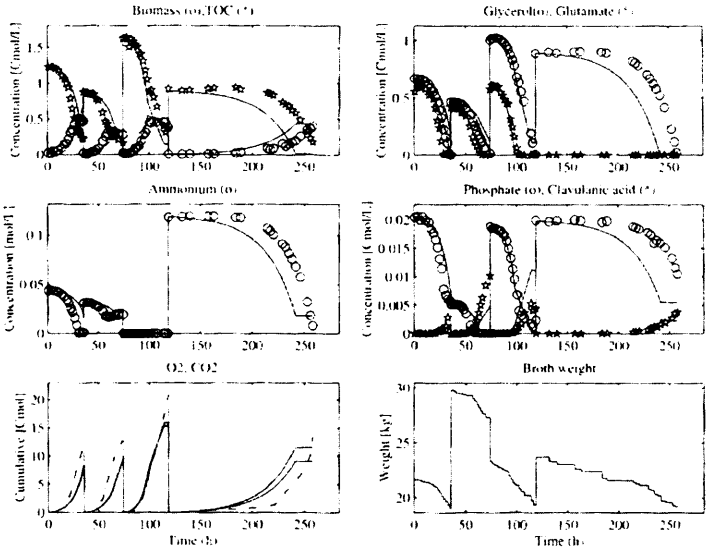
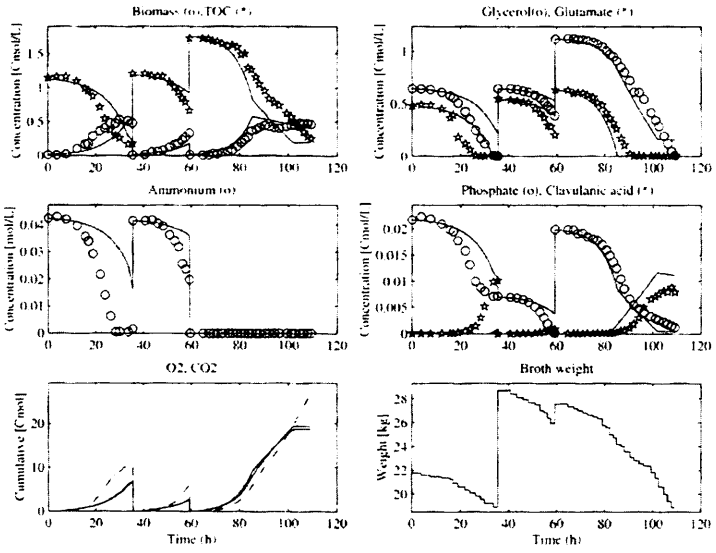


Figure 52. Hybrid SVM model performance on *S. clavuligerus* on testing batches. 2,7,8. Solid lines show model prediction. Points show measured data with (o) (*) as defined in each legend. Graph A shows biomass(o) and Total organic carbon(*) concentrations in Cmols/L. B shows glycerol(o) and glutamate(*) concentrations in Cmols/L. C shows Ammonia concentration(o) in mols/L. D shows phosphate(o) and clavulanic acid(*) concentrations in Cmols/L. E shows total cumulative oxygen consumption(dashed line) and carbon dioxide production(dotted line) in moles. Time is concatenated to enable all 3 batches to be displayed on a single graph.



(c) Identification; model with neural network model.



(d) Validation; model with neural network model.

Figure 53. Fits achieved by Hans Roubos using a hybrid metabolic-neural network model. Figure taken from Roubos, J. A.(2002). Solid lines show model prediction. Points show measured data with (o) (*) as defined in each legend. Figure c shows fit to training data, Figure d shows fit to testing data.

6.3.5 Discussion

The fit to training data is shown in Figure 51 and the fit to testing data in Figure 52. The fits achieved by Roubos using a hybrid metabolic feed forward neural network model are shown in Figure 53 for ease of comparison. Fits achieved by Roubos using Takagi-Sugeno Fuzzy logic and conventional kinetics can be found in appendix D3.

From the predicted profiles on the training data it can be seen that the O_2 consumption predicted for batch 6 is markedly different to that measured. However, the evidence suggests that it is the O_2 consumption data for batch 6 rather than the model prediction that is wrong. Firstly because it is normal in most aerobic batch fermentations for OUR and CER to closely follow each other. Secondly because the cumulative O_2 consumption data shown Roubos's in Figure 53 is not the same as the data used to train the model. It is somewhat encouraging that the model has corrected through PCA based denoising a flaw in the training data.

A second major discrepancy between the SVM hybrid model prediction and the testing data is the significant under estimation of the clavulanic acid production for testing batches 2 and 8. However as can be seen from Figure 53d Roubos's hybrid model predicts no clavulanic acid production for batch 2 and significantly overestimates the clavulanic acid production for batch 8. It is therefore clear that, while not without flaws, the fast PCA-SVM methodology is capable of building models of the *Streptomyces clavuligerus* batch process with similar accuracy to that achieved by conventional/NN/fuzzy models based round constraint matrix determined by a detailed metabolic network.

Judged on visual inspection[38] the naïve PCA-SVM technique appears to produce models of a comparable accuracy to those achieved by Hans Roubos using a detailed metabolic modelling. This result indicates that good interpolative models can be found using completely data driven models inferred using the SVM technique.

[38] Unfortunately the predictions of models built by Hans Roubos were only available in the form of graphs therefore no quantitative comparison between the RMS errors of models was possible.

6.4 Conclusion.

The support vector machine methodology has been demonstrated on 3 real batch cultivations: *Murine hybridoma*; *Saccharopolyspora erythraea* and *Streptomyces clavuligerus*. The methodology was able to infer key system dynamics in all cases. In particular on the *Streptomyces clavuligerus* system it has been demonstrated that the methodology is capable of building models of comparable accuracy (as judged on the basis of visual inspection) to published results achieved by Hans Roubos using a detailed metabolic network to determine constraints and three established modelling methods to determine reaction kinetics.

There are however two issues that should be considered:

- Firstly it is clear from the poor performance on the *S. erythraea* system the interpolation of the state vector ξ poses a problem for real systems where measurements are noisy and/or infrequent. This limitation motivates the development of "Bayesian hybrid modelling" in Chapter 8.

- Secondly it is clear that one must be cautious in drawing conclusions about the expected general performance of a modelling methodology from a qualitative judgement limited number of experimental systems. In the next chapter the support vector machine methodology is compared quantitatively with two other methodologies on 50 different simulated systems.

7. Comparison with existing techniques.

The addition of yet another modelling technique does not, in itself, represent an advance. It is necessary to compare the new technique to existing techniques. It is however surprisingly difficult to compare modelling methodologies for the following two reasons:-

- That a particular technique *'a'* outperforms another technique *'b'* on a particular data set does not necessarily imply that on a new dataset generated by an unknown system it would be rational to use *'a'* in preference to *'b'*.
- Model building is an interactive process between the user and the algorithm. While it would be wrong to accuse researchers of conscious bias there may be an unconscious tendency to put slightly more effort into new, or preferred methodologies, than existing benchmark methods.

The approach employed was therefore to take the user out of the loop by comparing completely automated modelling methodologies. This automation also allows the methods to be compared on data sets produced by multiple randomly generated dynamical systems. If there is a statistically significant difference between methodologies *'a'* and *'b'* on such a sample of dynamical systems then one can reasonably infer that the difference in performance is general.

7.1 Techniques compared

In the first part of this section the PCA methodology was compared with knowing the correct constraint matrix. In order to assess the performance of kinetic models produced using principle components to infer the kinetics models of two types were compared.

- Constraint matrix inferred using principle component analysis. Kinetics modelled by support vector machines.

$$\frac{d\xi}{dt} = \underbrace{M}_{\substack{inferred \\ using \\ PCA}} \times \underset{SVM}{f}(\xi, v) - D\xi + u \qquad (6.5)$$

- Constraint matrix known *a priori* Kinetics modelled by support vector machines.

$$\frac{d\xi}{dt} = \underbrace{K}_{\substack{correctly \\ known}} f_{SVM}(\xi, v) - D\xi + u \tag{6.6}$$

In the second part of this section the constraint matrix K was set to the correct constraint matrix and three methodologies for modelling the kinetics were compared, these were:-

- Kinetics modelled by: Feed forward neural networks.

$$\frac{d\xi}{dt} = K f_{FFNN}(\xi, v) - D\xi + u \tag{6.7}$$

- Kinetics modelled by: Genetic programming.

$$\frac{d\xi}{dt} = K f_{GP}(\xi, v) - D\xi + u \tag{6.8}$$

- Kinetics modelled by: Support vector machines.

$$\frac{d\xi}{dt} = K f_{SVM}(\xi, v) - D\xi + u \tag{6.9}$$

The support vector methodology is as described previously in section five. The feed forward neural network and genetic programming methodologies, including cross validation strategies, are detailed later in this chapter.

7.2 Basis of comparison

7.2.1 Systems used to compare the techniques.

The modelling methodologies were compared on separate data sets produced by simulating 50 different dynamical systems. A dynamic system consists of a set of kinetic functions and a set of constraints. Both the constraints and kinetic functions were generated randomly as detailed in appendix B2:

$$\overbrace{\frac{d\xi}{dt}}^{\substack{8 \text{ measured} \\ \text{series}}} = \overbrace{K}^{\substack{8\times4 \text{ matrix of random} \\ \text{floating point numbers} \\ \sim U(-1,1)}} \times \underbrace{f_{rnd}(\xi, v)}_{\substack{4\times1 \text{ vector of functions} \\ \text{in parse tree form} \\ \text{generated at random}}} - D\xi + u \tag{6.10}$$

Each of the 50 data sets consists of 20 batches of data created by running a particular generated system with random initial conditions, with 'sampling' every 'hour' of model time. This frequent sampling was necessary to obtain an accurate continuous

estimate of the state vector since, by definition, the behaviour of a randomly generated model is unknown and may include very sharp changes in the variables.

In each set three batches were designated for use as training and two for validation. The remaining 15 were reserved for testing. $\sigma = 0.1$ Gaussian noise was added to the training/validation batches.

7.2.2 Assessing the relative performance of each method.

For large numbers of test systems visual comparison[39] cannot be used to compare the performance of modelling techniques as so the basis of comparison needs to be some numeric measure of *'goodness of fit'* / *'accuracy of predictions'* such as the commonly used root mean squared error (RMS).

There is no reason to suppose that it is more important in general to accurately model series with high variances than series with small variances. The data was therefore normalised so that the variance of each series was between zero and one:-

$$scale(\xi_i(t)) = \frac{\left(\xi_i(t) - Min\left(\xi_i(t_s)\right)\right)}{\left(Max\left(\xi_i(t_s)\right) - Min\left(\xi_i(t_s)\right)\right)} \tag{6.11}$$

The root mean squared error between the prediction of each model and the scaled data was calculated for each batch as an average of the RMS error on each of the measured series:

$$RMS_b = \frac{\sum_{j=0}^{N_\xi - 1}\left(RMS_{b,j}\right)}{N_\xi}$$
$$RMS_{b,j} = \sqrt{\frac{1}{N_{meas}}\sum_{i=0}^{N_{meas}}\left(scale\left(\xi_j(t_s^{\langle i\rangle})\right) - scale\left(\xi_j^P(t_s^{\langle i\rangle})\right)\right)^2} \tag{6.12}$$

RMS error was then averaged over the five training batches generated by each dynamic system and the average RMS error on the 15 testing batches generated by the same system to give two statistics $\underset{\forall Train}{RMS}\binom{s}{m}$ and $\underset{\forall Test}{RMS}\binom{s}{m}$ which respectively indicate how well a particular modelling methodology m performs on training and testing data generated by a particular dynamical system s.

[39] A number of sample screenshots showing the difference between predicted profiles using the correct model and using the PCA model can be found in the appendix C1.

The average RMS errors on each system can be used to compare the relative performance of two different modelling techniques 'a' and 'b' in a pair wise fashion on each of the 50 systems i.e

$$\begin{aligned} &RMS_{\forall Train}\left(\begin{smallmatrix} s \\ a \end{smallmatrix}\right) vs\ RMS_{\forall Train}\left(\begin{smallmatrix} s \\ b \end{smallmatrix}\right) \\ &RMS_{\forall Test}\left(\begin{smallmatrix} s \\ a \end{smallmatrix}\right) vs\ RMS_{\forall Test}\left(\begin{smallmatrix} s \\ b \end{smallmatrix}\right) \\ &s = 0 \rightarrow 49 \end{aligned} \tag{6.13}$$

The possible outcomes this comparison are:

- Technique 'a' can be expected to have a lower RMS error on training and/or testing data than technique 'b'
- Technique 'b' can be expected to have a lower RMS error on training/testing and/or data than technique 'a'
- There is no statistically significant difference between the performance of the two techniques on training/testing and/or data.

The statistical significance (but not the practical significance) can be determined through the paired t-test if the differences between pairs $\left(RMS_{\forall Train}\left(\begin{smallmatrix} s \\ a \end{smallmatrix}\right) - RMS_{\forall Train}\left(\begin{smallmatrix} s \\ b \end{smallmatrix}\right)\right) or \left(RMS_{\forall Test}\left(\begin{smallmatrix} s \\ a \end{smallmatrix}\right) - RMS_{\forall Test}\left(\begin{smallmatrix} s \\ b \end{smallmatrix}\right)\right)$ are *approximately* normally distributed. If the data cannot be assumed to be normal the Wilcoxon signed-rank test is a more appropriate significance test.

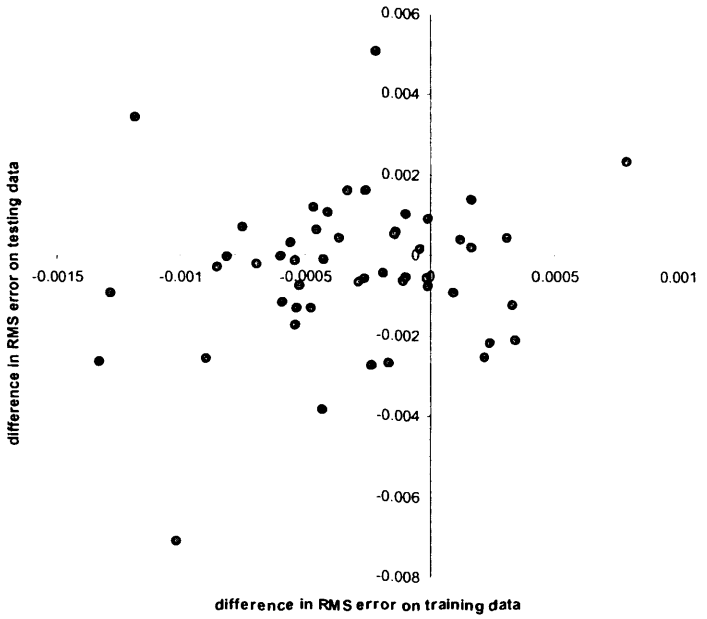
The Kolmogorov-Smirnov (KS) test was be applied to the test for normality and thus select the appropriate statistic. The paired t–test was applied if the differences were normally distributed at the 60%[40] significance level. In all other cases the Wilcoxon signed-rank test was used. Details of these statistical tests can be found in appendix E2.

[40] There is no consensus in the literature concerning what level of confidence that the data is normally distributed is required for the paired t-test to be used. A 60% level was chosen somewhat arbitrarily since it implies that the distribution is more likely to be normal than not. In any case the point is moot since the results of the paired t-test and Wilcoxon signed-rank test agree.

7.3 Results: Performance of the PCA methodology.

Figure 54 shows a comparison of the RMS errors of models built using the PCA methodology and models built knowing the correct constraint matrices on 50 randomly generated systems. Each point represents the average difference in RMS errors of models built using the two techniques measured on a randomly generated system. The x coordinate is the difference measured on training data generated by a randomly generated system. The y coordinate is the difference measured on testing data generated by the same randomly generated system.

$$x_s = \Delta \underset{\forall Train}{RMS}(s) = \underset{\forall Train}{RMS}\left({}^{s}_{PCA}\right) - \underset{\forall Train}{RMS}\left({}^{s}_{\text{correct } K}\right)$$
$$y_s = \Delta \underset{\forall Test}{RMS}(s) = \underset{\forall Test}{RMS}\left({}^{s}_{PCA}\right) - \underset{\forall Test}{RMS}\left({}^{s}_{\text{correct } K}\right) \qquad (6.14)$$
$$s = 0 \rightarrow 49$$



Training	Testing
p-value for the paired t-test=0	p-value for the paired t-test=0.18
p-value (Wilcoxon signed-rank test)=0	p-value (Wilcoxon signed-rank test)=0.14
Mean= -0.0003	Mean= -0.0004
sd=0.0004	sd=0.002

Figure 54. Scatter plot of the difference between the RMS errors of kinetic models where the constraint matrix has been inferred using PCA and where the constraint matrix is known. Each point represents the relative performance measured on a different system.

Figure 54 shows that in 80% of cases $\underset{\forall Train}{RMS}(\cdot_{PCA}) < \underset{\forall Train}{RMS}(\cdot_{\text{correct } K})$ and on 60% of cases $\underset{\forall Test}{RMS}(\cdot_{PCA}) < \underset{\forall Test}{RMS}(\cdot_{\text{correct } K})$. This means that, surprisingly, inferring the constraints using PCA appears to result in more accurate models than those built knowing the correct constraint matrix.

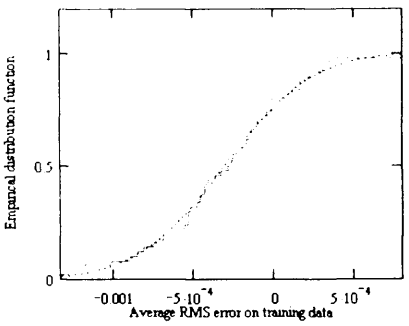


Figure 55. Solid line shows the cumulative distribution of difference in average RMS errors on training data. Dotted line shows the cumulative normal distribution.

D_n^+ =0.028 D_n^- =0.076 significance = 1

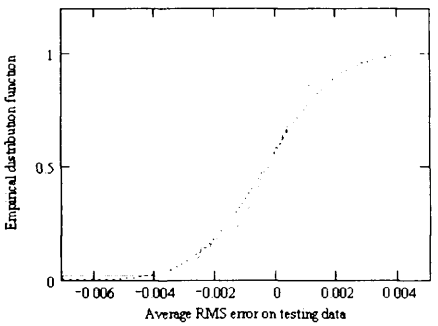


Figure 56 Solid line shows the cumulative distribution of difference in average RMS errors on training data. Dotted line shows the cumulative normal distribution.

D_n^+ =0.066 D_n^- =0.106 significance =0.601

As Figure 55 shows the $\underset{\forall Train}{\Delta RMS}(s)$ is distributed approximately normally. According to the KS test this result is significant at the 99.9% confidence level. $\underset{\forall Train}{\Delta RMS}(s)$ is also distributed approximately normally as shown in Figure 56, a result which is significant at the 60% confidence level. The paired t-test can therefore be applied to the difference on both training and testing data.

The t-test[41] indicates that the difference on training data is statistically significant at the 99.9% confidence level. The mean difference is however very small $\underset{\forall Train}{\overline{\Delta RMS}} = -0.0003$ and is unlikely to have any practical implications.

The difference on testing data is statistically significant only at the 72% confidence level. By convention the null hypothesis must be rejected above the 95% confidence for a statistically significance difference between the two distributions to be claimed.

[41] Approximately the same significance is given by the Wilcoxon signed-rank test.

Therefore it can concluded that:

- Inferring the constraint matrix using PCA can be expected to marginally improve the fit to training data compared with using the correct constraint matrix.
- No statistically significant difference can be observed on testing data.

The result is somewhat surprising since the imperfect inference of the constraint matrix by PCA was expected to introduce error compared with using a matrix that is by definition correct.

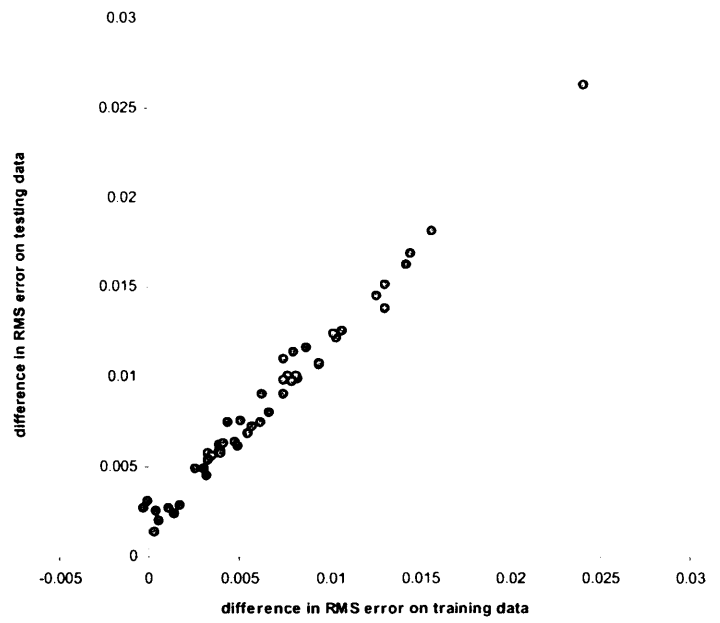
To investigate this surprising result it is necessary to separate the effect of the choice of constraint matrix from the effect of kinetic modeling. Figure 57 shows a plot of the difference in RMS errors for the same set of 50 systems but with the kinetics being modelled "perfectly[42]" by both methods. Perfect kinetics means that the kinetics are determined from the interpolation of the measured state as follows.

$$r(\xi,t) = K^{-1}\left(\frac{d\hat{\xi}^*(t)}{dt} + D(t)\hat{\xi}^*(t) - u(t)\right) \tag{6.15}$$

and:-

$$p(\xi,t) = M^T\left(\frac{d\hat{\xi}^*(t)}{dt} + D(t)\hat{\xi}^*(t) - u(t) - \bar{\Gamma}^*\right) \tag{6.16}$$

[42] Note that both models work with interpolated values $\hat{\xi}$ and hence we cannot expect the kinetics to be exactly the same as the true system. Also note that for the model with the correct matrix K the Penrose pseudo inverse $M^{-1} = V \times D \times U^T$ was used. This could introduce non negligible error if K was is not of full rank however this has near zero probability of occurring.



Training	Testing
p-value for the paired t-test=0 p-value for the Wilcoxon Signed-Rank Test=0 mean=0.0065 sd=0.004755	p-value for the paired t-test=0 p-value for the Wilcoxon Signed-Rank Test=0 mean=0.008553 sd=0.004858

Figure 57 Scatter plot of the differences in RMS error between models of 50 different systems where the kinetics have been 'modelled perfectly'

As Figure 57 shows, in almost every case $\underset{\forall Train}{RMS}\left(s_{PCA}\right) > \underset{\forall Train}{RMS}\left(s_{\text{correct } K}\right)$ and $\underset{\forall Test}{RMS}\left(s_{PCA}\right) > \underset{\forall Test}{RMS}\left(s_{\text{correct } K}\right)$. Inferring the constraint matrix using PCA introduces, on average, an additional RMS error of 0.006 on training data and an additional RMS error of 0.008 on testing data. This result is significant at the 99% confidence level according to both the paired t-test and the Wilcoxon signed rank test.

Therefore inferring the constraint matrix using PCA does introduce additional errors when compared with the correct model. However these errors are only noticeable in the context of models were the kinetics are perfectly modelled. When the kinetics are modelled by black box functions no statistically significant difference can be detected between models built knowing the correct constraint matrix and models built by inferring the constraint matrix from data using PCA.

7.4 Theoretical basis: a description of FFNN and GP methodologies.

7.4.1 Methodology for building models using genetic programming[43].

Genetic programming was used to produce models of the form:

$$\frac{d\xi}{dt} = K\, f_{GP}(\xi, v) - D\xi + u \tag{6.17}$$

where $f_{GP}(\xi, v) \in \Re^{N_\xi}$ is a set of equations represented in the form of GP trees. The function and terminal sets were defined as;

$$\begin{aligned} F &= \{+, -, \%, \times, sig, mm\} \\ T &= \{C, v_1, v_{2,\ldots,} v_{n_v}, \xi_1, \xi_2, \ldots, \xi_{N_\xi}\} \end{aligned} \tag{6.18}$$

where % denotes protected division, $sig(x) = \frac{1}{1+e^{-x}}$, $mm(x) = \frac{x}{x+1}$ and each C is a unique constant instantiated according to a predefined probability distribution, known in GP terminology as a ephemeral random constant (ERCs). The standard genetic programming algorithm is inefficient at determining real valued constants therefore a local search method (The modified simplex Nelder and Mead(1965)) was used to determine the values of these constants.

Two strategies were employed to minimise over-fitting. Firstly a regularisation constant was included in the fitness function. Secondly the individual with the best performance on validation data was saved every generation and the final model selected from this set of saved models. The overall algorithm is shown in Figure 58 below.

[43] A variation on this approach where the constraint matrix is the identity matrix is published as Hodgson, B. J. et al(2004).

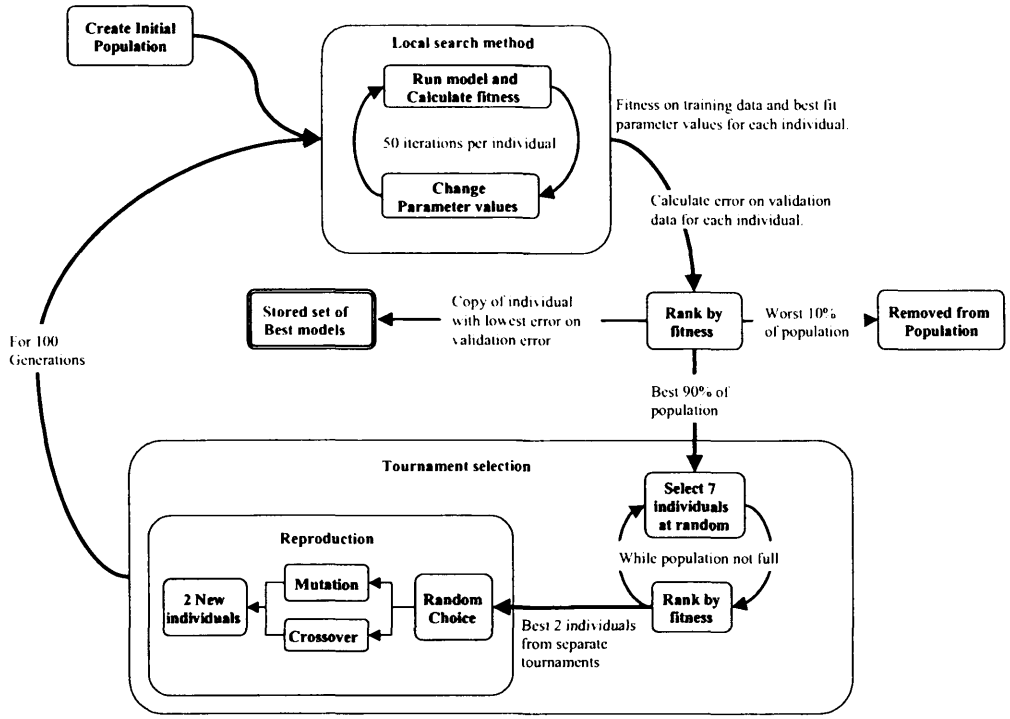


Figure 58. Modified Genetic programming Algorithm.

1) A population of 2000 models of the form given in (6.17) were created at random.

2) The performance of each model on training data was evaluated according the following *'fitness function'*: which is a weighed sum of the RMS error on training data and the number of nodes[44] in the individual.

$$fitness(individual) = \frac{1}{1 + N_{nodes} + C\frac{1}{N_{train}}\sum_{i=0}^{N_{train}-1} Error\left({}^{i}_{individual}\right)}$$

3) *where* (6.19)

$$Error\left({}^{batch}_{individual}\right) = \sum_{0}^{N_z}\sqrt{\sum_{i=0}^{N_{meas}}\left(\xi_s^{*}(t_s^{\langle i\rangle}) - \xi_s^{P}(t_s^{\langle i\rangle})\right)^2}$$

4) 50 iterations of the simplex method were used to locally optimise the values of the constants in each model so as to maximise the fitness of each individual (Nelder, J. A. et al(1965)) .

[44] The number of nodes in each individual ' N_{nodes} 'is a measure of the complexity of each model. The regularisation weight C, which determines the trade off between fit and complexity, was set to the value corresponding to the best of validation performance of 10 runs.

5) The RMS error on validation data was calculated for each individual and a copy of the individual with the lowest error on validation data added to a *set of saved models*.

6) The 10% of the population with the lowest fitness were removed and replaced by the offspring of the fittest individuals: Reproduction was determined by *'tournament selection'* as follows:

   a) A group of seven individuals is chosen at random from the population.

   b) The individual with the highest fitness of that group 'wins' and 'reproduces'. New individuals were created by either; crossover between the winners or successive tournaments or cloning and mutation of the winner of a single tournament.

   c) All the individuals are then returned to the population and can be reselected.

Steps 2-6 were repeated for 100 generations after 100 generations the model with the best fit to validation data was selected from the *set of saved models*. The selected model was then used for testing.

7.4.2 Methodology for building models using feed forward neural network.

Neural network models used in this section consist of a separate feed forward neural network for each of the full set of N_r equations defining the hybrid model

$$\frac{d\xi}{dt} = K \underset{FFNN}{f}(\xi, v) - D\xi + u \tag{6.20}$$

where $\underset{FFNN}{f}(\xi, v) \in \Re^{N_r}$ is a set of feed forward neural networks as discussed in chapter two with tanh as the activation function. The number of hidden layers and neurons in each layer was determined by building every possible network with between one and two hidden layers and between five and ten neurons per layer and then choosing the network structure with the lowest error on validation data for each kinetic function.

The output for a network with two hidden layers is given by:

$$\underset{NN}{f}(x,w) = \theta^{(2)} + \sum_{i=0}^{N^2_{neurons}-1} w_i^{(2)} \tanh\left(\theta_i^{(1)} + \sum_{j}^{N^1_{neurons}-1} w_{i,j}^{(1)} x'_i\right)$$

where (6.21)

$$x'_i = \tanh\left(\theta_i^{(0)} + \sum_{k}^{N_{inputs}-1} w_{i,j}^{(0)} x'^{-1}_j\right)$$

If the network structure only has one hidden layer the network output is given by:

$$\underset{NN}{f}(x,w) = \theta^{(1)} + \sum_{i=0}^{N^1_{neurons}-1} w_i^{(1)} \tanh\left(\theta_i^{(0)} + \sum_{k}^{N_{inputs}-1} w_{i,j}^{(0)} x'^{-1}_j\right) \quad (6.22)$$

Each network was trained to predict the reaction rate $r(\tau)$ as a function of the interpolated state vector $\hat{\xi}^*(\tau)$. The objective function for training is thus:-

$$\min_{w,\theta}\left(\frac{1}{\ell}\sum_{i=0}^{i-1}\left(\hat{r}^*(i\Delta t) - \underset{NN}{f}\left(\hat{\xi}^*(i\Delta t)\right)\right)^2\right) \quad (6.23)$$

Where $\hat{r}^*$ is the reaction rate obtained from training data. The validation error was determined separately for each kinetic function by simply evaluating the objective function given by equation (6.23) but with $\hat{r}^*$ determined from validation data.

The networks were trained by the *'back propagation algorithm'* (Rumelhart, D. E. et al(1986)) detailed in chapter three. An 'early stopping method' (Prechelt(1998)) was employed to prevent over-fitting due to the selection of large weight values by stopping the back propagation process when the validation data increased consistently for 6 successive epochs. Back propagation can become stuck in local optima therefore training process was repeated five times with different initial weights.

The systematic search[45] for a network topology is similar to the approach of van Can et al(1996) who varied the number of nodes of a one layer neural network from 1 to 25 in determining a grey box model of a dynamic process. It is also similar to the approach of Warnes, M. R. et al(1998) who investigated topologies ranging from a

[45] The architecture of each neural network was determined separately so as to minimise the error between the network prediction and the actual reaction rate of the reaction it represents. It would have been prohibitive to determine the architecture of the complete model since this would require considering $\left(\left(\max(N_{neurons}) - \min(N_{neurons})\right)\times\left(\left(\max(N_{neurons}) - \min(N_{neurons})\right)+1\right)\times N_{hidden\ layers}\right)^{N_r}$ which is clearly impractical.

$= (5\times 6\times 5)^4$

$= 506{,}250{,}000$ combinations

Even the approach used here was hugely time consuming since it requires 900 different neural networks to be trained using back propagation to produce one model.

single hidden layer with up to 12 nodes to two hidden layers with up to Five nodes in each layer. Hornik(1991) claims that solutions to most practical problems can be obtained with one hidden layer containing less than 10 neurons . Models ranging in complexity from 5 to 20 neurons should therefore have sufficient[46] capacity to model the data.

Pseudocode for the process can be found below. The shown algorithm returns the trained network with the lowest error on validation data of all those tried:

```
Main_function()
for(N_layers = 1 → 2)
    if(N_layers == 1)
        for(N^1_neurons = 5 → 10)
            train_network(1, N^1_neurons, 0)
    if(N_layers == 2)
        for(N^1_neurons = 5 → 10)
            for(N^2_neurons = 5 → 10)
                train_network(2, N^1_neurons, N^2_neurons)
return best network
```

```
train_network(N_layers, N^1_neurons, N^2_neurons)
for(r = 1 → 5)
    create network with N_layers, N^1_neurons, N^2_neurons and random weights
    for(iteration = 1 → 2000)
        do backpropogation_step
        calculate validation error
        if(validation error is getting consistently worse)
            return best_network
        if(validation error<best so far)
            best so far=validation error
            best_network=this network
return best_network
```

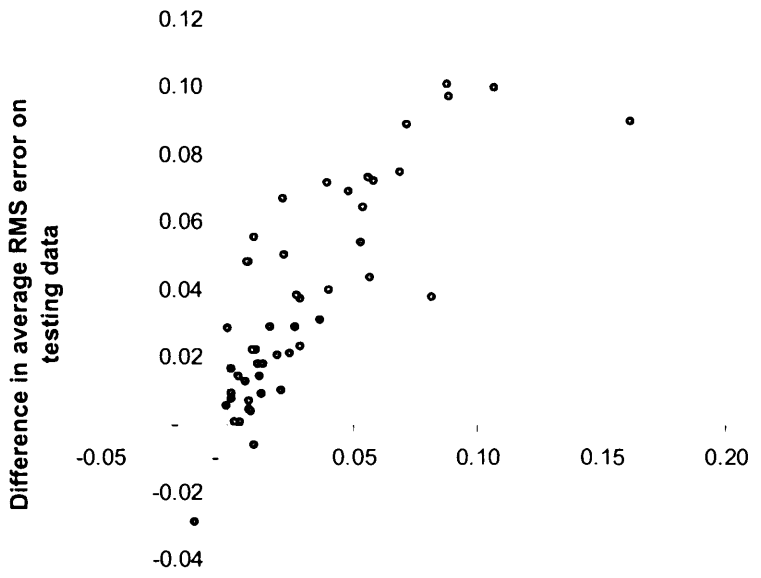
Figure 59 Pseudocode for identification of the 'best' neural network structure for each kinetic function.

7.5 Results: Comparison of three kinetic modelling methodologies.

Figure 60 shows the difference between the average RMS errors of models built using the SVM methodology and the FFNN methodology. The x coordinate is the difference measured on training data. The y coordinate is the difference measured on testing data.

$$\begin{aligned} x_s &= \underset{\forall Train}{\Delta RMS}(s) = \underset{\forall Train}{RMS}\left(\overset{s}{}_{FFNN}\right) - \underset{\forall Train}{RMS}\left(\overset{s}{}_{\text{SVM}}\right) \\ y_s &= \underset{\forall Test}{\Delta RMS}(s) = \underset{\forall Test}{RMS}\left(\overset{s}{}_{FFNN}\right) - \underset{\forall Test}{RMS}\left(\overset{s}{}_{\text{SVM}}\right) \\ s &= 0 \rightarrow 49 \end{aligned} \tag{6.24}$$

[46] The functions within the each randomly created system typically range from 6 to 15 nodes in length so regardless of whether a particular network structure is applicable to practical problems range of available topologies is sufficient for this artificial problem.



Training	Testing
p-value for the paired t-test=0 p-value for the Wilcoxon Signed-Rank Test=0 mean=0.03 sd=0.033	p-value for the paired t-test=0 p-value for the Wilcoxon Signed-Rank Test=0 mean=0.04 sd=0.031

Figure 60. Difference between performance of FFNN and SVM models on 50 different dynamical systems.

Since the points are concentrated in the top right hand quadrant of the graph, it is immediately clear, that on almost every data set generated, that the SVM methodology outperforms the neural network methodology. On 48 data sets models built using support vector machines to model the kinetics had a lower average RMS error on training data than models build using feed forward neural networks to model the kinetics. On 47 data sets models built using support vector machines to model the kinetics had a lower average RMS error on testing data than models built using feed forward neural networks to model the kinetics.

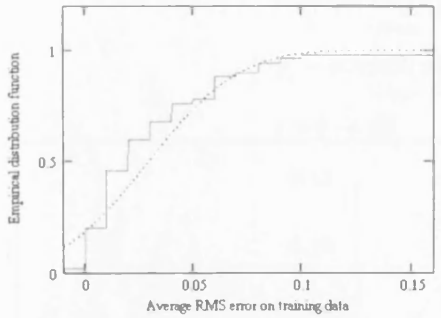


Figure 61 Solid line shows the cumulative distribution of difference in average RMS errors on training data. Dotted line shows the cumulative normal distribution.

D_n^+ =0.093 D_n^- =0.165 significance = 0.117

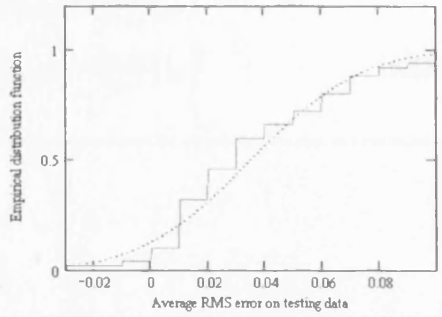


Figure 62 Solid line shows the cumulative distribution of difference in average RMS errors on testing data. Dotted line shows the cumulative normal distribution.

D_n^+ =0.038 D_n^- =0.107 significance = 0.597

As Figure 61 and Figure 62 show the variables cannot be assumed to be normal distributed since the hypothesis that the variables are normally distributed is significant only at the 11.7% level for training data and 59.7% level for testing data. Since the variables are not normally distributed the Wilcoxon Signed-Rank Test rather than the t-test should be applied. According to this test the difference between the SVM and FFNN techniques on both training and testing data is statistically significant at the 99.9% confidence level.

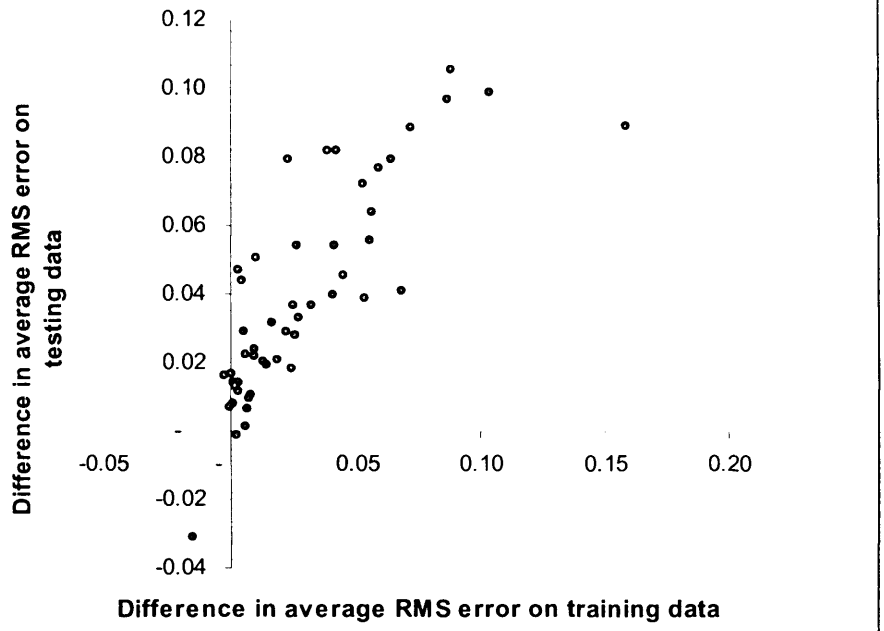
The mean differences $\overline{\underset{\forall Train}{\Delta RMS}} = -0.03$ and $\overline{\underset{\forall Test}{\Delta RMS}} = -0.04$ is non negligible and depending on the context the difference in accuracy may be of practical significance.

Therefore

- Models built using the SVM methodology to infer the kinetics can be expected to have a lower RMS error on both training and testing data compared with models built using the feed forward neural network methodology.

Figure 63 shows the difference between the average RMS errors of models built using the GP methodology and the FFNN methodology. The x coordinate is the difference measured on training data. The y coordinate is the difference measured on testing data.

$$x_s = \Delta \underset{\forall Train}{RMS}(s) = \underset{\forall Train}{RMS}\left(\text{}_{FFNN}\right) - \underset{\forall Train}{RMS}\left(\text{}_{GP}\right)$$
$$y_s = \Delta \underset{\forall Test}{RMS}(s) = \underset{\forall Test}{RMS}\left(\text{}_{FFNN}\right) - \underset{\forall Test}{RMS}\left(\text{}_{GP}\right) \quad (6.25)$$
$$s = 0 \rightarrow 49$$



Training	Testing
p-value for the paired t-test=0	p-value for the paired t-test=0
p-value for the Wilcoxon Signed-Rank Test=0	p-value for the Wilcoxon Signed-Rank Test=0
mean=0.03	mean=0.04
sd=0.033	sd=0.034

Figure 63. Difference between performance of FFNN and GP models on 50 different dynamical systems.

On almost every data set the GP methodology outperforms the feed forward neural network methodology. On 44 data sets models built by using genetic programming to model the kinetics had a lower average RMS error on training data than models built using feed forward neural networks to model the kinetics. On 45 data sets models build by using genetic programming to model the kinetics had a lower average RMS error on testing data than models built using feed forward neural networks to model the kinetics.

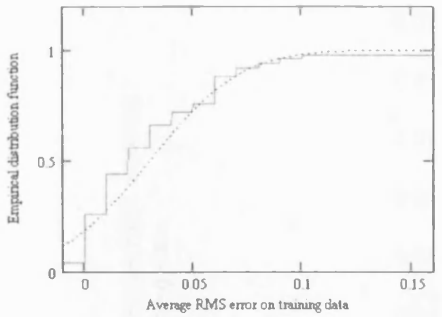


Figure 64 Solid line shows the cumulative distribution of difference in average RMS errors on training data. Dotted line shows the cumulative normal distribution.

D_n^+ =0.058 D_n^- =0.147 significance = 0.214

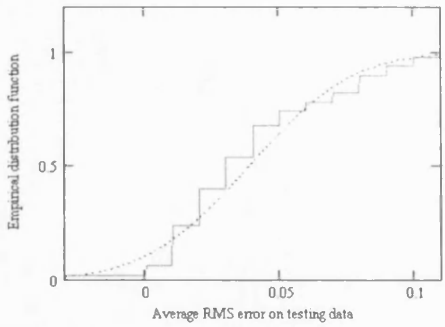


Figure 65 Solid line shows the cumulative distribution of difference in average RMS errors on testing data. Dotted line shows the cumulative normal distribution.

D_n^+ =0.036 D_n^- =0.116 significance = 0.483

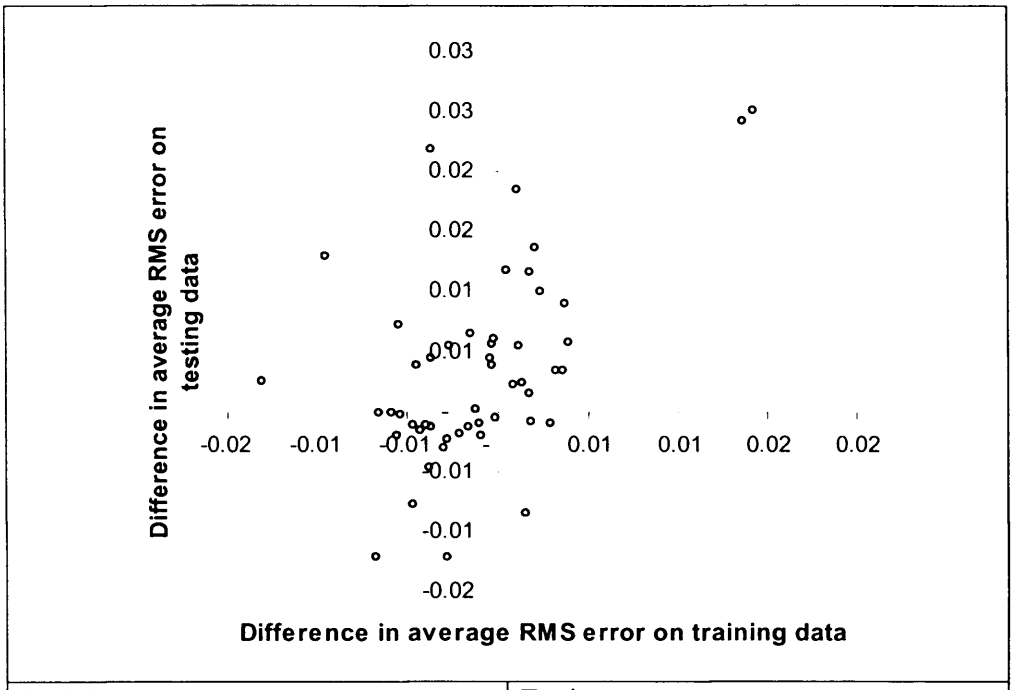
As shown in Figure 64 and Figure 65 the variables are not normally distributed at the 60% significance level therefore the Wilcoxon Signed-Rank Test should be applied. According to this test the difference between the SVM and FFNN techniques on both training and testing data is statistically significant at the 99.9% confidence level. The mean differences $\overline{\underset{\forall Train}{\Delta RMS}} = -0.03$ and $\overline{\underset{\forall Test}{\Delta RMS}} = -0.04$ are both non negligible.

Therefore

- Models built using the GP methodology to infer the kinetics can be expected to have a lower RMS error on both training and testing data compared with using the feed forward neural network methodology.

Figure 66 shows the difference between the average RMS errors of models built using the SVM methodology and the GP methodology. The x coordinate is the difference measured on training data. The y coordinate is the difference measured on testing data.

$$\begin{aligned} x_s &= \underset{\forall Train}{\Delta RMS}(s) = \underset{\forall Train}{RMS}\left(\begin{smallmatrix} s \\ SVM \end{smallmatrix}\right) - \underset{\forall Train}{RMS}\left(\begin{smallmatrix} s \\ \mathrm{GP} \end{smallmatrix}\right) \\ y_s &= \underset{\forall Test}{\Delta RMS}(s) = \underset{\forall Test}{RMS}\left(\begin{smallmatrix} s \\ SVM \end{smallmatrix}\right) - \underset{\forall Test}{RMS}\left(\begin{smallmatrix} s \\ \mathrm{GP} \end{smallmatrix}\right) \\ s &= 0 \rightarrow 49 \end{aligned} \tag{6.26}$$



Training	Testing
p-value for the paired t-test=0.08	p-value for the paired t-test=0
p-value for the Wilcoxon Signed-Rank Test=0.02	p-value for the Wilcoxon Signed-Rank Test=0.01
mean= -0.001	mean= +0.003
sd=0.05	sd=0.08

Figure 66. Difference between performance of SVM and GP models on 50 different dynamical systems.

The comparison of SVM and GP models gives a less clear result than previous comparisons. On training data, models built by using support vector machines to model the kinetics had a lower average RMS error on 47 data sets than models built by using genetic programming to model the kinetics. On testing data Models build by using support vector machines to model the kinetics had a lower average RMS error than models build using genetic programming on 24 data sets.

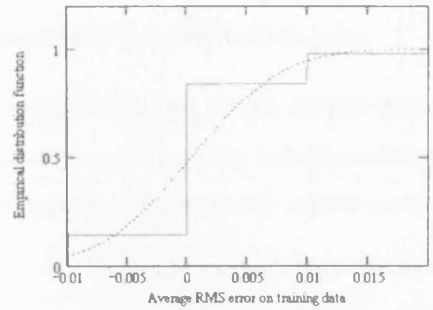


Figure 67 Solid line shows the cumulative distribution of difference in average RMS errors on training data. Dotted line shows the cumulative normal distribution.

D_n^+ =-0.019 D_n^- =0.334 significance = 0

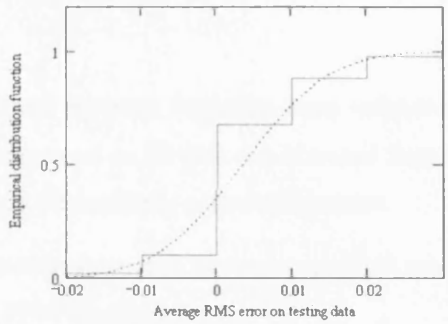


Figure 68 Solid line shows the cumulative distribution of difference in average RMS errors on testing data. Dotted line shows the cumulative normal distribution.

D_n^+ =0 D_n^- =0.255 significance = 0

As Figure 67 and Figure 68 show the KS test gives a 0% significance level for the hypothesis that the variables are normally distributed. The variables cannot be assumed to be normal distributed therefore the Wilcoxon Signed-Rank Test should be applied. According to this test the difference between the SVM and GP techniques on training data is statistically significant at the 98% confidence level and the difference on testing data is significant at the and testing 99% confidence level. The mean differences $\overline{\Delta RMS}_{\forall Train} = -0.001$ and $\overline{\Delta RMS}_{\forall Test} = 0.003$ very small and while statistically significant they are unlikely to have any practical implications.

- Models built using the SVM methodology to infer the kinetics can be expected to have a slightly lower RMS error on training data compared with using the genetic programming methodology.
- Models built using the GP methodology to infer the kinetics can be expected to have a slightly lower RMS error on testing data compared with using the SVM methodology.

7.6 Conclusion.

In the first part of this chapter the accuracy of 50 models built using PCA based method for inferring the constraint matrix were compared with 50 models built knowing the correct constraint matrix. Surprisingly it was found that inferring the

constraint matrix using the PCA methodology did not result in a statistically significant increase in RMS error.

In the second part of this chapter three automatic methods, including cross validation strategies, for building hybrid models were compared on 50 data sets obtained from a simulated hybridoma cell culture system and on 50 randomly generated systems.

- The SVM methodology can be expected in general to have a lower RMS error on both training and testing data than FFNN methodology.
- The GP methodology can be expected in general to have a lower RMS error on both training and testing data than FFNN methodology.
- The SVM methodology an be expected in general to have a slightly lower RMS error than the GP methodology on training data but a larger error on testing data.

A natural question to ask is whether the differences between each modelling methodology are significant in the context of bioprocess modelling. The question of whether a particular level RMS error is important cannot be answered in the general case since it depends on the expected loss associated with inaccurate prediction and therefore requires specific information relating to the process of interest. The strategy of using multiple randomly generated problems as a basis by which to compare automated methodologies, is in itself of note. It provides a principled method for comparing modelling methodologies and thus drawing general conclusions rather than conclusions restricted to a single test system. It does however open up a difficult question[47] with practical and philosophical implications: *What distribution should the dynamical systems used for comparing modelling methodologies be drawn from?*

[47]Ideally, of course, the distribution would reflect our prior beliefs about the normal behaviour of the class of systems of interest. But formalising this, let alone drawing from such a distribution, presents a serious challenge and is beyond the scope of this thesis .An alternative and related question would be: *What relationship is there between the performance of different methodologies on each test system and the characteristics of that dynamical system.*

8. A Bayesian approach to hybrid modelling.

In this chapter a Bayesian approach to hybrid modelling is described. This is a principled framework for making use of prior beliefs and accounting for uncertainty.

Firstly, the maximum *a posteriori* method for identifying model parameters is described. This approach is then extended to produce model predictions based not on a single set of parameter values but rather 'marginalised' over the model's uncertain parameters, so as to make predictions in the form of a probability density function. The model structures can be any mixture of mechanistic equations and neural networks.

It is then shown, on simulated systems, that the Bayesian approach can infer useful models from data with missing measurements, where the hybrid modelling method described in previous sections would fail.

Finally, an attempt is made to provide a framework for selecting the 'best' model structure from a set of candidate equations and for marginalising over the set of possible model structures to take into account uncertainty.

8.1 Introduction.

8.1.1 Motivation.

The methodology for kinetic modelling outlined in section 3 has been shown to be capable of building accurate models of real and simulated systems. However, it is also clear that it suffers from three important deficiencies:-

(1) The method requires continuous signal $\hat{\xi}^*(t) \approx \xi(t)$ for the full state vector. Firstly, the use of $\hat{\xi}^*(t)$, estimated from an interpolant or state observer rather than directly using the measured data, may introduce systematic bias into the model. Secondly, where species are unmeasured, or samples are infrequent, data cannot be used to build models.

(2) It provides no systematic method for incorporating prior knowledge. Knowledge can be incorporated into models in various ad hoc forms such as:

$$r(\xi,v) = \underset{1}{f}(\xi,v) \tag{8.1}$$

$$r(\xi,v) = \underset{1}{f}(\xi,v) \times \underset{SVM}{f}(\xi) + \underset{2}{f}(\xi,v) \tag{8.2}$$

$$r(\xi,v) = \underset{2}{f}(\xi,v) + \left(\underset{1}{f}(\xi,v) - \underset{2}{f}(\xi,v) \right) \times \underset{SVM}{f}(\xi) \tag{8.3}$$

where $\underset{1}{f}(\xi,v), \underset{2}{f}(\xi,v)$ are fully specified functions with no unknowns.

However, if less certain information in the form of a prior belief is available for example a statement such as; "*the rate is determined by either inhibited Michaelis Menton kinetics or is first order with respect to the substrate*" or "*the maximum specific growth rate is between 1.3h^{-1} and 0.8 h^{-1}*", there is no systematic way of making use of this information.

(3) When the model is used to predict the process behaviour under new conditions, no information is given as to the expected accuracy of the prediction.

8.1.2 Further motivation.

Engineering can be viewed as the science and art of rational decision-making. A rational decision, such as a choice of operating conditions, is the one that can be 'expected' to produce the best outcome given all the information available at the time the decision is made. Formally the best decision 'a' is the one that maximises the 'expected' utility (Berger(1980)):-

$$\max_{a} \int \underbrace{U(x,a)}_{\text{utility of outcome}} \times \underbrace{P(x|a)}_{\substack{\text{probability of outcome} \\ \text{given some choice } a \\ \text{has been made}}} dx \tag{8.4}$$

where the variable x represents the possible outcomes of decision a and the

utility function $U(x,a)$ reflects the payoff received if the world is in state x. This function may well be very complex. For example, if the goal was to optimise the profitability of a process then $U(x,a)$ would have to take into account: material costs; utility costs; penalties for crossing validated process conditions manpower requirements; as well as process yields.

Ideally, if not practically, expression (8.4) could be used to make any sequence of decisions such as designing optimum experiments and making decisions on the basis

of the results: -An overview of Bayesian experimental design can be found in Clyde(2001).

If the utility function accurately reflects the goals of the decision maker and the conditional probability[48] $P(x|a)$ ascribed to each outcome is correct then expression (8.4) is provably the best decision making strategy[49]. What is therefore required for optimum decision making is a method for rationally assigning conditional probabilities to outcomes, based on all available information about system behaviour, rather than simply making point predictions.

8.2 Test system.

Throughout this section the following test system, consisting of two reactions involving four species and no transport effects, is used for demonstrating the methods as they are developed.

$$\begin{array}{l} A \xrightarrow{r_1} B + C \\ A + B \xrightarrow{r_2} D \end{array} \left| \begin{array}{l} \dfrac{dC_A}{dt} = -r_1 - r_2 \\ \dfrac{dC_B}{dt} = r_1 - r_2 \\ \dfrac{dC_C}{dt} = r_1 \\ \dfrac{dC_D}{dt} = r_2 \end{array} \right. \tag{8.5}$$

r_1 and r_2 are two 'reactions' chosen from the following two sets of kinetic functions.

$$r_1 \in \left\{ w_0 C_A, w_0 C_A + w_0 C_A{}^2, \frac{w_0 C_A}{w_1 + C_A}, \frac{w_0 C_A}{w_1\left(1 + \frac{C_B}{w_2}\right) + C_A}, \frac{w_0 C_A}{w_1\left(1 + \frac{C_C}{w_2}\right) + C_A} \right\} \tag{8.6}$$

[48] There are two meanings of probability. The *frequentialist* position is that probabilities describe the frequencies of outcomes in random experiments. The *Bayesian* position is that probabilities describe degrees of belief regarding an unknown outcome. This section takes a firmly Bayesian position but readers should note that the disagreement is a continuing philosophical battle. A 'correct' probability is therefore a justified belief, the belief need not be true only rational given all current information. For a detailed course in Bayesian statistics and inference see "Probability Theory: The Logic of Science" Jaynes E.T.(2003) and "Information Theory, Inference, and Learning Algorithms"MacKay(2004).

[49] See "Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability" Marcus Hutter(2004) for proof that an agent based on Bayesian mixture prediction and decision theory will outperform all other agents in general.

$$r_2 \in \left\{ w_0 C_A C_B,\ w_0 C_A C_B + w_1 C_A^2 C_B + w_2 C_A C_B^2,\ \frac{w_0 C_A C_B}{(w_1 + C_A)(w_2 + C_B)},\ \frac{w_0 C_A C_B}{w_1 C_A + w_2 C_B + C_A C_B} \right\} \tag{8.7}$$

Since the test systems are for use as demonstrative examples and not intended to be representative of any specific real world process the values of the constants (w_0, w_1, w_2) chosen from a uniform distribution over the interval $[0,5]$ denoted $U(0,5)$. Unless otherwise stated training data consists of three batches of data generated by simulating the system from $t = 0$ to $t = 2.5$ with a step length of 0.01. Initial conditions for each batch are concentrations drawn from a uniform distribution $U(0,10)$. 11 samples were taken at regular intervals and $\sigma = 0.1$ Gaussian noise was added to each sample.

8.3. Posterior likelihood methods.

8.3.1 Identification of parameters.

A common problem in classical modelling is determining the parameters of a model where the model structure is known. The model structure is denoted H_i and the unknown parameters of this model as w. A central problem of parameter identification is that for complex models there may not be a unique combination of parameter values that fit the data. In this case the problem is termed *'ill posed'*, which means it can only be solved for a unique solution if additional information is available.

The Bayesian approach to parameter estimation makes use of prior information about the possible values each parameter can take in the form of a probability density function $P(w|H_i)$ of dimension equal to the number of parameters in the model $N_w = rows(w)$.

Assuming the proposed model structure H_i is true, then the probability of the parameters being a particular set of values w is given by Bayes theorem as:-

$$\underbrace{P(w|D,H_i)}_{\text{Posterior}} = \frac{\overbrace{P(D|w,H_i)}^{\text{Likelihood}}\ \overbrace{P(w|H_i)}^{\text{Prior}}}{\underbrace{P(D|H_i)}_{\text{Evidence}}} \tag{8.8}$$

The most likely set of values for w are those that maximise the posterior. The problem of parameter identification becomes one of optimisation.

The likelihood $P(D|w,H_i)$ is the likelihood of the parameters given the data. It is calculated by determining the probability[50], predicted by the model, of obtaining the measured training data with the chosen model parameters.

The evidence term $P(D|H_i)$ is constant for a given model and therefore can be ignored for the purposes of parameter estimation.

The prior $P(w|H_i)$ reflects any knowledge we have about the distribution of the parameters. The prior probability of a given set of parameter values can then be obtained simply by calculating the probability of each parameter value according to its prior distribution. From each elementary probability the prior probability of the whole parameter vector can be obtained:-

$$P(w|H_i) = \prod_{j=0}^{N_w-1} P(w_j|H_i) \tag{8.9}$$

For example, an engineer might know from experience that the maximum specific growth rate must lie within a certain range and thus choose a uniform distribution between these bounds[51]. Alternatively they may have obtained a value from literature and consider this value likely, in which case a distribution having a maximum probability at this point might be an appropriate prior. The problem of specifying priors and particularly 'non informative priors' representing ignorance is an area of ongoing research in statistics and beyond the scope of this thesis.

8.3.2 Calculating the likelihood.

Consider the simple problem of maximum likelihood regression with one output variable $y_i \in \Re$, where measurement error is Gaussian and the standard deviation is

[50] Technically the likelihood is not a probability density function but an 'inverse probability'. Since the data D is fixed it makes no sense to say "*likelihood of the data*" rather one says "*likelihood of the parameters given the data*".

[51] The specification of priors is beyond the scope of this thesis. However, it is worth highlighting the difficulty of specifying non-informative priors. It is not possible to be equally ignorant of the values of parameters and the predictive distribution of some non-linear function of these parameters e.g. a uniform distribution over parameter values may imply a non-uniform distribution over the model output.

the same for all points. The likelihood of the parameters for the model $y_i = H(x_i, w)$ given data $y = \left[y^0, y^1 \ldots y^{N_{meas}-1}\right]$ $y \in \Re$, $x = \left[x^0, x^1 \ldots x^{N_{meas}-1}\right]$ $x^i \in \Re^{N_\xi}$ is as follows:-

$$P(D|w,H) = \prod_{i=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\left(y^i - H(x^i,w)\right)^2}{2\sigma^2}\right) \tag{8.10}$$

Notice that since N_{meas} and σ are constants maximising the likelihood is equivalent to the classical least squares fit:-

$$\arg\min_w \left(\sum_{i=0}^{N_{meas}-1} \left(y^i - H(x^i,w)\right)^2\right) \tag{8.11}$$

For large amounts of data the likelihood $P(D|w,H)$ can become very small and lead to computational issues and so it is normal to work with the log of the likelihood. For multiple outputs series, assuming a constant Gaussian noise model for each output, the log likelihood is:-

$$\ln\left(P(D|w,H)\right) = -\frac{N_\xi N_{meas}}{2}\ln 2\pi - \frac{N_\xi}{2}\ln|\underset{\sim}{V}| - \frac{1}{2}Tr\left[\underset{\sim}{V}^{-1}M(w)\right] \tag{8.12}$$

Where; N_ξ is the number of independent variables, N_{meas} the number of measurements, $\underset{\sim}{V}$ is the measurement error covariance matrix. Tr denotes the trace of this matrix, $|\underset{\sim}{V}|$ its determinant and $M(w)$ is the moment matrix of residuals:-

$$\begin{aligned} &M(w) = \sum_{i=0}^{N_{meas}-1} residual_i(w)residual_i(w)^T \\ &where \\ &residual_i(w) = Y^i - H(x^i,w),\ Y^i \in \Re^{N_\xi},\ H(x^i,w) \in \Re^{N_\xi} \end{aligned} \tag{8.13}$$

For one variable equation (8.12) reduces to least squares regression and for multiple measured variables, if there is no covariance, equation (8.12) reduces to weighed least squares regression. e.g. for a 2 variable system $N_\xi = 2$:-

$$\min_w \left\{\frac{1}{\sigma_0^2}\sum_{i=0}^{N_{meas}-1}\left(Y_{0,i} - \hat{Y}_{0,i}\right)^2 + \frac{1}{\sigma_1^2}\sum_{i=0}^{N_{meas}-1}\left(Y_{1,i} - \hat{Y}_{1,i}\right)^2\right\} \tag{8.14}$$

The general hybrid-modelling problem involves fitting to multiple series where the standard deviation of the measurement error for each series (σ_i) is unknown.

Knightes C.D and Peters C.A(2000)[52] tackle a relevant problem of maximum likelihood estimation of the parameters of Monod Biodegradation models where both biomass and substrate are measured. They estimate the error covariance alongside the model as follows:-

$$\underset{\sim}{V}(w) = \frac{1}{N_{\xi} - N_w / N_{meas}} M(w) \tag{8.15}$$

In view of this application and since negligible covariance cannot be assumed this seems like a directly applicable approach. However, since (8.15) suggests the error covariance structure depends on H_i and number of parameters k this could introduce bias when comparing models and hence this method should only be used for parameter identification.

8.3.3 Maximum *a posteriori* parameter values.

The posterior $P(w|D,H_i)$ can be maximised by choosing those values of w which maximise $P(D|w,H_i)P(w|H_i)$. This can be calculated in terms of the log un-normalised posterior by adding the log prior $\ln P(w|H_i) = \sum_{j=0}^{N_w-1} \ln P(w_j|H_i)$ to the log likelihood. The MAP parameter values are therefore those which solve the following optimisation problem:-

$$w_{MAP} = \arg\max_{w} \left(\ln P(D|w,H) + \ln P(w|H_i) \right) \tag{8.16}$$

Simple example.

Consider the toy problem of fitting the parameters of a model of a competitively inhibited enzyme model.

$$r(E,S,I,V_{\max},k_m,k_I) = \frac{V_{\max} ES}{K_m\left(1+\frac{I}{k_I}\right)+S} \tag{8.17}$$

[52] Note the log likelihood equation printed in their paper carries a typographic mistake - a missing minus sign.

Leaving any attempt at biological plausibility to one side, values of the constants $V_{max} = 0.6, K_m = 0.5, K_I = 0.025$ were chosen arbitrarily and data generated at 25 random points over the range $[0,10]$ for E,S and I. Training data consists of 'measurements' of the enzyme rates with Gaussian noise $(\sigma = 0.025)$ at these 25 conditions. Testing data consists of noise free 'measurements' at 1000 conditions within this same range.

Finding MAP parameter values.

ssuming a uniform prior U(0.001,2) for all parameters the most likely parameter values w_{MAP} were then found by maximising the log posterior as given by equation (8.16). The MAP estimates obtained were:-

$$V_{max^{MAP}} = 0.654, K_{m^{MAP}} = 0.804, K_{I^{MAP}} = 0.037$$

The estimates for V_{max} and K_I appear to be reasonably close to the true values however, the estimate for K_m is not as good. As the plots below show the poor estimate of K_m does not translate into inaccurate predictions on unseen data.

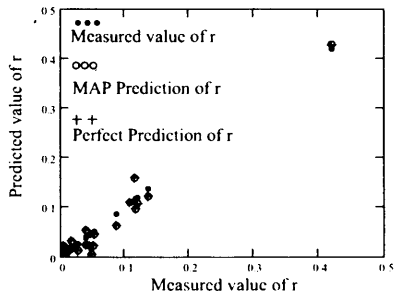


Figure 69. Fit of MAP model to 25 training data points.

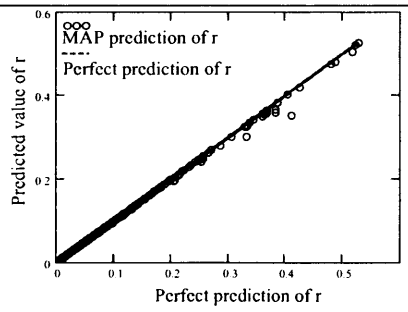


Figure 70. Prediction of MAP model on 1000 noise free validation points.

Figure 69 shows the fit to the training data. The x axis is the measured rate at each data point. The y axis position of the circles shows the rate predicted by the model at the conditions corresponding to each data point. The crosses show the true rate at the conditions corresponding to each data point without noise.

Figure 70 shows the performance of the fitted model on a large validation data set. The x axis and y position of the dots are noise free rates. The circles are the predictions of the fitted model.

Calculating credible intervals for parameter values.

The parameter covariance matrix V_w can be found by evaluating a Hessian round the MAP parameter values.

$$V_w = A^{-1} = \left(\nabla\nabla \ln P(w|D,H_i)\big|_{w_{MAP}} \right)^{-1} \tag{8.18}$$

The elements of the matrices being:-

$$\begin{pmatrix} \sigma_1^2 & \cdots & \sigma_1\sigma_n \\ \vdots & \ddots & \vdots \\ \sigma_n\sigma_1 & \cdots & \sigma_n^2 \end{pmatrix} = - \begin{pmatrix} \dfrac{\partial^2 \ln P(w|D,H_i)\big|_{w_{MP}}}{\partial w_1^2} & \cdots & \dfrac{\partial^2 \ln P(w|D,H_i)\big|_{w_{MP}}}{\partial w_1 \partial w_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 \ln P(w|D,H_i)\big|_{w_{MP}}}{\partial w_n \partial w_1} & \cdots & \dfrac{\partial^2 \ln P(w|D,H_i)\big|_{w_{MP}}}{\partial w_n^2} \end{pmatrix}^{-1} \tag{8.19}$$

where σ_n is the standard deviation with respect to parameter n.

Following Knightes C.D et al(2000) the Hessian was evaluated by fitting, in a linear least squares sense, a second order polynomial to the likelihood function in the region around the maximum likelihood parameters and differentiating the polynomial analytically. It is difficult to illustrate this for three parameters so for illustration purposes K_l is held constant at $K_{l\,MAP}$. Figure 71 shows the sample points used to fit the second order polynomial. Figure 72. shows the true log likelihood (z axis) and polynomial approximation (surface).

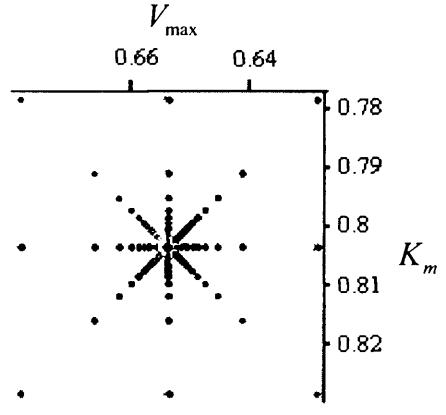


Figure 71. Sample points used to construct the Hessian.

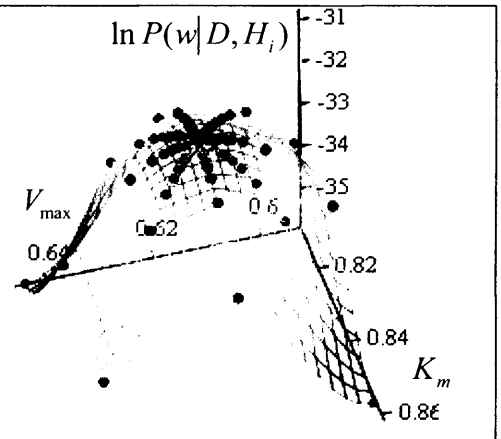


Figure 72. Posterior likelihood surface in the neighbourhood of the MAP estimate.

The Hessian summarises the curvature of the log likelihood function in the neighbourhood of the best-fit parameter values. The curvature measures how rapidly the likelihood changes as the parameter value changes, thus the greater the curvature the tighter the bounds on the w_{MAP} estimate. The parameter covariance matrix (V_w) can be obtained by taking the inverse of the negative Hessian. For this toy problem this is:

$$V_w \approx \begin{pmatrix} 2.164\times10^{-4} & 1.058\times10^{-5} & -6.621\times10^{-7} \\ 1.058\times10^{-5} & 2.973\times10^{-4} & 1.918\times10^{-6} \\ -6.621\times10^{-7} & 1.918\times10^{-6} & 3.749\times10^{-5} \end{pmatrix}$$

Having obtained V_w, a Gaussian with dimension equal to the number of parameters can then be fitted to the likelihood surface simply by specifying the location of the mean as the MAP parameter estimate w_{MAP} and the covariance as the parameter covariance.

$$p(w) = \frac{1}{2\pi \|V_w\|^{1/2}} \exp\left(-\frac{1}{2}(w - w_{MAP})^T V_w^{-1} (w - w_{MAP}) \right) \tag{8.20}$$

Since this is a normalised distribution the posterior probability $p(w)$ is a proper probability density function and so contours corresponding to different 'confidence levels' can be drawn. Figure 73 below shows pair-wise iso-probability contours for the toy problem.

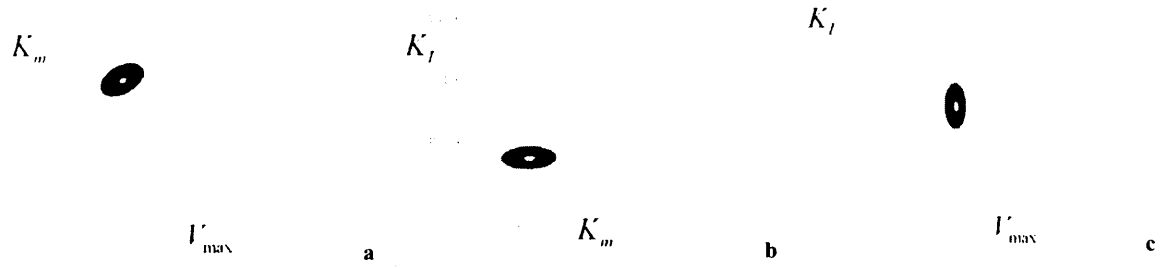


Figure 73. Iso-probability contours for the toy problem.

8.4 Marginalised predictions.

8.4.1 Theory of Marginalisation.

Parameter confidence intervals, such as calculated for the previous toy problem are useful where the parameters have a relevant physical meaning. However from a practical engineering standpoint quantifying the uncertainty with respect to the model predictions is more likely to be of interest. It is these model predictions rather than parameter estimates, which will be used to assist decision-making in the context of either optimisation or control.

In most cases the two most useful things to calculate are the expected value and the variance. The expected value is calculated by enumerating every possible outcome and weighing it by its likelihood, given everything known about the system.

Thus for a correct model H with uncertain parameters the expected value given some input is:

$$E\{H(x)\} = \hat{\Theta} = \int \underbrace{H(x,w)}_{\substack{\text{model prediction} \\ \text{given parameter} \\ \text{set } w \text{ and inputs } x}} \times \underbrace{P(w|D,H)}_{\substack{\text{posterior likelihood} \\ \text{of parameter set } w}} dw \tag{8.21}$$

Note that this is the expected value of a predicted quantity taking into account parameter uncertainty and not the same as predicting using the w_{MAP} parameter values.

The variance is given by:-

$$\sigma^2 = \int P(w|D,H)\left(H(x,w) - \hat{\Theta}\right)^2 dw \tag{8.22}$$

Equations (8.21) and (8.22) are complex multi-dimensional integrals over all the parameters in the model and so instead of attempting an analytical or quadrature solution a Monte Carlo method is proposed.

Monte Carlo method.

Monte Carlo methods compute the integral (8.21) by drawing N_s samples from $P(w|D,H)$ and then estimating the expected value from these samples as follows:-

$$E\{H(x)\} = \hat{\Theta} \simeq \frac{1}{N_s}\sum_{s=0}^{N_s-1} H(x,w^s)$$
$$\text{where} \tag{8.23}$$
$$w^s \sim P(w|D,H)$$

Samples cannot be directly drawn from the desired distribution, not least because the normalising constant $P(D|H_i)$ from equation (8.8) is unknown. So, samples are drawn using the Metropolis-Hastings method (Hastings(1970); Metropolis et al(1953)). This is a Monte Carlo sampling method typically used for high dimensional distributions. It must be emphasised that it is not a method for estimating the normalising constant of the distribution.

The method works by constructing a Markov chain the state space of which is the parameters of system with an equilibrium distribution that is the distribution of interest. This can be accomplished by ensuring the Markov chain satisfies a detailed balance:

$$T\left(w^a, w^b\right)P(w^b|D,H) = T\left(w^b, w^a\right)P(w^a|D,H) \qquad \forall w^a, w^b \qquad (8.24)$$

Equation (8.24) states that the probability of drawing[53] state w^b from the target density $P(w|D,H)$ and then making a transition $T\left(w',w\right)$ to state w^a is exactly the same as the probability of selecting state w^a and then making a transition to state w^b.

The Metropolis-Hastings method produces a Markov chain with the required properties by including a rejection step which compares the probability of the current state w_t and a proposed new state w'.

```
metropolis (p(w),N_s,w)
w^0 = w
s = 0
do
    w' = draw(Q(w'|w^s))
    a = [P(w')/P(w^s)] [Q(w^s|w')/Q(w'|w^s)]
    if((a ≥ 1) ∨ (rnd(1) ≤ a))
        accept
        w^(s+1) = w'
        s++
    else
        reject
while(s ≤ N_s)
return w
```

[53] By magic. Clearly we cannot directly draw from the distribution otherwise MCMC would not be required.

The above pseudo code generates a Markov chain consisting of N_s dependant samples from the distribution $p(w)$. For any positive $Q(w', w)$ the probability distribution of w' tends to $p(w)$ as $s \to \infty$.

Neither the proposal or the target distribution needs to be normalised since z (the normalising constant $z = P(D|H_i) = \int P(D|w,H)P(w|H)dw$) cancels out in the ratio.

As an illustration, Figure 74 shows 100 steps of the Metropolis algorithm starting at point (0,0) and beginning to converge to a 2D Gaussian distribution. This period during which the Markov chain is converging to the distribution of interest is known as a *'burn in'* period. It is only once convergence has been achieved that samples can be drawn from the distribution of interest therefore the first n_s samples are discarded. A lower bound on the number of iterations required for convergence can be calculated as:

$$n_s \simeq \left(L/\varepsilon\right)^2 \tag{8.25}$$

where L is the length scale of the distribution and ε the length scale of the proposal distribution. Equation (8.25) gives a rough estimate of the number of steps required for the random walk to traverse the distribution of interest.

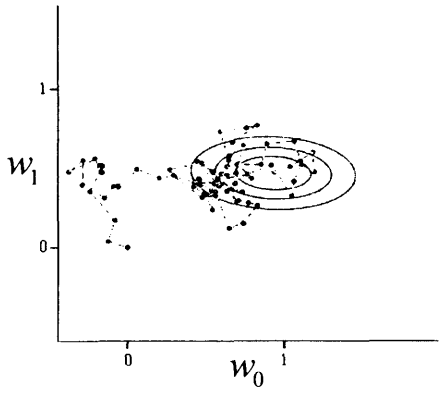


Figure 74. Example of samples generated by metropolis algorithm. Contours are for distribution of interest, points are 100 steps of the metropolis algorithm.

8.4.2 Demonstration: Marginalised prediction of a mechanistic model with uncertain parameters.

Training[54] data is generated from the test system introduced at the start of this chapter with $\sigma = 0.1$ noise added to the samples. The sub-models and initial conditions used to generate the data are set out in Table 8.

	Table 8. Demonstration system 1.			
Reaction 1.	$r_1 = \frac{0.311 C_A}{2.049\left(1+\frac{C_B}{4.847}\right)+C_A}$			
Reaction 2.	$r_2 = 2.05 C_A C_B + 0.699 C_A^2 C_B + 0.775 C_A C_B^2$			
Training.	*Species*	*Batch* 1	*Batch* 2	*Batch* 3
	C_a	4.639	2.761	0.438
	C_b	1.018	1.894	1.253
	C_c	3.834	3.136	4.413
	C_d	0.133	0.395	3.266
Testing.	*Species*	*Batch* 1	*Batch* 2	*Batch* 3
	C_a	2.248	3.202	4.203
	C_b	2.08	1.95	3.983
	C_c	3.11	0.452	3.148
	C_d	4.964	1.886	2.762

[54] Since nothing is actually trained on the data it is perhaps a misnomer to refer to the *data available for inference* as *'training'* data. However, since it fills a role similar to that of *'training data'* in previous sections we shall refer to it as such. Similarly plots of the marginalized model predictions vs data will be referred to as fits even though they are no such thing but rather *'probability distributions conditioned on the data'*.

The model structure was assumed to be correctly known but the parameter values unknown.

$$\frac{d\xi}{dt} = \begin{pmatrix} -1 & -1 \\ 1 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} r_1(w,\xi) \\ r_2(w,\xi) \end{pmatrix}$$

where

$$r_1(w,\xi) = \frac{w_0\xi_0}{w_1\left(1+\frac{\xi_1}{w_2}\right)+\xi_2} \tag{8.26}$$

$$r_2(w,\xi)_2 = w_0\xi_0\xi_1 + w_1\xi_0{}^2\xi_1 + w_2\xi_0\xi_1{}^2$$

$$\xi = [C_a, C_b, C_c, C_d]^T$$

Uniform priors $U(0,5)$ were placed over the parameters. These priors are by definition correct since this is the distribution the parameters of the target system were drawn from. Marginalised predictions were made by sampling using Markov chain Monte Carlo with the proposal distribution tuned by hand in order to obtain approximately 50% rejections.

The marginalisation process produces a full probability distribution conditioned on the data and priors $P(\xi_t|D,H,\xi_0,t)$ at every time point. For clarity the probability distribution is summarised by a line representing the expected value and 'error bars' representing the standard variance. The 'fit' to training data is shown in Figure 75 and to testing data in Figure 76.

Figure 75. Demonstration system 1: Plots of Expected values and variance against training data. Points show training data. Line shows expected value of Bayesian model. Error bars show variance.

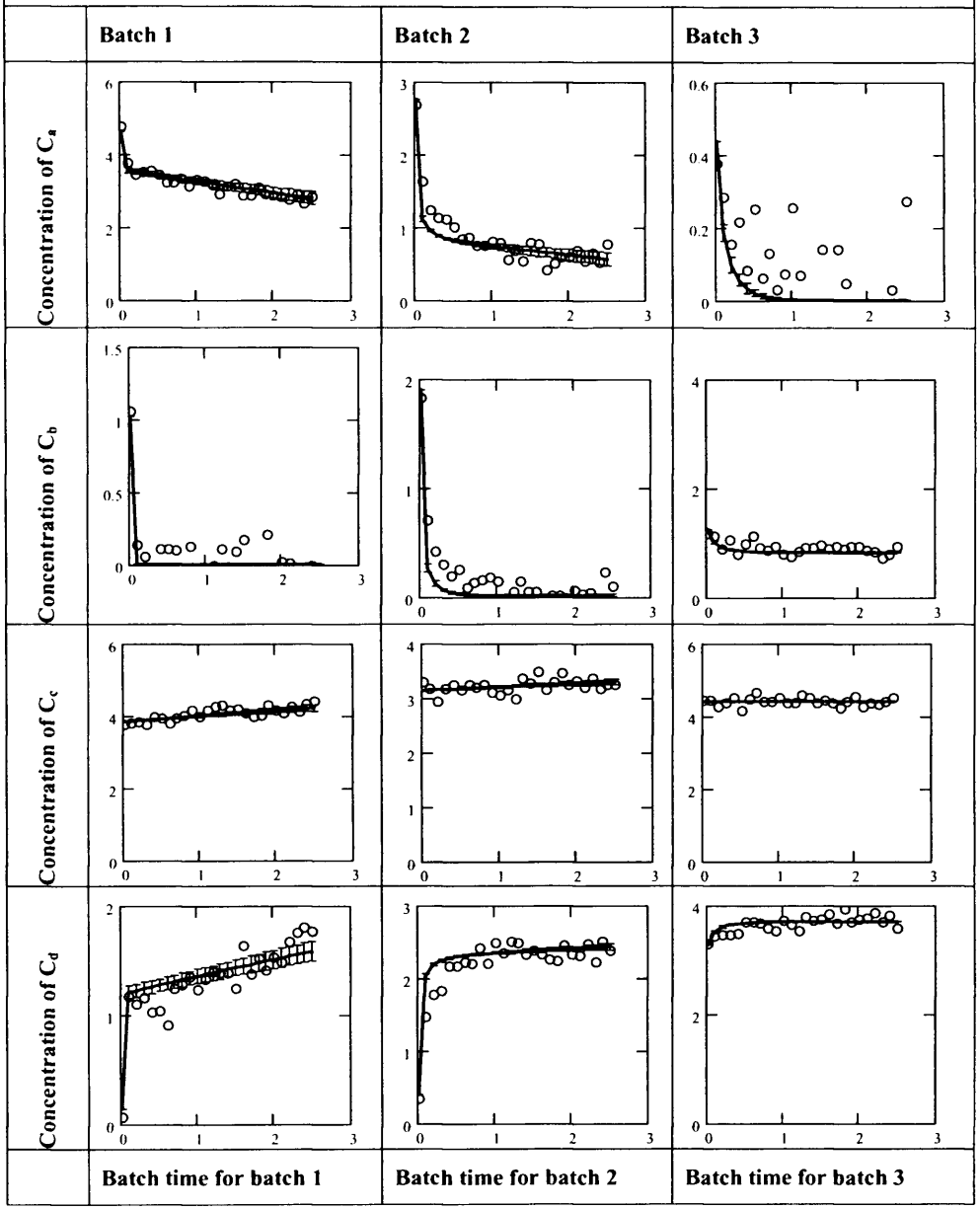
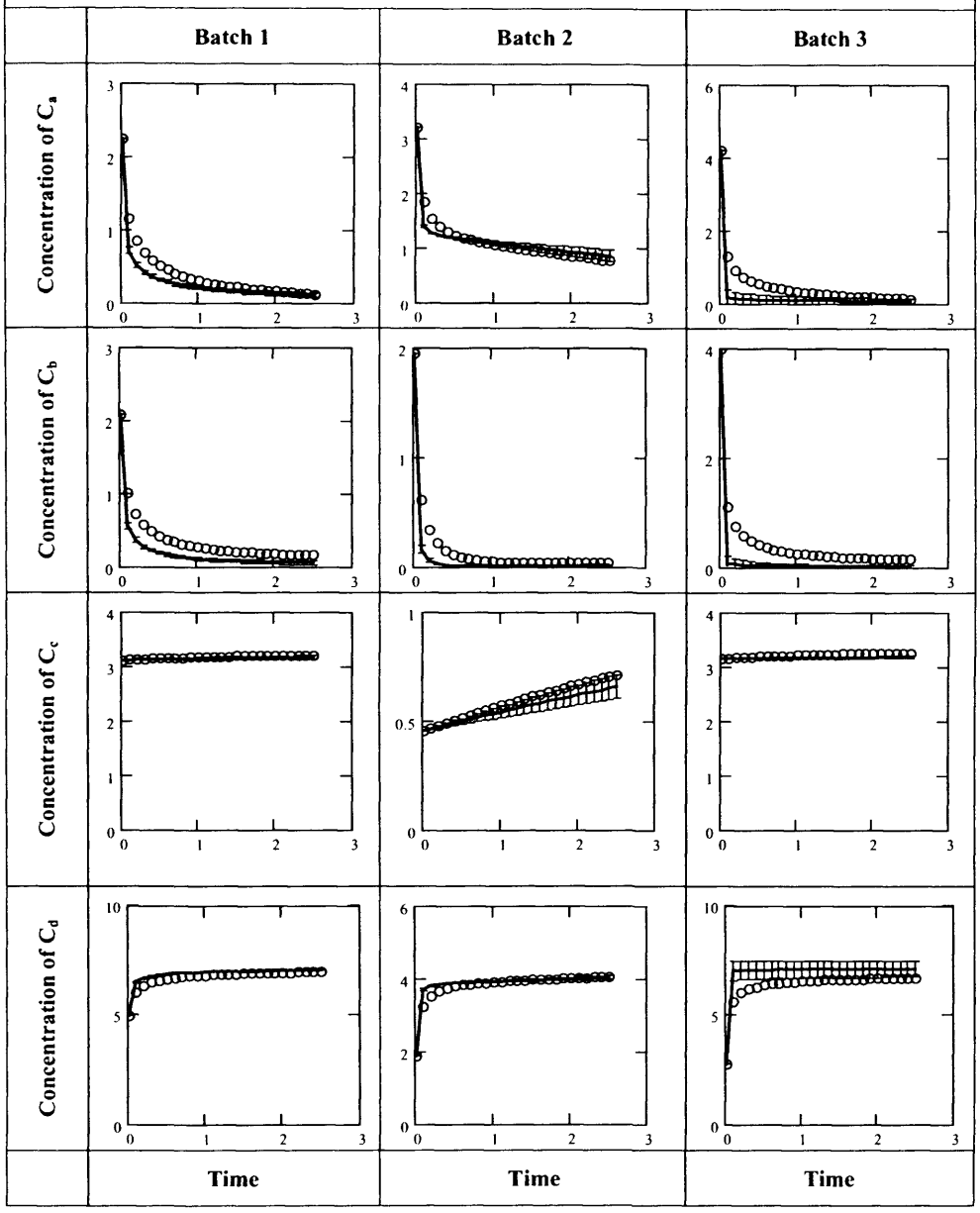


Figure 76. Demonstration system 1: Plots of Expected values and variance against testing data. Points show testing data. Line shows expected value of Bayesian model. Error bars show variance.



8.5 Marginalised Neural network predictions.

8.5.1 Theoretical basis

In many cases the correct model structure will not be known. In these situations it is perfectly possible to replace 'mechanistic' kinetic sub models with some parametric function capable of universal approximation, such as a neural network, and then marginalize over the parameters of that function in the same way as before.

Recall the definition of a feed forward neural network from section 7. For a single hidden layer network with one output, the network output can be written as:-

$$\begin{aligned} \underset{NN}{f}(x,w) &= \theta^{(1)} + \sum_{i=0}^{N_{neurons}-1} w_i^{(1)} \sigma\left(\theta_i^{(0)} + \sum_{j}^{N_{inputs}-1} w_{i,j}{}^{(0)} x_j\right) \\ w &= \left[w^{(0)}, w^{(1)}, \theta^{(0)}, \theta^{(1)}\right] \end{aligned} \tag{8.27}$$

The parameter vector is split into 4 parts: $w^{(0)} \in \Re^{N_{neurons} \times N_{inputs}}$ is a matrix of weights connecting nodes in the input to nodes hidden layer; $w^{(1)} \in \Re^{N_{neurons}}$ is a vector of weights connecting nodes in the hidden layer to the output node; $\theta^{(0)} \in \Re^{N_{neurons}}$ and $\theta^{(1)} \in \Re$ are bias terms applied to the nodes of the hidden and output layer respectively.

To formulate the problem in Bayesian terms prior distributions need to be defined for the network weights. However, unlike mechanistic equations the weights have no specific meaning. One option followed by Neal(1992) is to design the priors to reduce over fitting by penalising large weights;-

$$P(w) = (2\pi s^2)^{-\frac{N_w}{2}} \mathrm{e}^{-\left(\frac{|w|}{2s^2}\right)} \tag{8.28}$$

Where N_w is the length of the parameter vector and s is the expected scale of the weights. Over fitting is prevented, both by the use of the regularising prior and by the act of marginalisation.

8.5.2 Simple example: fitting a two dimensional function.

Training data consisting of $\ell = 100$ points at random input values $x \in \Re^2 \sim U\begin{pmatrix}-10,10\\-10,10\end{pmatrix}$ were generated from an unknown function $y = f(x)$ with $N(\mu = 0, \sigma = 0.1)$ added noise. A neural network with 10 nodes in the hidden layer

was then marginalised using Markov chain Monte Carlo method drawing from the posterior likelihood.

The unnormalised posterior probability of the parameters can be obtained by combining equation (8.10) for the likelihood and equation (8.28) for the prior:-

$$P(w|D) = \underbrace{\left(\prod_{i=0}^{\ell-1} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\left(y^i - \underset{NN}{f}(x^i, w) \right)^2}{2\sigma^2} \right) \right)}_{\substack{\text{likelihood of parameters given the data} \\ P(D|w,H)}} \times \underbrace{\left((2\pi s^2)^{-\frac{N_w}{2}} e^{-\left(\frac{|w|}{2s^2}\right)} \right)}_{\substack{\text{Prior probability of parameters} \\ P(w)}} \quad (8.29)$$

The parameter length scale was set to 2 and the Metropolis proposal distribution was a symmetric 41 dimensional Gaussian ($\sigma = 0.025$). The method was run until the chain of accepted states was 5000 states long. The first 2000 accepted states were discarded, since there is no guarantee that the Markov chain is sampling from the distribution of interest during the initial burn in period. The remaining 3000 samples used for estimating the expected value and variance.

Results on simple problem.

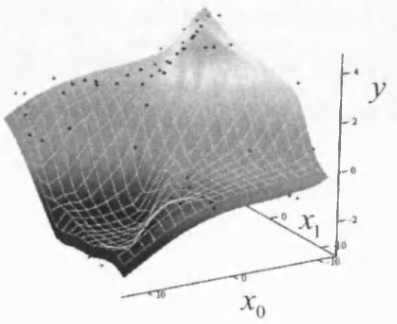


Figure 77. Fit of Bayesian neural network to simple problem. Points[55] are the training data. Filled surface is the expected value of the neural network prediction. The unfilled grid shows the standard variance of the prediction.

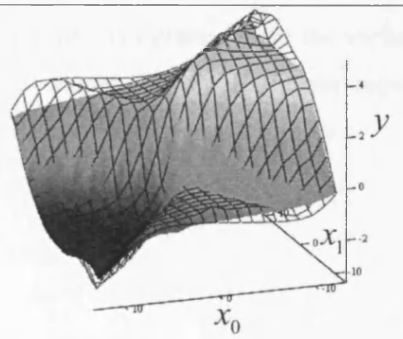


Figure 78. Prediction of Bayesian neural network The filled surface shows the neural network prediction. The grid shows the true function at 1000 validation points with no added error.

The neural network is clearly capable of fitting the data, as can be seen from Figure 77, which shows the fit to the training data and Figure 78, which shows fit to a surface generated from 1000 validation points.

[55] Note that some points are obscured by the filled surface.

The ability of the expected values of the marginalised neural network to fit the function without over fitting to the training data can be more clearly seen in the contour plots of the same validation data shown in Figure 79 and Figure 80 below. It is clear from these that the shape, if not the exact values predicted by the true function and the NN approximation are identical.

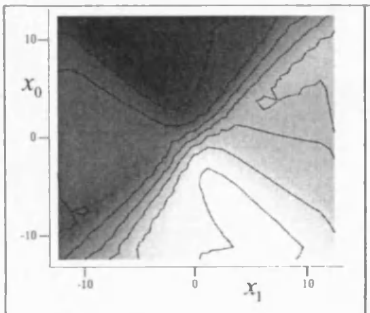


Figure 79. Contour plot of true system.

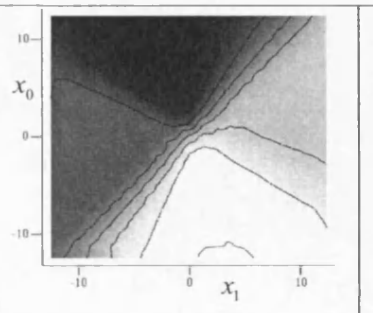


Figure 80. Contour plot of marginalised NN.

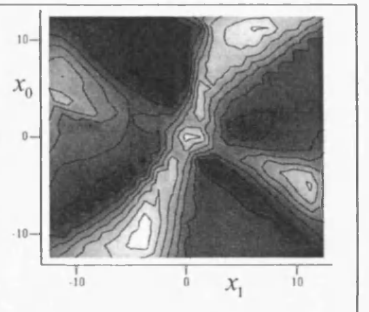


Figure 81. Contour plot of variance of NN estimate.

The variance of the estimate is shown in Figure 81. The variance of the estimate is due to both sensitivity of the model to parameter variance and the ability of information to restrict the parameter variance. The plot is lightest where the variance is highest, that is where the prediction is least certain. In this case these regions correspond to input conditions where the function value changes most rapidly.

8.5.3 Demonstration on simulated system.

The previous example of conditioning on data generated by demonstration system 1.was repeated but in the hybrid model the known equation for the r_2 was replaced with a neural network.

$$\frac{d\xi}{dt} = \begin{pmatrix} -1 & -1 \\ 1 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} r_1(w,\xi) \\ r_2(w,\xi) \end{pmatrix}$$

where

$$r_1(w,\xi) = \frac{w_0 \xi_0}{w_1\left(1 + \frac{\xi_1}{w_2}\right) + \xi_2} \tag{8.30}$$

$$r_2(w,\xi) = \underset{NN}{f}(\xi, w)$$

$$\xi = [C_a, C_b, C_c, C_d]^T$$

Uniform priors were assigned to the parameters of r_1 as before and regularising priors to the neural network parameters. The number of neurons in the hidden layer was set to 5. The 34 dimensional integration was accomplished by MCMC in this case requiring over 30,000 function evaluations, in order to produce 5000 samples. Since each function evaluation requires simulation of the model on all training batches the marginalisation process is computationally very expensive.

The 'fit' to training and testing data is summarised in Figure 82 and Figure 83 The 'fit' to training data is close to that achieved by the fully mechanistic model, the 'fit' to testing data is however markedly worse. Note that in both cases the variance is greater than that of the mechanistic model, reflecting the less certain nature of the predictions.

Figure 82. Demonstration system 1. Training data, expected values and variance.

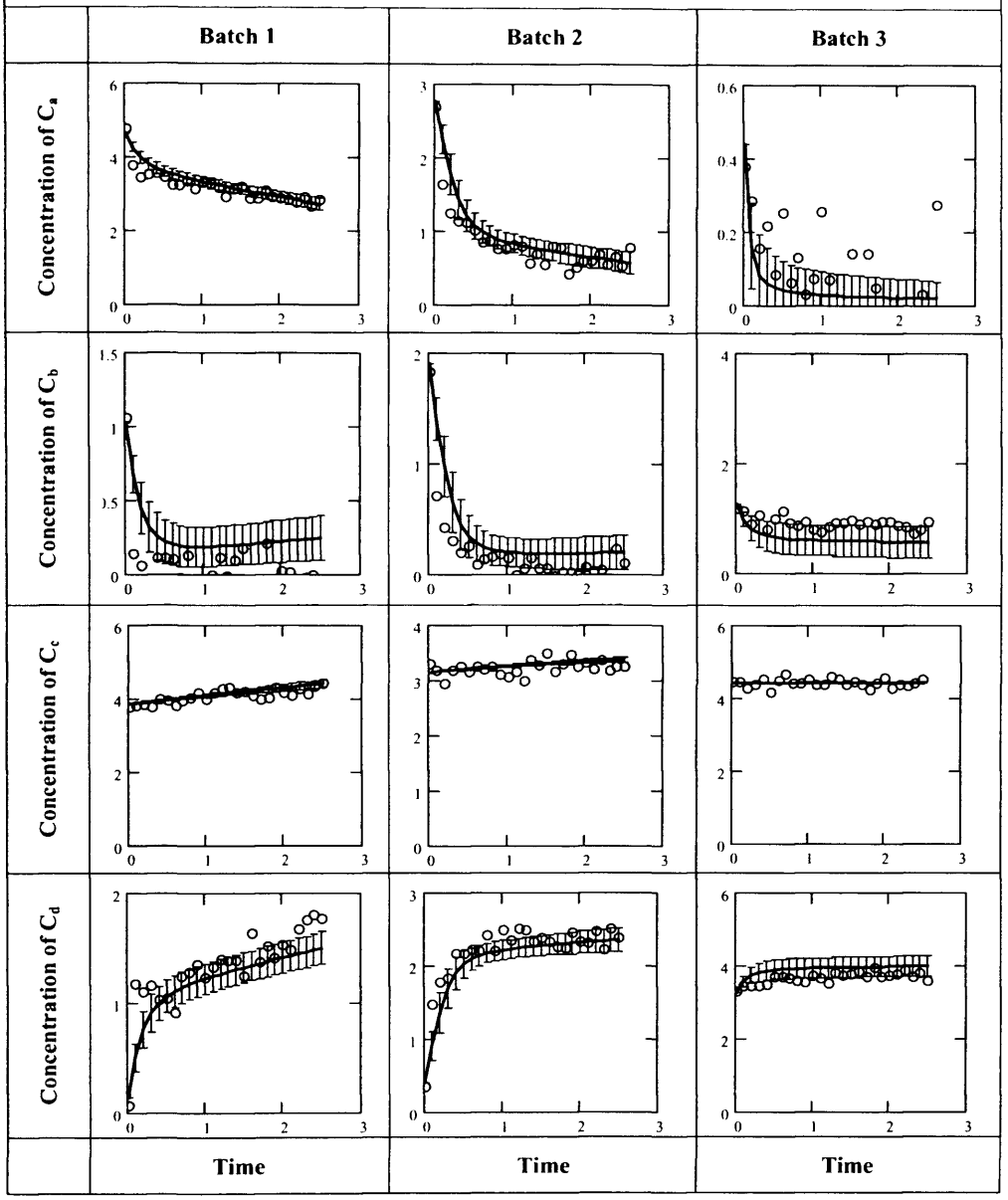
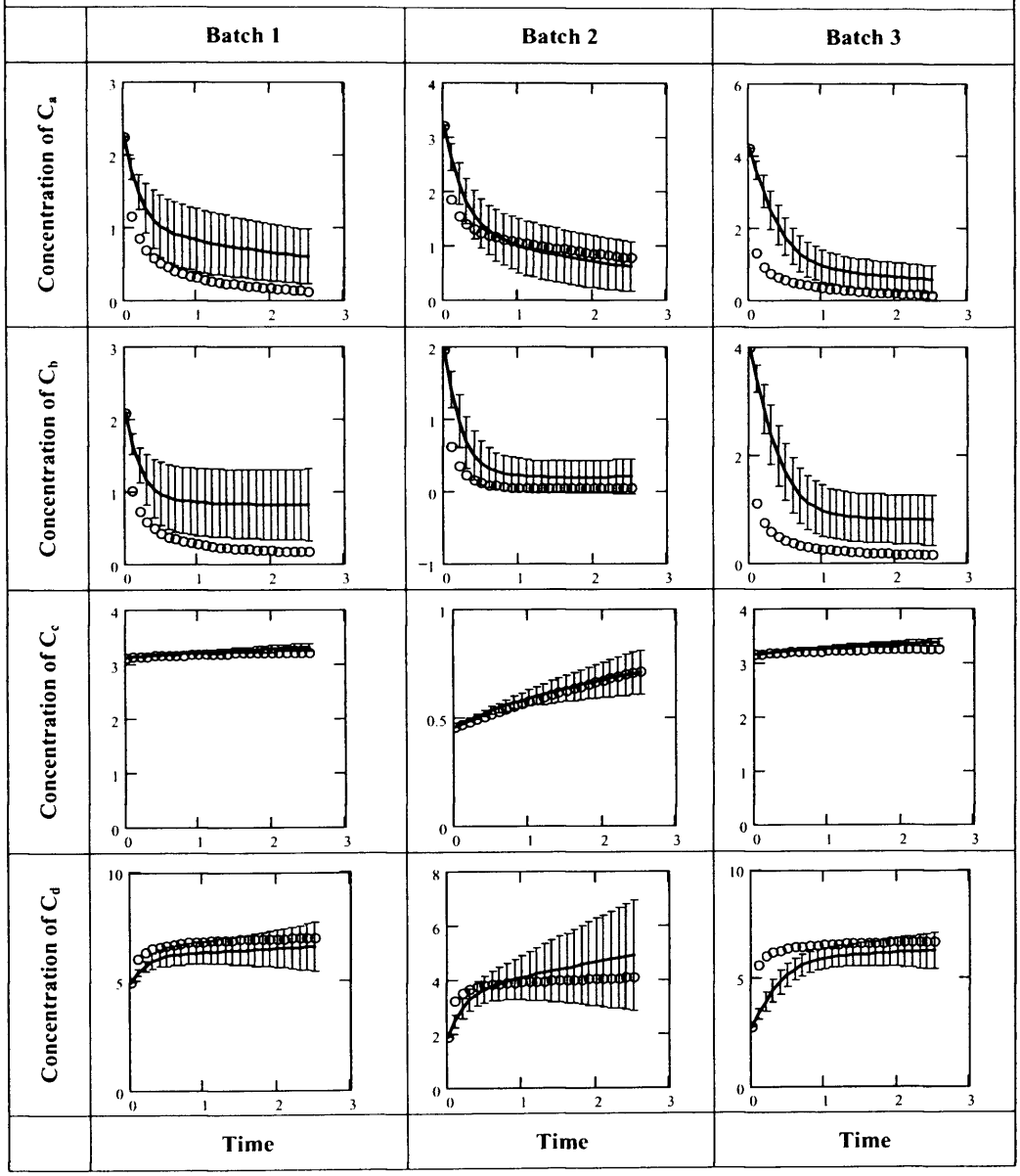


Figure 83. Demonstration system 1.Testing data, expected values and variance.



8.6 Coping with missing data.

8.6.1 Problem statement.

Consider the problem of building a hybrid model from four batches of data where measurements for one series (including its initial state) are completely missing from each batch of data.

Table 9. Available data series by batch.

Series	Batch 1	Batch 2	Batch 3	Batch 4
C_a	**Missing**	Measured	Measured	Measured
C_b	Measured	**Missing**	Measured	Measured
C_c	Measured	Measured	**Missing**	Measured
C_d	Measured	Measured	Measured	**Missing**

Because of the completely missing series it is impossible to obtain a continuous signal for the full state from this data. Hybrid models of the form outlined in section 5 cannot be trained on this data.

The situation is even more problematic because the initial conditions $\xi^*_{t=0}$ are incomplete: The model prediction given some parameter value $H(x,w)$ cannot be evaluated when x is unknown. Therefore the posterior probability $P(w|D,H)$ of the parameters cannot be found since the model prediction $H(x,w)$ must be known to evaluate the likelihood $P(D|w,x,H)$. This not only means that the methods described so far fail but that *any* method, based on the principle of running the model from defined initial conditions and comparing the fit to data, would also fail.

8.6.2 Approach.

Within the Bayesian philosophy any uncertain variable can be treated as a random variable described in terms of its probability density function. Moreover such random variables can be integrated out. The missing data problem can therefore be solved simply by marginalising with respect to the uncertainty in the initial conditions:

$$E\{H(x)\} = \hat{\Theta} = \int H(x,x_*,w) \times P(w,x_*|D,H) dw\, dx_* \qquad (8.31)$$

The probability density function for the joint likelihood of the parameters w and uncertain initial conditions x_* is given by the posterior likelihood as defined below, where $P(x_*)$ is the prior distribution in this case $U(0,5)$ over the uncertain variables:-

$$P(w, x_* | D, H) = \frac{P(D | w, x_*, H_i) P(w | H_i) P(x_*)}{P(D | H_i)} \tag{8.32}$$

The integral given by equation (8.31) can be evaluated by extending the state space of the Monte Carlo method to include the unknown initial conditions.

$$E\{H(x)\} = \hat{\Theta} \simeq \frac{1}{N_s} \sum_{s=0}^{N_s - 1} H(x, x_*^s, w^s)$$

where (8.33)

$$\left(w^s, x_*^s\right) \sim P(w, x_* | D, H)$$

8.6.3 Demonstration.

Table 10 Demonstration system 2.

Reaction 1	$r_1 = \frac{2.107C_A}{4.18 + C_A}$				
Reaction 2	$r_2 = \frac{3.774C_AC_B}{(0.735 + C_B)(1.789 + C_A)}$				
Training	*Species*	*Batch* 1	*Batch* 2	*Batch* 3	*Batch* 4
	C_a	3.917	2.599	4.38	4.779
	C_b	2.67	2.31	4.311	3.898
	C_c	4.984	3.057	1.331	4.201
	C_d	1.879	3.386	0.044	1.379
Testing	*Species*	*Batch* 1	*Batch* 2	*Batch* 3	
	C_a	3.428	3.714	2.571	
	C_b	3.286	0.571	1.428	
	C_c	2.857	0.023	1.256	
	C_d	1.861	0.857	2.714	

Both mechanistic and neural network models were 'trained' on data produced by the system defined in Table 10. Measurements for one series were removed from the data as outlined in Table 9. Fits to training data for both models are presented in Figure 84 and Figure 86. The fits to the true noise free missing series are also presented in the lightly shaded graphs. The accuracy of predictions on unseen testing batches are displayed in Figure 85 and Figure 87.

Figure 84. Demonstration system 2: Performance of mechanistic model on training data with missing measurements. Points show training data. Line shows expected value of Bayesian model. Error bars show variance. Grey graphs are for missing series.

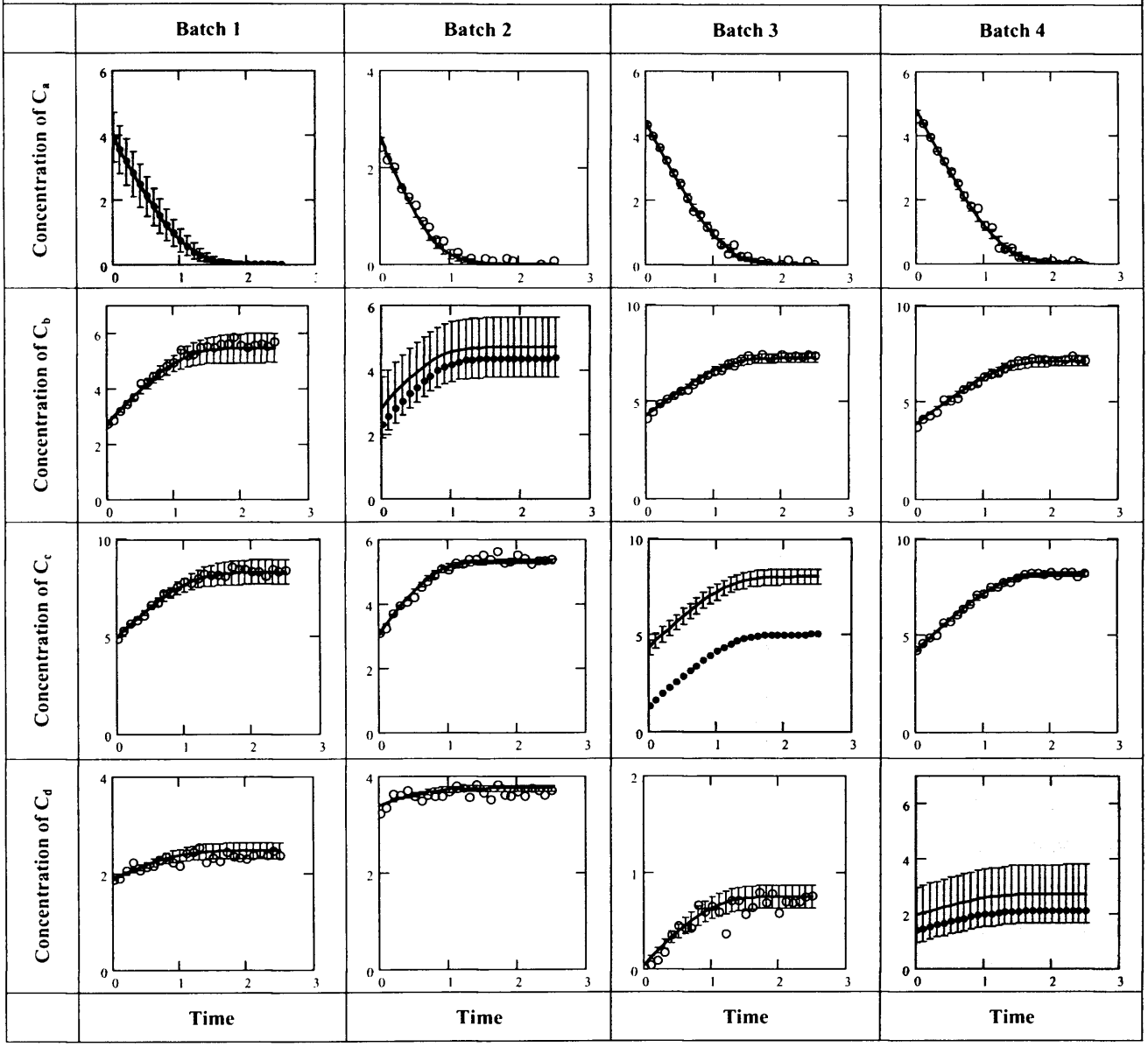


Figure 85. Demonstration system 2: Performance of mechanistic model on testing data. Points show testing data. Line shows expected value of Bayesian model. Error bars show variance.

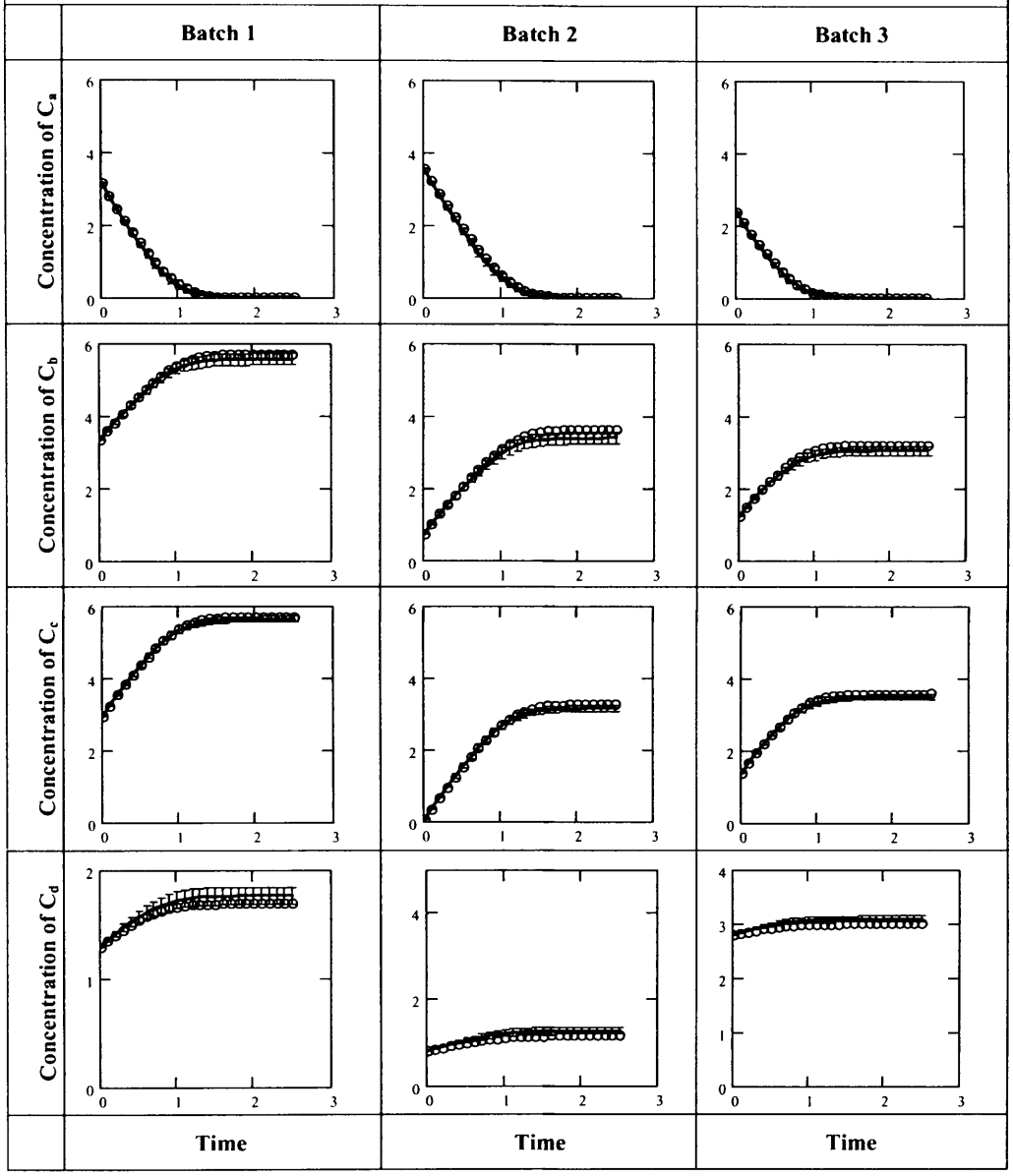


Figure 86. Demonstration system 2: Performance of neural network model on training data with missing measurements. Points show training data. Line shows expected value of Bayesian model. Error bars show variance. Grey graphs are for missing series.

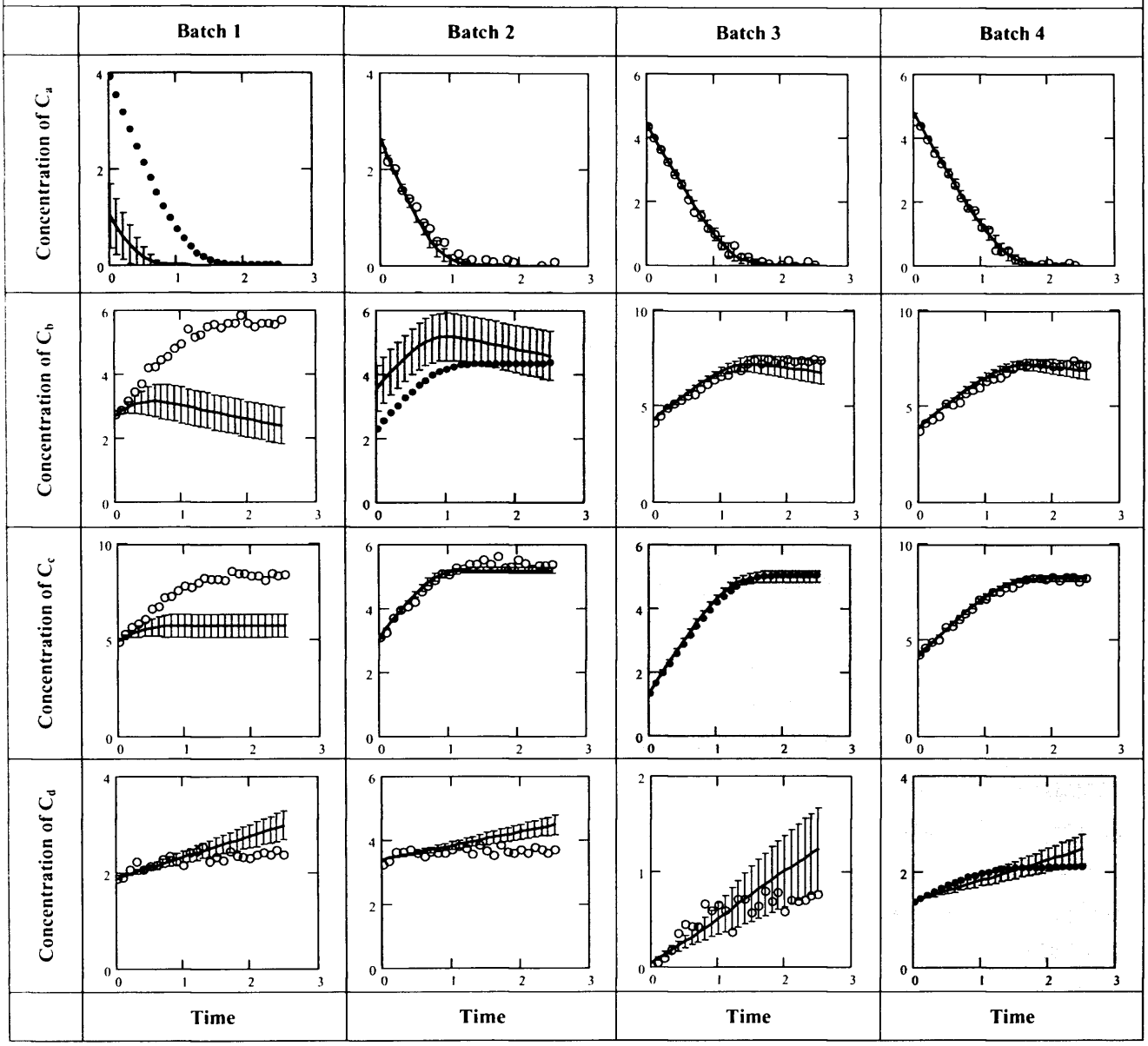
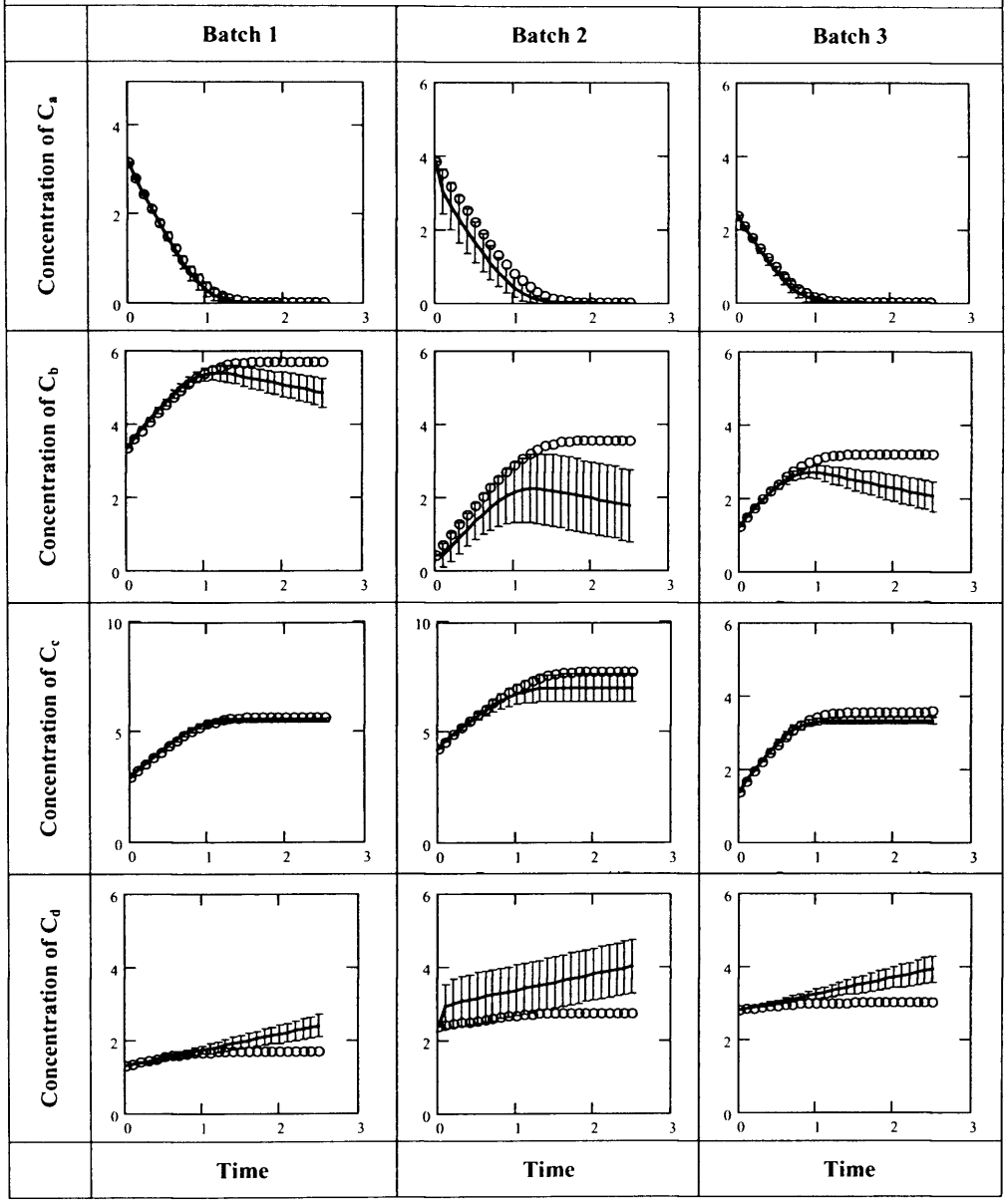


Figure 87. Demonstration system 2: Performance of neural network model on testing data. Points show testing data. Line shows expected value of Bayesian model. Error bars show variance.



8.6.4 Demonstration 3.

Table 11. Demonstration system 3.

Reaction 1	$r_1 = \frac{1.218C_A}{1.888\left(1+\frac{C_c}{1.908}\right)+C_A}$				
Reaction 2	$r_2 = \frac{3.737C_AC_B}{(1.53+C_B)(2.174+C_A)}$				
Training	*Species*	*Batch* 1	*Batch* 2	*Batch* 3	*Batch* 4
	C_a	1.3	0.321	1.722	4.499
	C_b	4.995	0.023	4.703	0.51
	C_c	0.577	3.09	2.792	3.335
	C_d	2.559	1.871	2.932	2.845
Testing	*Species*	*Batch* 1	*Batch* 2	*Batch* 3	
	C_a	2.141	1.623	1.627	
	C_b	3.276	4.32	4.495	
	C_c	2.549	3.57	1.874	
	C_d	4.194	3.454	3.036	

Both mechanistic and neural network models were 'trained' on data produced by the system defined in Table 11 with a different series missing from each batch as outlined in Table 8. Fits to training data for both models are presented in Figure 88 and Figure 90. The fits to the true noise free missing series are also presented in the lightly shaded graphs. The predictions on unseen testing batches are displayed in Figure 89 and Figure 91 respectively.

Figure 88. Demonstration system 3: Performance of mechanistic model on training data with missing measurements. Points show training data. Line shows expected value of Bayesian model. Error bars show variance. Grey graphs are for missing series.

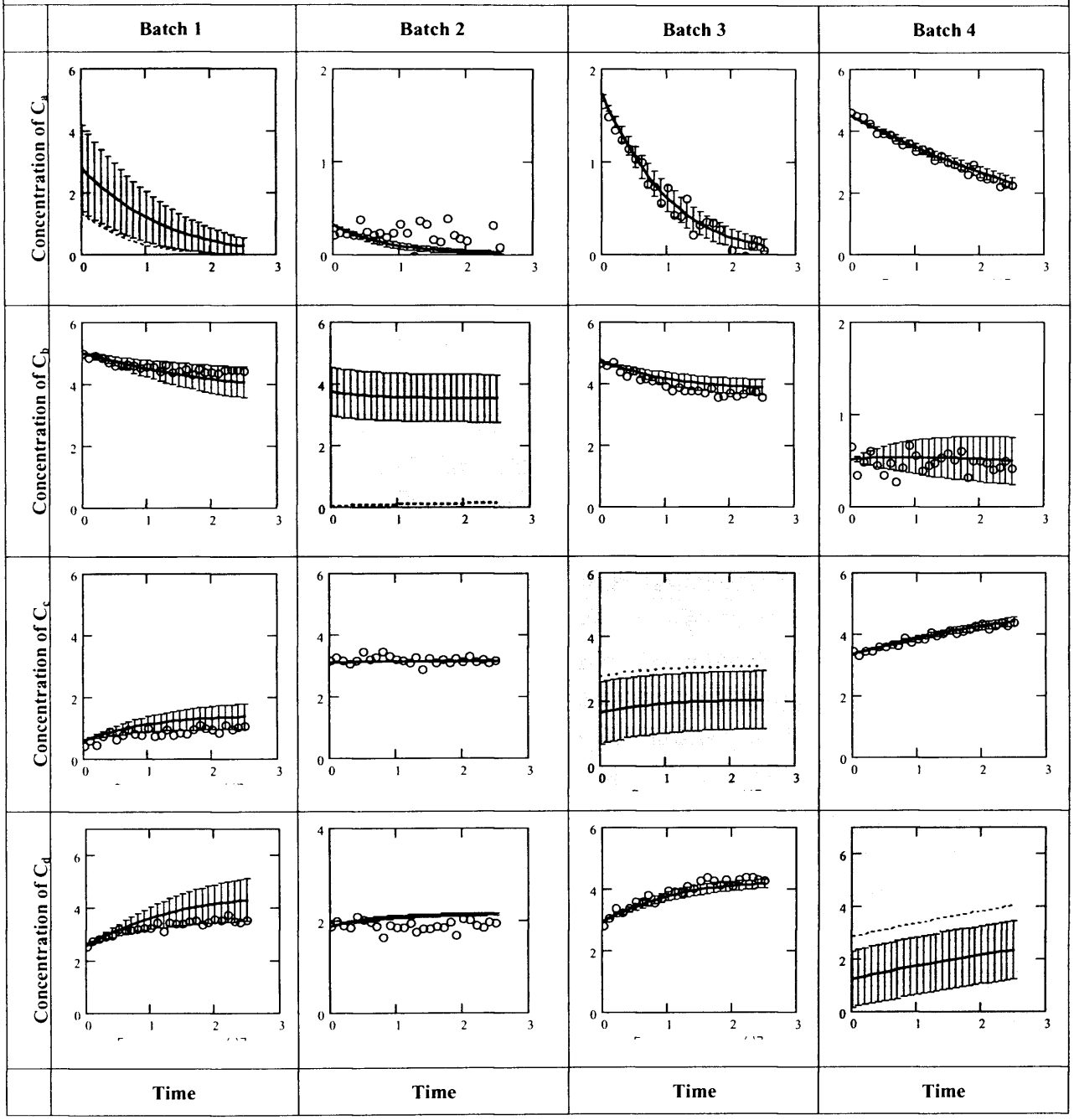


Figure 89. Demonstration system 3: Performance of mechanistic model on testing data. Points show testing data. Line shows expected value of Bayesian model. Error bars show variance.

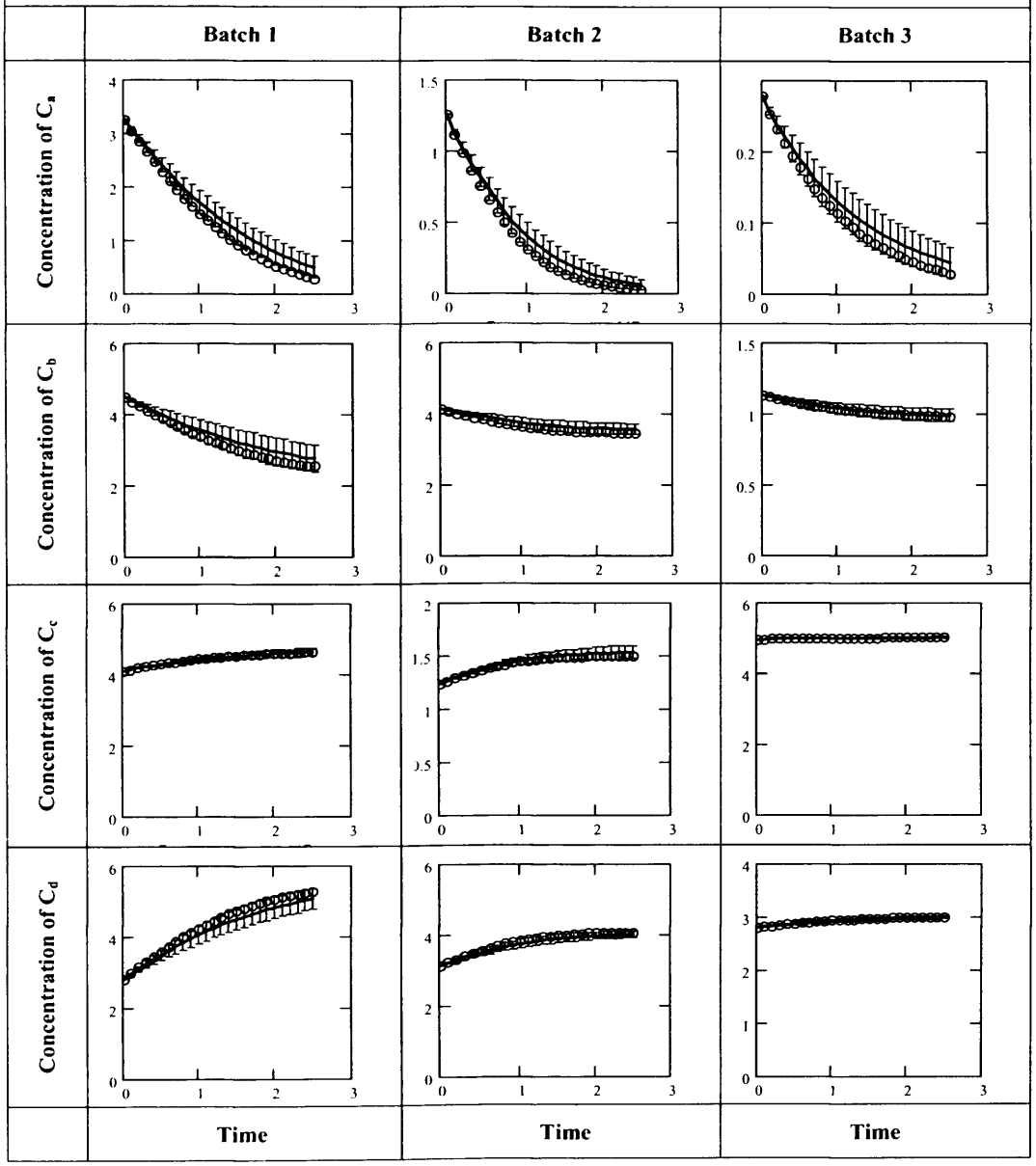


Figure 90. Demonstration system 2: Performance of neural network model on training data with missing measurements. Points show training data. Line shows expected value of Bayesian model. Error bars show variance. Grey graphs are for missing series.

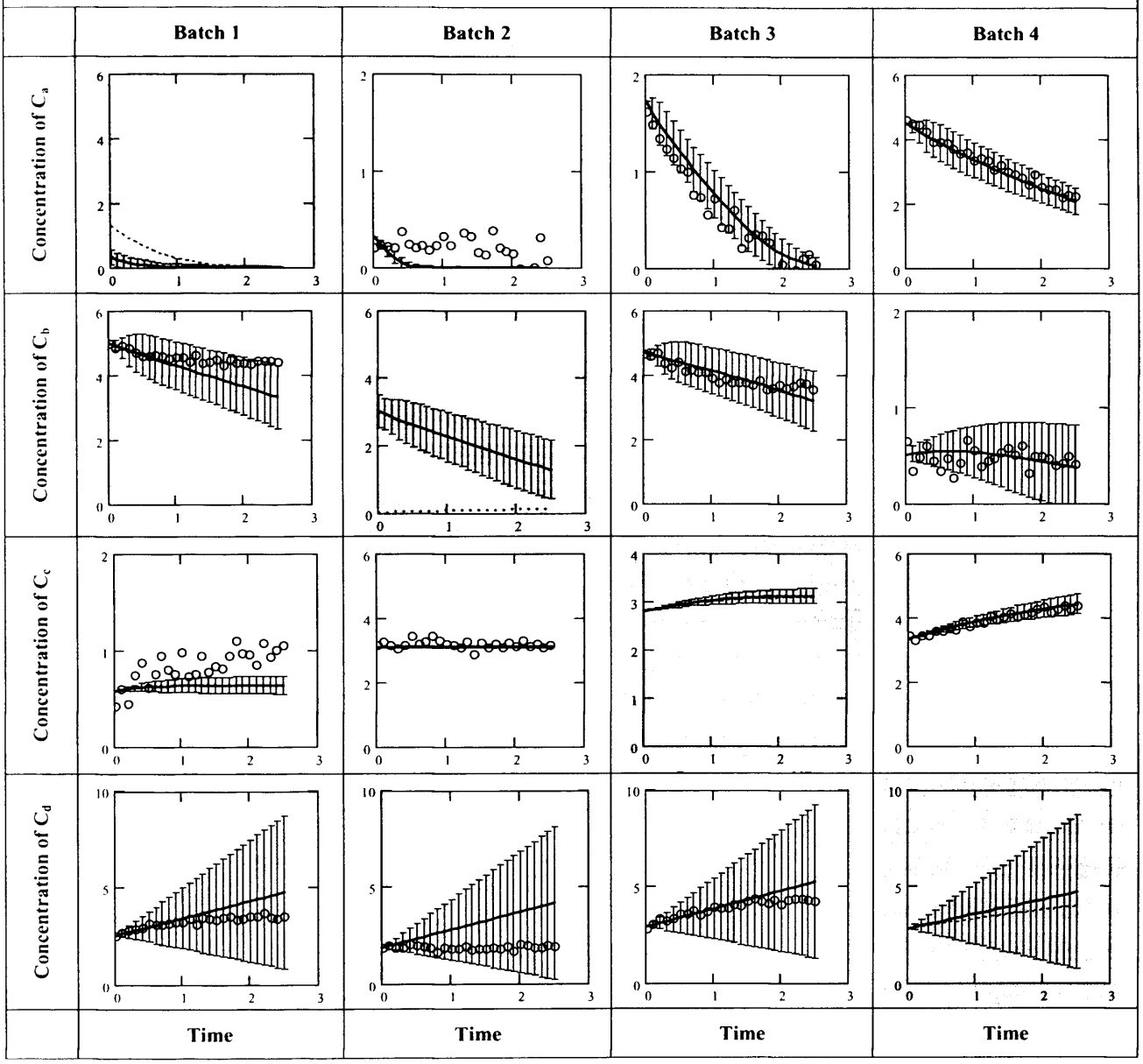
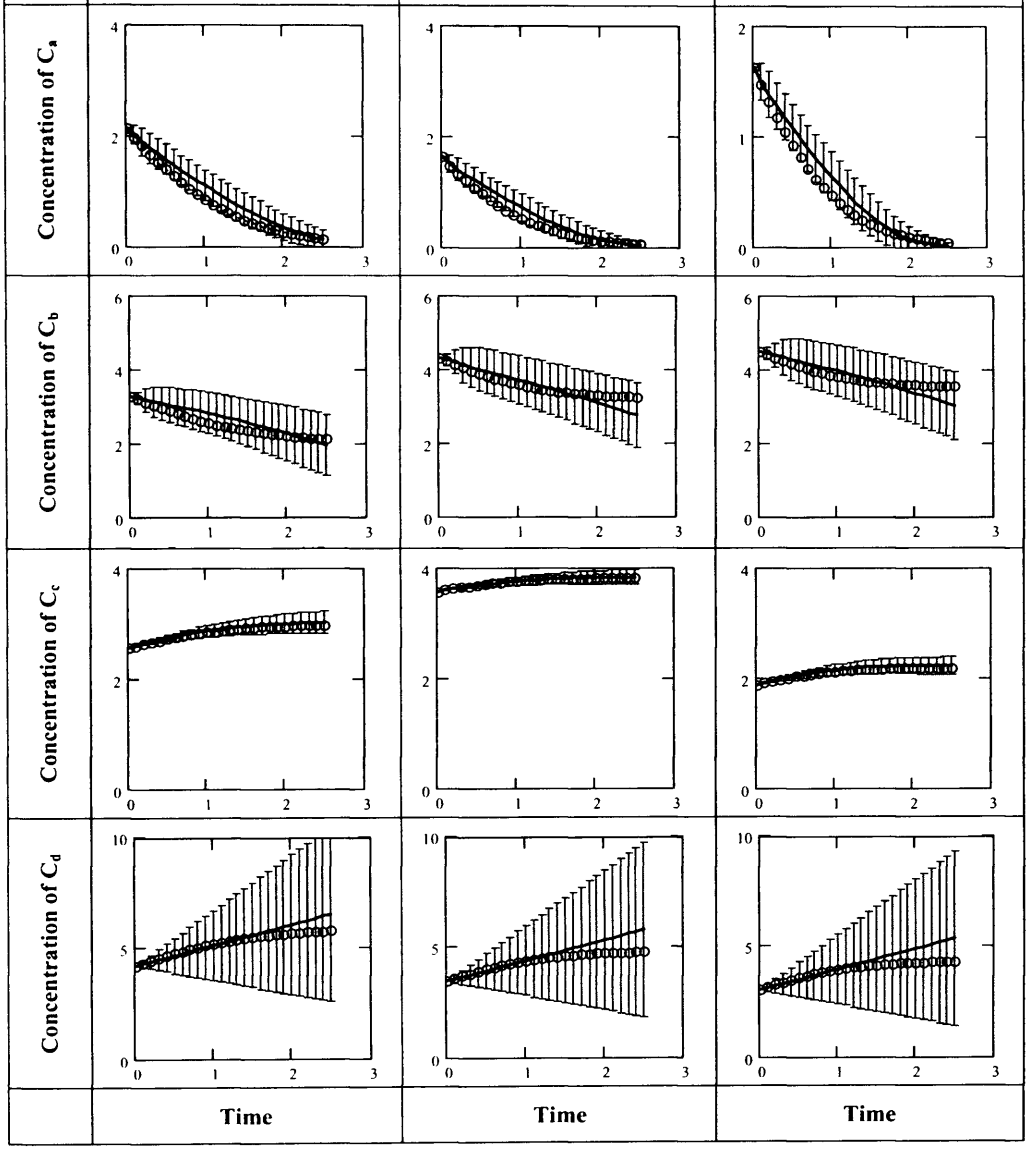


Figure 91. Demonstration system 2: Performance of neural network model on testing data. Points show testing data. Line shows expected value of Bayesian model. Error bars show variance.



8.6.5 Discussion

On both of the demonstration systems with missing series a similar picture emerges: The system with known kinetic equations manages to fit the measured series of the training data. It is also clear that the system has been correctly inferred from the available data since the model closely fits the testing data.

What makes this result significant is that in some, but not all cases, the missing initial conditions are recovered. This means that the Bayesian approach has managed to extract enough information from the data to recover the system despite continued uncertainty about the initial conditions.

The performance of the neural network model is markedly worse than that of the mechanistic model, particularly in fitting Batch one of the training data in both demonstration systems two and three. The greater uncertainty of the model compared with the mechanistic model is reflected in the variance of the estimate.

However, the point is not that the fits are good but rather that reasonable fits have been achieved by inferring a model from training data that would otherwise have been unusable.

8.7 Conclusion.

A Bayesian approach to hybrid modelling has been outlined and demonstrated on a number of simulated systems. It has been shown that the system allows for the incorporation of a priori knowledge and the construction of models containing a mixture of mechanistic and neural network models. The system makes probabilistic predictions rather than point predictions and is capable of making useful inferences from incomplete data sets. Moreover, if combined with decision theory the Bayesian approach can be seen as a step towards 'automatic engineering' since it provides a rational method for evaluating the expected utility of decisions.

The posterior distributions of Bayesian hybrid models, which include neural networks, are likely to be very complex and multimodal and thus difficult to sample from. As noted in the previous sections, neural network based models have many local optima, resulting in the poor performance of back propagation. Simple Markov chain methods, as used here, may become trapped in local modes and not sample from the full distribution. This may explain the slightly disappointing performance of the NN models. Thermodynamic integration techniques, such as *annealed importance sampling* introduced in the next section can sample from multi-modal distributions but are not without considerable computational cost.

9. Bayesian model selection.

In chapter eight a Bayesian approach modelling was outlined. The method required the model structure to be fixed although this structure could be any mixture of mechanistic equations and parametric functions such as neural networks. However, the Bayesian approach also provides a rational method for selecting the most probable model from a set of possible structures.

The computational challenges involved preclude fully implementing such a system with the resources available so, this section takes the form of a review with simple demonstrations.

Firstly, the issues involved in model selection are reviewed. Then, the problem of evaluating the 'evidence' for a particular model is considered. The ability of the annealed importance sampling method to evaluate the evidence for a mechanistic sub-model is then demonstrated on the simulated system introduced earlier.

9.1 Problem statement.

Consider the problem of selecting the best model from a set of competing model structures. One evaluates how probable each model is given the data. The model that has the greatest probability is then chosen, (equation (9.1)). Alternatively one can marginalise with respect to structural uncertainty[56] (equation (9.2)).

$$\arg\max_i P(H_i|D) \tag{9.1}$$

$$p(y,x|D) = \sum_i H_i(y|x,D)P(H_i|D) \tag{9.2}$$

Where $H_i(y|x,D)$ is the marginal distribution of a single structure evaluated in the previous section and $P(H_i|D)$ is the posterior probability of the i^{th} model given by Bayes theorem as:

[56] This section will concentrate on evaluating the evidence for a model. See appendix E5 for a method for marginalizing over structural uncertainty without evaluating the evidence.

$$\overbrace{P(H_i|D)}^{\text{Posterior}} = \frac{\overbrace{P(D|H_i)}^{\text{Evidence for model i}} \overbrace{P(H_i)}^{\text{Prior probability of model i}}}{\underbrace{P(D)}_{\text{Probability of data}}} \tag{9.3}$$

The $P(D)$ is simply a normalising constant $\sum_i P(D|H_i)P(H_i)$ so that the posterior probabilities of the alternative models sum to 1. For the purposes of model selection this can be ignored since only the relative probabilities of the different models need to be compared.

The prior over the models $P(H_i)$ can be used to express beliefs about how plausible the competing models are before seeing the data. For now assume that equal priors are assigned to the different models.

This leaves the evidence $P(D|H_i)$. The evidence is the likelihood of the model given the data, not at a specific set of parameter values but rather averaged over the prior distribution of parameter values. It can be viewed as the probability that the model will generate the given data set using parameter values drawn from the prior.

$$P(D|H_i) = \int P(D|w, H_i)P(w|H_i)dw \tag{9.4}$$

As MacKay, D. J. C.(2004) notes the evidence automatically embodies 'Occam's Razor'. Figure 92 shows this for a one-dimensional case of three models of the same form $H_i(w_0) = w_0$ but with different prior distributions. The data D obtained is shown as a horizontal dotted line. The value of w_0 for each model corresponding to this measured data is shown as a vertical dotted line. The prior distributions for the parameter values of each model are shown as solid lines. Posterior distributions for the parameter values as dotted lines.

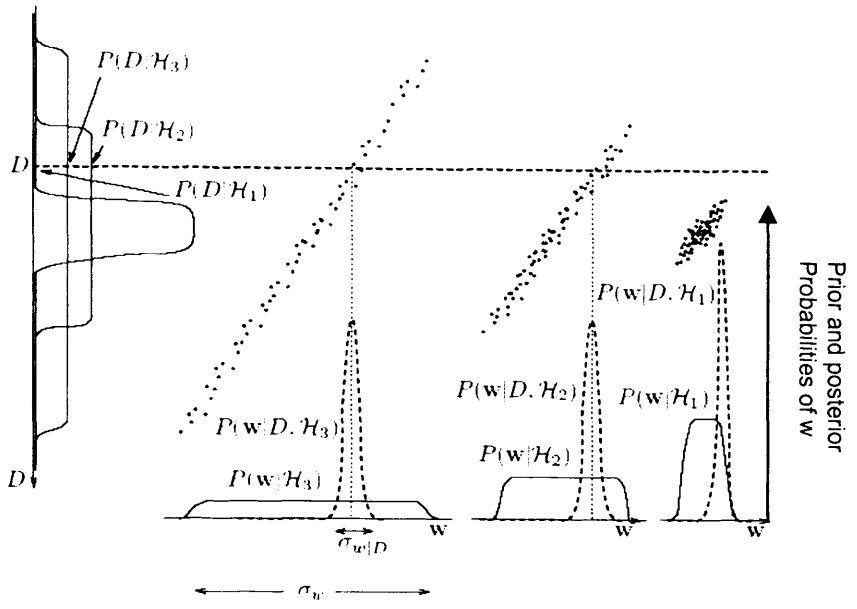


Figure 92. Illustration of Bayesian Occams Razor Taken from 'Information theory, Inference and learning algorithms' MacKay, D. J. C.(2004).

The prior distribution for the right most model H_1 is tightly defined. However predictions using parameters drawn from this prior are unlikely to be close to the observed data. The model makes specific predictions but those predictions do not match the observations. It is impossible for a value drawn from the prior $P(w|H_1)$ to fit the data and consequently there is no evidence for this model.

The prior distribution of the left most model H_3 is vague. The model can be made to fit the data but because the prior is uninformative the chance of drawing a parameter from the $P(w|H_3)$ prior, which fits the data, is low. The model can be made to fit the observations but the evidence is relatively low since it makes unspecific predictions[57].

Model H_2 makes relatively specific predictions, and parameter values with high prior $P(w|H_2)$ probabilities correspond to parameter values with high likelihood $P(D|w,H_2)$.

[57] The philosophically inclined reader may wish to consider how this links not only with Occums Razor but also Karl Poppers falsificationism.

Similarly, the more parameters a model has, the larger the volume of the prior parameter space. Hence, the less likely it is that a sample drawn from the prior will be at an area of high posterior probability.

9.2 Evaluating the evidence.

The Bayesian evidence provides a systematic and powerful method for model discrimination but unfortunately for most problems the integral given by (9.4) is analytically intractable and must be approximated in some way.

The normalising constant needs to be calculated, this is a more difficult problem than marginalising with respect to parameter uncertainty. It is not enough to simply draw samples from the distribution or indeed the prior. Possible methods for calculating the evidence are reviewed in the following section.

9.2.1 Laplace approximation.

If the posterior is unimodal; that is there is a strong peak in the posterior likelihood at w_{MAP}, the evidence can be approximated by the volume of this peak. This approach is known variously as the 'Laplace Approximation', the 'Saddle-Point Approximation' and the 'Evidence Approximation'.

The essential idea of the Laplace Approximation is to fit a multidimensional Gaussian to the maximum posterior peak, with covariance given by the 'error bars' for parameters using the procedure outlined previously for identification of the MAP parameter values. The volume can then be calculated analytically as the normalising constant of this Gaussian.

$$\underbrace{P(D|H_i)}_{evidence} \simeq \underbrace{P(D|w_{MAP},H_i)}_{Best\ fit\ likelihood} \times \underbrace{P(w_{MAP}|H_i)\sqrt{\frac{2\pi^{N_w}}{\det A}}}_{Occam\ factor} \qquad (9.5)$$

$$where\ A^{-1} = \left(\nabla\nabla \ln P(w|D,H_i)\big|_{w_{MAP}}\right)^{-1}$$

The above equation consists of two components; the best fit likelihood and an 'Occam factor', MacKay, D. J. C.(2004)). The 'Occam factor' is calculated from the Hessian and the prior probability of the parameters. It can be visualised as the ratio of the posterior accessible volume of the models parameter space to the prior accessible volume of the parameter space. Thus it penalises general models with many parameters, which require specific tuning. The 'Occam factor' has the following

properties which favour models with desirable characteristics with respect to the local best fit.

- It penalises powerful models with larger numbers of parameters since $P(w_{MAP}|H_i)$ will be lower the more free parameters.
- It favours models with informative priors, since $P(w_{MAP}|H_i)$ will be higher for such a model than a corresponding model with non-informative priors or where the priors turned out to be incorrect.
- It favours models, which are robust with respect to parameter uncertainty through the Hessian matrix.

Simple example.

The problem is to calculate the evidence using the Laplace approximation for $r_1 = w_0 C_A$, where the true system is $r_1 = 3.856C_a, r_2 = \dfrac{0.618C_A C_B}{(2.555+C_A)(3.824+C_B)}$.

A number of samples were taken in the neighbourhood of w_{MAP} and a polynomial fitted to the log likelihood at these points. This is shown in Figure 93 with circles being the log posterior and the line being the polynomial approximation $\ln P(w|D,H_0) \simeq -162.61+5.479w_0-0.727(w_0)^2$.

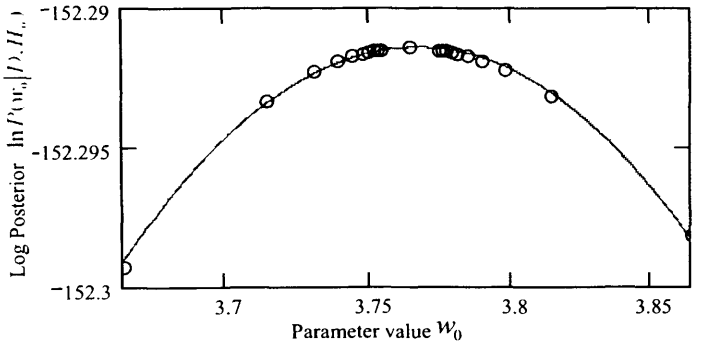


Figure 93. Fitting a polynomial approximation to the Posterior likelihood surface in neighbourhood of MAP parameter estimate.

The second derivatives can be found analytically from the coefficients of the polynomial $-\nabla\nabla \ln P(w|D,H_0) \cong 1.454$ and hence the normalising constant obtained.

$$\ln P(D|H_i) \simeq -152.291+\ln\sqrt{\frac{2\pi}{1.454}} = -150.213 \qquad (9.6)$$

Figure 94 shows the true posterior (solid line) against the Laplace approximation (dotted line). The Laplace approximation assumes unbounded variables and so because the curvature is slight and w_{MAP} not too far from the boundary the evidence is over estimated. The true log evidence in this case being -151.736.

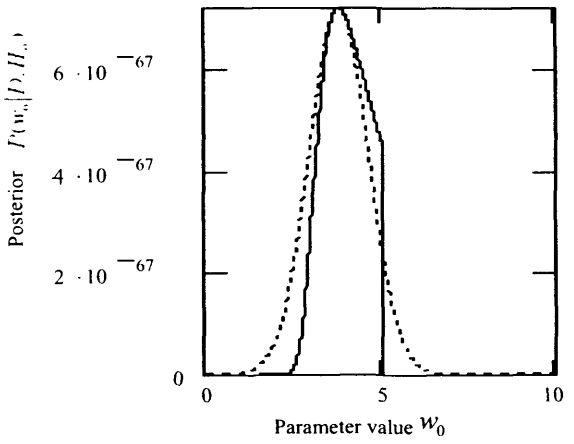


Figure 94. Laplace approximation (dotted line) against true posterior probability(solid line).

The Laplace approximation can only be applied if the Hessian $A^{-0.5}$ can be calculated. This requires A to be non-singular and positive definite. The reason for this is a meaningful parameter variance matrix can only exist if there is negative curvature in all directions indicating a strict maximum has been found.

In practice this may not to be the case for one of the following reasons:

- The mode is at a plateau or the model is collinear with respect to some parameters. (If there is zero curvature we have no information in which case the matrix is singular and the variance is not defined since 1/0 is not defined)
- The mode is at a constraint boundary.
- The posterior surface in the neighbourhood of the mode is a ridge or saddle point.

9.2.2 The Bayesian information criteria.

A simple approximation for the evidence can be derived from the Laplace approximation. The Laplace approximation in terms of logs is as follows;

$$\underbrace{\ln P(D|H_i)}_{evidence} \simeq \underbrace{\ln P(D|w_{MP},H_i)+\ln P(w_{MP}|H_i)}_{posterior\ likelihood\ at\ MAP} + \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln\det\left(I(D:w_{MP},H_i)\right) \tag{9.7}$$

For large amounts of data the last term can be written following Schwarz(1978)as:-

$$-\frac{1}{2}\ln\det\left(I(D:w_{MP},H_i)\right) \simeq -\frac{k}{2}\ln N - \frac{1}{2}\det\left(NE_{x_1:w_{MP}}\left[I(x_1:w_{MP},H_i)\right]\right) \tag{9.8}$$

hence:

$$\underbrace{\ln P(D|H_i)}_{evidence} \simeq \ln P(D|w_{MP},H_i)+\sum_{n=1}^{N}\ln P(t_n|w_{MP},H_i)\ldots + \frac{d}{2}\ln 2\pi + \frac{k}{2}\ln N - \frac{1}{2}\ln\det\left(E\left[I(D:w_{MP},H_i)\right]\right) \tag{9.9}$$

As the number of data points $N \to \infty$ the terms containing N become dominant leading to the classical and easy to evaluate Bayesian information criteria(BIC):-

$$BIC = \ln P(D|w_{MP},H_i) - \frac{1}{2}k\ln(N) \tag{9.10}$$

The BIC, unlike other simple methods such as the Akaike Information Criterion, (Akaike(1973)), does not favour more complex models as more data N is collected. The BIC depends solely on the number of parameters k and maximum likelihood. It assumes equal priors on each model and vague priors on the parameters. More over the sensitivity of the likelihood to parameter values is not included.

Remark on the Bayesian information criteria.

As a matter of principle the evidence for a model should be independent of the language used to describe it. It is therefore of concern that we are able to arbitrarily change the evidence of a model by writing it differently or introducing redundant parameters. This *"language independence"* principle may seem counter intuitive since we typically view complexity as related to the number of parameters or variables. But if two models written in different ways give identical predictions on all possible data sets then it should not be possible to distinguish them since they are identical. See Solomonoff(1997) for a related discussion.

It is also clear that for the assumptions behind the BIC to be valid the Laplace approximation must also be valid. Therefore, from the point of view of accurately approximating the evidence there is no reason to favour the BIC over the Laplace approximation. However, its computational simplicity is very attractive for practical usage.

9.2.3 Importance sampling.

Theory of importance sampling.

Consider the evaluation of the marginal likelihood by the integral

$$P(D|H_i) = \int P(D|w,H_i)P(w|H_i)dw. \tag{9.11}$$

Monte Carlo draws from the prior $P(w|H_i)$ create an estimator, which converges to the evidence as $N_s \to \infty$

$$P(D|H_i) \simeq \hat{z} = \frac{1}{N_s}\sum_{s=0}^{N_s-1} P(D|w^s,H_i) \quad \text{where } w^s \sim P(w|H_i) \tag{9.12}$$

However if the prior distribution does not match areas where $P(D|w,H_i)$ is large, then the estimator will have large variance and will be very slow to converge. Moreover, it may be difficult to sample directly from the prior distribution.

The idea behind importance sampling is to draw samples from some fully known *'importance sampling distribution'*, with probability density function is denoted $q(w)$, and then correct the estimate to take into account the fact samples were drawn from the wrong distribution.

$$\hat{z} = \int P(D|w,H_i)\frac{P(w|H_i)}{q(w)}q(w)dw \tag{9.13}$$

If a sample w^s is drawn from $q(w)$ at a point where $q(w^s) > P(w^s|H_i)$ then those samples will to be over-represented in the estimator. Conversely, samples where $q(w^s) < P(w^s|H_i)$ will be under-represented in the estimator. Importance sampling adjusts the weight given to each sample point in the estimator by the ratio of the two distributions.

$$\hat{z} = \frac{1}{N_s}\sum_{s=0}^{N_s-1} P(D|w^s, H_i)\frac{P(w^s|H_i)}{q(w^s)} \qquad (9.14)$$
$$where\ w^s \sim q(w)$$

If $q(w) > 0$ whenever $P(w|H_i) > 0$ then (9.14) converges to the true evidence as $N_s \to \infty$. If this condition is not satisfied some areas of the distribution will never be sampled however many samples are taken.

A key problem with the importance sampling estimator is that it can become dominated by infrequent draws from points where $q(w'') \gg P(w''|H_i)$ since the importance weight at these points is huge. Particularly in large dimensions this can result in the estimator having high variance. The variance of the estimator is given by

$$\mathrm{var}\{\hat{z}\} = \frac{1}{N_s}\left(\int \frac{P(D|w^s, H_i)^2 P(w^s|H_i)^2}{q(w^s)} dw - \hat{z}^2\right) \qquad (9.15)$$

Rasmussen and Ghahramani(2003) state that by calculus of variations it can be shown that the optimal importance sampling distribution $q^*(w)$ is

$$q^*(w) = \frac{|P(D|w, H_i)|P(w|H_i)}{\int |P(D|w, H_i)|P(w|H_i) dw}. \qquad (9.16)$$

The optimal importance sampling distribution is the normalised version of the distribution we are trying to normalise! Clearly using such an optimal distribution is impossible, but it does provide a rationale for the choice of importance distribution.

If the Laplace approximation were correct then a multivariate normal, centred over the MAP mode and with variance given by the inverse of the negative Hessian matrix, would be exactly the true distribution and therefore would be the optimal importance sampling distribution.

For distributions with a single mode, but where the Laplace approximation fails, samples around w_{MAP} can still be used to construct an importance distribution close to optimal as defined in equation (9.16).

This importance sampling distribution will be in the form of a multivariate normal and therefore not only is its normalising constant known but samples can easily be drawn from it by the following procedure:

Drawing from a multivariate normal .

Kaiser and Dickman(1962) give the following method for generating a vector of correlated variates $w^s = \left(w_0, \ldots, w_{N_w-1}\right)^T$ with entries drawn from a multivariate Gaussian distribution with mean $\mu \in \Re^{N_w}$ and covariance matrix $V_w \in \Re^{N_w \times N_w}$.

Method.

First calculate the Cholesky factorisation of the covariance matrix $V_w = R^T R$. Then draw N_w variates $z^s = \left(z_0, \ldots, z_{N_w-1}\right)^T$ from the standard normal distribution[58] $N\left(\mu = 0, \sigma = 1\right)$. A draw w^s from the desired distribution can then be obtained by

$$w^s = \mu_w + R^T z^s . \tag{9.17}$$

This means that within each sample the nth variate is given by the following expression:-

$$\begin{aligned} w_0 &= \mu_1 + R_{1,1} z_1 \\ w_1 &= \mu_2 + R_{1,2} z_1 + R_{2,2} z_2 \\ &\vdots \\ w_n &= \mu_n + R_{1,n} z_1 + R_{2n,2} z_2 \cdots + R_{n,n} z_n \end{aligned} \tag{9.18}$$

The covariance matrix of the samples has elements

$$\begin{aligned} C_{i,j} &= \left(w_i - \mu_i\right)\left(w_j - \mu_j\right) \\ C_{i,j} &= \left(\sum_k R_{k,i} z_k\right)\left(\sum_l R_{l,j} z_l\right). \\ C_{i,j} &= \left(\sum_{k,l} R_{k,i} R_{l,j} \operatorname{cov}(z_k z_l)\right) \end{aligned} \tag{9.19}$$

Which since $\operatorname{cov}(z_k z_l) = 1$ means that the samples have the desired covariance structure since:

[58] This can be easily accomplished by the Box-Muller transform Knuth(1981).

$$C_{i,j} = \sum_{k,l} R_{k,i} R_{l,j} = (V_w)_{i,j}. \tag{9.20}$$

Notice that draws are only possible if V_w is symmetric positive definite in which case the Laplace approximation would not have failed. In this situation importance acts simply as a correction for parameter boundaries.

In order to apply the 'optimal importance sampling' method to cases where the Laplace approximation fails the requirement that $V_w = A^{-1} = \left(\nabla\nabla \ln P(w|D, H_i)|_{w_{MP}}\right)^{-1}$ needs to be relaxed. Instead of the true variance a pseudo variance matrix, which is as close as possible to the inverse of the negative Hessian A^{-1} but is guaranteed to be positive definite, is used.

There are two techniques, which can be used to calculate such a pseudo variance matrix.

- The Moore-Penrose generalised inverse $A^{\#}$ can be found even when the inverse A^{-1} does not exist.
- V_w can be forced to be positive definite using a generalised Cholesky decomposition.

In doing this we are taking a similar approach to Gill.J. and King.G.(2004), however they use rejection sampling to estimate the mean and variance of the distribution of interest rather than importance sampling to estimate the normalising constant of the distribution of interest.

Cholesky decompositions compute the matrix square root R *where* $V_w = R^T R$, if V_w is not positive definite then this cannot be calculated. Generalised Cholesky decompositions find a non-negative diagonal matrix E such that $V_w + E$ is positive definite and the diagonal values of E are as small as possible.

The Schnabel and Eskow(1990) Cholesky factorisation was to calculate the matrix square root used since this algorithm usually produces a matrix E with smaller diagonal values than the alternative Gill et al(1981) algorithm. Source code for both methods can be found in Gill and King (2004) as well as appendix E2 and E3.

A 1 dimensional example.

Recall the one dimensional example used to demonstrate the Laplace approximation. Importance sampling can be used as a boundary correction of the evidence calculated in the previous section.

The importance distribution was defined by the Laplace approximation calculated previously. 200 draws from this distribution are displayed in histogram form in Figure 97 and the corresponding importance weights in Figure 96. It can be seen from Figure 95 that the importance sampling corrects the over estimate of the evidence by the Laplace approximation. The evidence calculated by importance sampling is -151.725 which is very close to the true value of -151.736 and a significant improvement on the Laplace and BIC estimates.

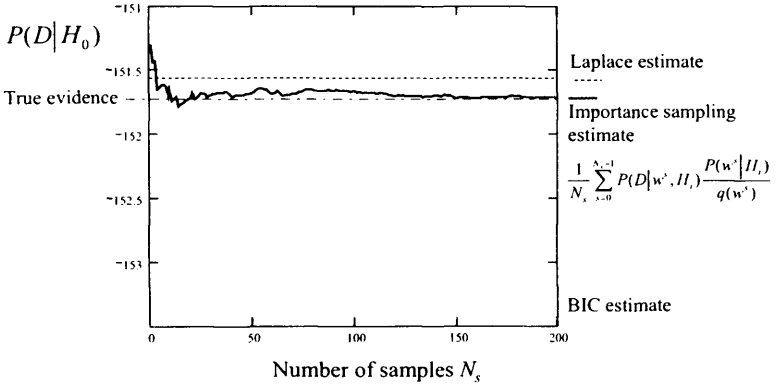


Figure 95. Comparison of 3 approximations to the true evidence.

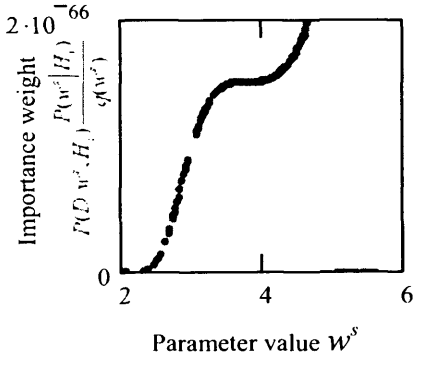


Figure 96. Importance weights.

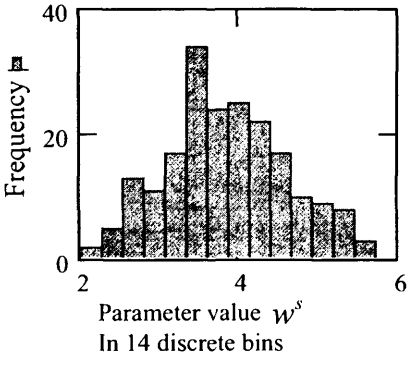


Figure 97. 200 draws from the importance distribution as a 14 'bin' histogram.

A 2 dimensional example.

Consider evaluating the evidence for a Michaelis Menten model with parameter priors $U(0,2)$ where the true system is first order with respect to the substrate.

Figure 98 shows a contour plot of the true likelihood surface. The mode for this model lies on the boundary . The Hessian matrix therefore cannot be positive definite and so the Laplace approximation fails. Figure 99 shows contours for importance sampling distributions. Lines show the importance sampling distribution defined by the pseudo variance matrix calculated by the Schnabel-Eskow generalised Cholesky decomposition of the Hessian. Filled contours show the importance sampling distribution defined by the pseudo variance matrix calculated by the Gill-Murray generalised Cholesky decomposition of the Hessian.

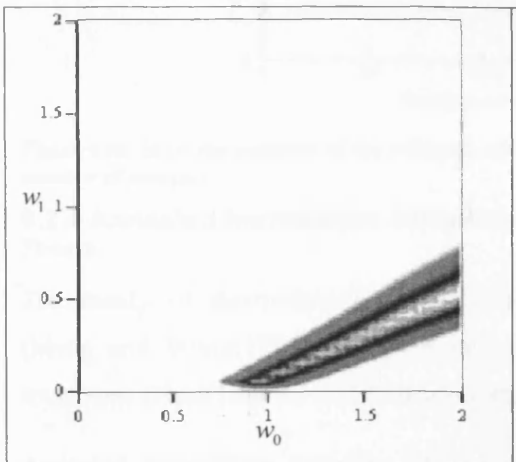


Figure 98. Contours of true posterior surface.

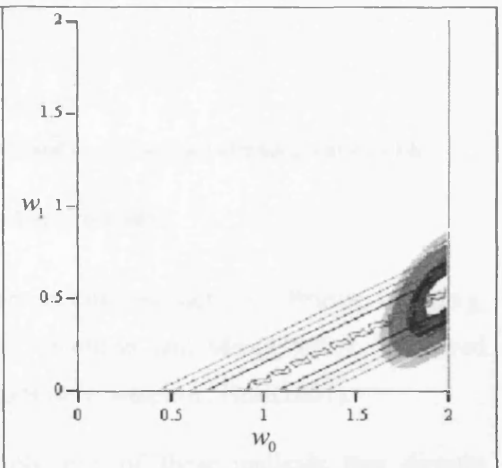


Figure 99. Contours of importance sampling distributions. Filled contours are for Gill-Murray based distribution. Unfilled contours are for Schnabel-Eskow based distribution.

Figure 100 shows the estimate of the evidence calculated by importance sampling. The grey dotted line shows the true evidence[59] it is clear that both importance-sampling methods decay to this value. It is also clear that the choice of importance distribution has a great effect on the convergence properties of the estimator, with the importance distribution converging within Gill-Murray 700 samples and the Schnabel-Eskow distribution taking 2000 samples. It is also clear that despite the importance distributions being good approximations to the true distribution, a large

[59] Calculated using a quadrature method, which is only possible because the distribution is only two-dimensional.

number of samples were required to estimate the evidence for a simple 2 dimensional distribution. This would suggest that it would not be possible to practically apply this method to 40-100 dimensional distributions unless they are well approximated by a single peak. Therefore simple importance sampling is unlikely to be applicable to realistic bioprocessing problems.

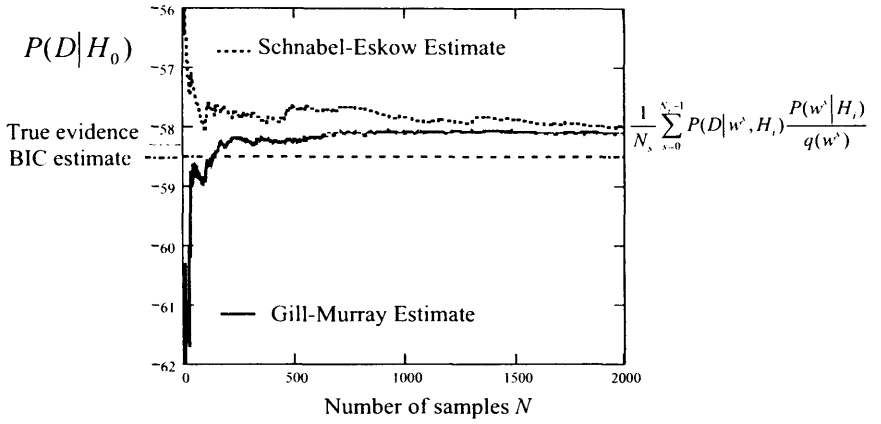


Figure 100. How the estimate of the evidence obtained by importance sampling varies with number of samples.

9.2.4 Annealed importance sampling techniques.

Theory.

The family of thermodynamic integration techniques includes: Bridge sampling, (Meng and Wong(1996)); Path sampling, (Gelman and Meng(1998)); Tempered transitions (Neal(1996b)) and Annealed importance sampling, Neal(2001).

Annealed importance sampling is the only one of these methods that directly evaluates the normalising constant. However, it should be noted that all of these techniques stem from the same central idea. This is to divide a difficult and perhaps multi-modal distribution into a series of easier ones parameterised by inverse 'temperature'.

Consider a difficult integral such as the posterior distribution

$$\int P(D|w,H_i)P(w|H_i)dw. \tag{9.21}$$

Define f_o as the distribution of interest $P(D|w,H_i)P(w|H_i)$ and f_n as some simple distribution from which it is possible to directly draw samples. For example, the prior density $P(w|H_i)$ would be a natural choice since its support is the support of the posterior. The idea is to construct a sequence of n distributions:-

$$f_j = f_0(w)^{\beta_j} f_n(w)^{1-\beta_j}$$
$$where\ 1 = \beta_0 > \beta_1 \ldots > \beta_{n-1} > \beta_n = 0 \tag{9.22}$$

so that as β_j approaches zero the distribution becomes flatter, more like $P(w|H_i)$ and therefore easier for a Markov chain to move around.

Code for AIS.

The following pseudo code is for a single annealing run generating a single sample from the distribution of interest and its corresponding importance weight. Updates of β can be by any method. Following Neal, R. M.(1996b) an arithmetic series was chosen where β<0.01 and a geometric series thereafter.

$\beta = 0$
$j = 0$
while$(\beta \le 1)$
  if $(\beta = 0)$ w=draw$(f_n(w))$
  else w_j= metropolis $(f_n(w)^{\beta} f_n(w)^{1-\beta_{j-1}}, w)_{j-1}$
  $\beta_j = \beta$
  $j++$
  β = update(β)
end while
sample = w

$$\text{log\_importance\_weight} = \sum_{j=0}^{length(\beta_j)-1} \ln \frac{f_n(w_j)^{\beta_j} f_n(w_j)^{1-\beta_j}}{f_n(w)^{\beta_{j-1}} f_n(w)^{1-\beta_{j-1}}}$$

$$return\ \underset{\text{sample}}{e^{\text{importance weight}}}$$

A 2 dimensional demonstration.

To illustrate the ability of AIS to draw samples from highly multi modal distributions consider using AIS to sample from a distribution consisting of a mixture of two-dimensional Gaussians.

$$f_0(w) = L(w) f_n(w)$$
$$L(w) = \sum_m \frac{1}{2\pi \begin{vmatrix} 0.05 & 0 \\ 0 & 0.05 \end{vmatrix}^{0.5}} \exp\left(-\frac{1}{2} \left(w - \begin{matrix} m \\ m \end{matrix} \right)^T \begin{vmatrix} 0.05 & 0 \\ 0 & 0.05 \end{vmatrix}^{-1} \left(w - \begin{matrix} m \\ m \end{matrix} \right) \right)$$
$$where \tag{9.23}$$
$$m \in \{-2, -1.0, 1, 2\}$$
$$f_n(w) = \begin{cases} \frac{1}{10}\frac{1}{10} & if\ (-5 \le w_0 \le 5) \wedge (-5 \le w_1 \le 5) \\ 0 & otherwise \end{cases}$$

The distribution is characterised by 5 separated peaks and thus would be hard to draw samples from by any simple MCMC method. However as Figure 101 shows 200 samples drawn using annealed importance sampling characterise all 5 peaks.

The complexity would also make it very difficult to find a good importance sampling distribution and thus calculating the normalising constant would be very difficult, yet as Figure 100 shows, AIS manages to calculate the normalising constant correctly within 200 samples.

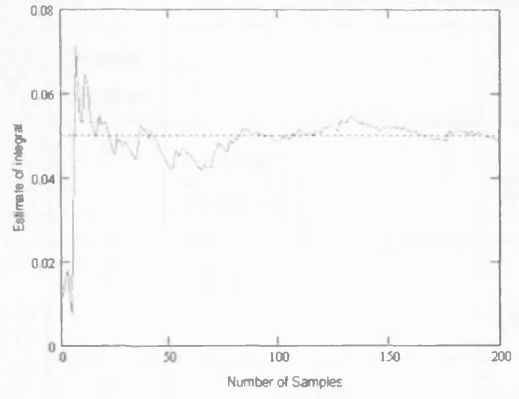


Figure 101. Estimate of integral as a function of number of samples.
True integral is shown as dotted line.

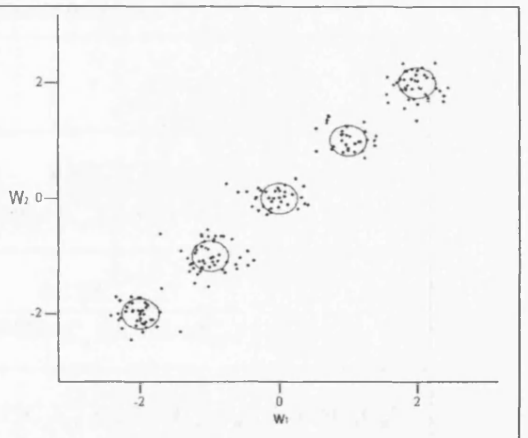


Figure 102. Samples drawn using AIS.
Points show 200 Samples drawn by AIS. Circles are contour lines showing position of the peaks of f_o.

The AIS method is clearly capable of handling complex multi-modal distributions. Neal, R. M.(2001) also notes that computational time scales linearly with the dimension of the problem. As such it would, in theory, be able to handle the distributions involved in hybrid modelling and allow both Bayesian model selection and marginalisation by sampling from the whole distribution.

Unfortunately the use of intermediate distributions means that there is a large constant increase in computation time. In the above simple example at least 2000 function evaluations were required to generate a single sample. Applying the technique to large hybrid models incorporating neural networks is beyond the computational capabilities of my personal laptop however since samples are independent it would be relatively easy to implement AIS in parallel on multiple networked computers.

9.3 A demonstration of Bayesian Model selection.

9.3.1 Problem definition.

Recall the two reaction system introduced at the start of chapter eight. Six test systems were defined where the kinetic models for each reaction are as defined in Table 12. The kinetic sub-model for r_2 is known and three batches of data generated from the system are available. The problem is to select the best model from the set of candidate models for r_1.

Table 12. Candidate kinetic functions.

System Number	r_1	r_2
1	$\frac{2.501C_A}{3.005+C_A}$	$\frac{1.85C_AC_B}{(1.096+C_A)(2.446+C_B)}$
2	$2.17C_A$	$\frac{0.128C_AC_B}{(0.943+C_A)(1.331+C_B)}$
3	$\frac{3.93C_A}{2.336\left(1+\frac{C_C}{3.379}\right)+C_A}$	$2.775C_AC_B+2.998C_A^2C_B+1.878C_AC_B^2$
4	$\frac{3.071C_A}{0.264\left(1+\frac{C_B}{3.488}\right)+C_A}$	$\frac{0.151C_AC_B}{(0592+C_A)(1.701+C_B)}$
5	$2.339C_A+2.575C_A^2$	$1.922C_AC_B$
6	$3.77C_A$	$\frac{3.144C_AC_B}{1.105C_A+1.337C_B+C_AC_B}$

9.4.2 Results and Discussion.

Table 13. Log evidence $\ln P(D|H_i)$ for each candidate model.

Candidate models	System Number >>	1	2	3	4	5	6
r_1	k_1C_A	-158	**-154**	-155	-195	-198	**-153**
	$k_1C_A + k_2C_A^2$	-165	-159	-162	-203	**-156**	-155
	$\frac{k_1C_A}{k_2 + C_A}$	**-156**	-206	-159	-162	-619	-161
	$\frac{k_1C_A}{k_2\left(1+\frac{C_B}{k_3}\right)+C_A}$	-160	-414	-154	**-158**	-355	-164
	$\frac{k_1C_A}{k_2\left(1+\frac{C_C}{k_3}\right)+C_A}$	-157	-503	**-154**	-160	-520	-182

Table 14. Posterior probability $P(H_i|D)$ for each candidate model.

Candidate models	System Number >>	1	2	3	4	5	6
r_1	k_1C_A	16%	**99%**	12%	0%	0%	**92%**
	$k_1C_A + k_2C_A^2$	0%	1%	0%	0%	**100%**	8%
	$\frac{k_1C_A}{k_2 + C_A}$	**53%**	0%	0%	2%	0%	0%
	$\frac{k_1C_A}{k_2\left(1+\frac{C_B}{k_3}\right)+C_A}$	2%	0%	35%	**86%**	0%	0%
	$\frac{k_1C_A}{k_2\left(1+\frac{C_C}{k_3}\right)+C_A}$	29%	0%	**52%**	12%	0%	0%

Table 13 shows the Log of the evidence for each model in the candidate set estimated from 400 samples drawn using the annealed importance sampling algorithm. Table 14 shows the normalised posterior probability of each model in the candidate set. The correct model for each test system is shown in bold.

In all cases AIS gives the highest probability to the correct model and thus the validity of the Bayesian approach to model discrimination has been confirmed on the simulated system.

However, there is a serious problem with this approach. In the above example, while it was possible to evaluate the marginal likelihood of the candidate models for r_1, the correct kinetic sub-model was used for r_2. In reality we would be ignorant of this. The dimensionality of the problem would be that of the whole hybrid model containing several reactions, rather than just one reaction as in the simple 3 dimensional problem demonstrated. Furthermore, to perform model selection every possible combination of sub-models would have to be considered. This is likely to be prohibitive for real problems with large numbers of reactions.

9.4.4 Conclusion.

It has been shown that not only does the Bayesian framework provide a principled framework for hybrid modelling but that it also provides a framework for model selection. The 'evidence' criterion for model selection was introduced and several methods for approximating the evidence outlined. The ability of evidence approximation to select the correct model from a complete candidate set was then demonstrated on 6 dynamical systems, although there are serious computational challenges which would need to be solved if this approach were to be used in industry.

Evaluating the evidence for a model is very slow using the outlined Monte Carlo methods. While the time is reasonable if the best model is to be selected from a small number of competing models it is not reasonable to use this technique for large multiple reaction systems since the evidence for every combination of sub models needs to be evaluated. Therefore while the Bayesian hybrid modelling approach in chapter eight looks promising. The Bayesian approach to model selection would appear to have limited applicability in a practical bioprocessing context.

10. Conclusion.

10.1 Thesis summary.

Model based strategies have the potential to speed up process development by reducing the number of real experiments required for process optimisation. The greatest benefit is realised during the later stages of bioprocess development when experiments are performed to optimise time varying control profiles in pilot scale bioreactors. Therefore a useful model should be capable of performing *'virtual experiments'* which predict the dynamic response of the system to initial conditions and control actions. A model with this capacity may be used operationally as the basis of model predictive control strategies or formulated as a 'state observer' using the extended/unscented Kalman filter formulation.

The use of such model-based strategies is hampered by the time consuming nature of bioprocess modelling. Building mechanistic models is extremely difficult due to the complexity and heterogeneous nature of biological systems and the consequent effort involved in determining the underlying mechanisms.

The approach employed in this thesis was thus characterised as providing methodologies for building models of bioprocesses from data within a frame work which allows *a priori* knowledge to be incorporated if available. This 'serial hybrid' approach can be characterised as building models in the following form (detailed in section 1.3.1).

$$\frac{d\xi}{dt} = \underbrace{K}_{constraints} \times \underbrace{r(\xi, v)}_{kinetics} - \underbrace{D\xi - g(\xi) + F\xi_{in}}_{transport} \tag{9.24}$$

A mass balance over the reactor, defined by the transport term, explicitly handles the effect of known actions such as feeding, sampling, aeration and product removal. Constraints are then imposed on the system behaviour. These can be: determined from biological knowledge; inferred from data by using regression to find the coefficients of a proposed reaction network or inferred from data using principle component analysis without any prior knowledge of the system. The kinetics of the free reactions can be modelled using standard equations if the mechanisms are known. However the focus of this thesis was on inferring the kinetics from data.

A summary of the SVM method.

Building data driven models can also be time consuming since black box structures such as neural networks need to be carefully designed in order to avoid problems of overfitting and local minima. In response to these issues a method was introduced based round the use of support vector machines for 'black box' modelling of reaction kinetics. The support vector machine training process is a considerably simpler process than neural network training.

- SVMs do not suffer from local optimum. They can be solved quickly using simple training methods. This is in contrast to traditional approaches to neural network training, which require neural networks to be trained multiple times with random initial weights.
- The complexity of the SVM is determined from data so a to minimise the generalisation error. Only two hyper parameters need to be determined on the basis of cross validation performance. Difficult decisions about the number of neurons or basis functions are therefore avoided.

The method was demonstrated on both real and simulated systems. On a *S. clavulingerus* system the SVM method produced a model of comparable[60] accuracy to published work.

The SVM methodology was then compared with existing neural network and genetic programming based techniques on a large sample of randomly created simulated systems. This approach allows conclusions to be drawn about the general performance of an automated modeling methodology on a particular class of problems. The approach itself may therefore be a useful tool which could be used in future for evaluating which methodologies are most appropriate for modeling a particular class of system. On these test systems the SVM methodology consistently outperformed a FFNN methodology but no statistically significant difference could be observed between the SVM and GP methodologies. This result validates the belief that the SVM hybrid modeling methodology is competitive with existing methods.

[60] Only visual comparison was possible.

A summary of the SVM method.

Building data driven models can also be time consuming since black box structures such as neural networks need to be carefully designed in order to avoid problems of overfitting and local minima. In response to these issues a method was introduced based round the use of support vector machines for 'black box' modelling of reaction kinetics. The support vector machine training process is a considerably simpler process than neural network training.

- SVMs do not suffer from local optimum. They can be solved quickly using simple training methods. This is in contrast to traditional approaches to neural network training, which require neural networks to be trained multiple times with random initial weights.

- The complexity of the SVM is determined from data so a to minimise the generalisation error. Only two hyper parameters need to be determined on the basis of cross validation performance. Difficult decisions about the number of neurons or basis functions are therefore avoided.

The method was demonstrated on both real and simulated systems. On a *S. clavulingerus* system the SVM method produced a model of comparable[60] accuracy to published work.

The SVM methodology was then compared with existing neural network and genetic programming based techniques on a large sample of randomly created simulated systems. This approach allows conclusions to be drawn about the general performance of an automated modeling methodology on a particular class of problems. The approach itself may therefore be a useful tool which could be used in future for evaluating which methodologies are most appropriate for modeling a particular class of system. On these test systems the SVM methodology consistently outperformed a FFNN methodology but no statistically significant difference could be observed between the SVM and GP methodologies. This result validates the belief that the SVM hybrid modeling methodology is competitive with existing methods.

[60] Only visual comparison was possible.

Summary of the Bayesian framework for hybrid modeling.

The Achille's heal of the support vector machine method is that it requires a continuous signal for the reaction rates and state variables. In practice this is achieved by interpolating the data and then taking the derivative. Where the data is noisy and sparsely sampled, or where key variables are simply unmeasured, the SVM methodology and other similar approaches cannot be used.

The hybrid-modelling problem was thus recast in terms of Bayesian inference, directly using non interpolated data. The Bayesian approach is able to use data even if some series are completely missing. This allows the Bayesian approach to use experimental data, which would otherwise not be appropriate for model building. This feature is important since the effort involved in model building cannot be simply quantified in terms of experiments performed and time required to train the model but rather must take into account the manpower required to perform sampling and analysis.

The Bayesian approach provides a principled framework in which both mechanistic and black box components can be embedded and *a priori* beliefs about parameter values incorporated. An additional advantage is that the predictions of Bayesian models are in the form of probability distributions and therefore uncertainty is explicitly handled. Finally it was shown that the Bayesian approach was capable of performing model selection. Unfortunately the computational cost associated with Monte Carlo integration limits the practical application of this approach at present.

10.2 Suggestions for future research.

The SVM methodology.

Clearly the support vector methodology can be refined further. For example: minor improvements could be realised by trying different kernel functions and greater improvements could be realised by improving the interpolation method. More intriguingly there are steps in AI research towards the unification of support vector machines and Gaussian processes (Kwok(2000); Seeger(2003)). It therefore may be possible, in future, to use support vector machines as part of the Bayesian methodology, although it is not possible at present.

Bayesian methodology.

Practical usage of the Bayesian methodology is likely to be limited by the need to evaluate high dimensional integrals using Monte Carlo methods. There are a number of issues that would need to be addressed for the Bayesian approach to become widely used:

Firstly we note that the design of Markov chain sampling techniques is something of an art and it is unlikely that engineers in industry will be willing to spend time becoming Monte Carlo experts. Therefore a method for automatically tuning the step size and proposal distribution should be developed.

The most problematic issue is the time required to perform Monte Carlo integration. The issue is particularly acute as each step is computationally expensive since the hybrid model needs to be numerically integrated and compared with training data in order to evaluate the posterior likelihood. A practical approach might be to simply to use parallel processing. Rasmussen(2003) suggests modelling the posterior distribution using Gaussian processes (see appendix E1). Then using the Gaussian process approximation (which is less computationally demanding to evaluate than the true posterior) to simulate the Hamiltonian dynamics (see appendix E4) of the distribution, the final acceptance/rejection step being from the true distribution. Unfortunately Gaussian processes require a O^3 matrix inversion step and this limits the approach to problems of a maximum dimension of 15. Using another method which does not require a matrix inversion such as SVMs instead of Gaussian processes it might be possible to extend this approach to higher dimensional problems.

For model discrimination all possible combinations of models need to be considered. This will clearly be difficult if there are a large number of candidate structures. Instead of selecting the most probable model, as was the focus of chapter nine, it is possible to marginalise over the competing model structures without directly evaluating the evidence for each model using reversible jump Markov chain Monte Carlo (Green(1995)).

Perhaps the most interesting and important area for future work would be to consider efficient algorithms for optimisation under uncertainty. Traditional optimisation

algorithms such as the simplex method work by comparing the value of some objective function at various *trial points*. Within a decision theoretic framework the algorithm should optimise the expected value of the objective function. However the accuracy of the estimate of the expected value and variance obtained from a Bayesian model depends on the number of samples. Accurately evaluating the expected value and variance of the objective function at all trial points is unlikely to be computationally feasible in a practical situation. However the degree of accuracy required at a particular trial point will depend on the position of that point. For example: High accuracy may be required close to the optimum or areas where the objective function is relatively flat. Areas where the gradient of the objective function is steep or which are far from the optimum do not need to be evaluated accurately. An example of such an algorithm is 'stochastic annealing' (Painton and Diwekar(1995)).

Appendix A1: A brief note on process validation issues.

The pharmaceutical and biologics industries are highly regulated. Falling under the remit of the European Agency for the Evaluation of Medicinal Products, (EMEA), within the European Union and, in the United States the Food and Drug Administration (FDA). As such the process needs to be validated according to defined guidelines to prove a safe product can be consistently produced.

The process is defined as including control systems therefore, the use of models for optimal control or inferential sensors as part of process operation, requires validation.

Guidelines for software validation.

To comply with both FDA and EMEA criteria control processes should follow ' Good Automated Manufacturing Practice' (GAMP[61]). The main requirement is that each software module should have known and consistent behaviour. Sample GAMP documentation for software developed as part of this thesis can be found at start of the appendix F1. The documentation is not complete but indicates that every method in the software should be understood and tested.

One could however argue that such documentation is not necessary. Since, where software does not form part of a control system it cannot be said to directly affect the process. The tools used for bioprocess development or building models therefore need not be validated[62]. Rather only those software models which form part of a control system would need validation.

An explicitly readable model can easily be produced by training genetic programming trees to reproduce the output of the NN or SVM. The process model then consists of a small number of defined readable equations, which can be slotted into an existing validated control system. The validation of such a process model would be trivial.

[61] The GAMP Forum http://www.ispe.org/gamp/

[62] It would be contradictory to require a hybrid model used in process development to be validated while allowing the use of unvalidated software such as Excel for data processing as part of development.

Process Analytical technology.

The FDA's PAT initiative is an interesting if unclear development. As such this review like all other current interpretations of PAT is likely to be speculative.

The FDA[63] define PAT as:-

> *"a system for designing, analysing, and controlling manufacturing through timely measurements (i.e., during processing) of critical quality and performance attributes of raw and in-process materials and processes with the goal of ensuring final product quality".*

More interestingly they also state:-

> *"It is important to note that the term 'analytical' in PAT is viewed broadly to include chemical, physical, microbiological, mathematical, and risk analysis conducted in an integrated manner."*

PAT represents a shift in ethos from 'testing quality into products' towards improved understanding and control of processes. The PAT initiative is a doctrine of continuous improvement which can be seen as similar to 'six sigma' in other manufacturing industries.

PAT is already a driver for increased online/at line monitoring of processes. It is expected that technologies such as Near infrared spectroscopy, at line HPLC, Biosensors and gas sensor arrays(electronic noses) will become increasingly used. Therefore far more information will be available as part of the process development phase than is currently the case. This will make it more feasible to use hybrid modelling as part of the development strategy.

Another part of PAT seems to be a major shift towards modelling and statistical tools. There is even the suggestion that correlations between process variables and final product quality could form the basis of product release.

> *"product release could be based on relationships established during product-process development and confirmed by both validation and routine review of product-process data for commercial lots". -Balboni(2003)*

[63] **Process Analytical Technologies Subcommittee http://www.fda.gov/cder/OPS/PAT.htm**

It remains to be seen how far this philosophy will go in practice. However many techniques already in use, such as NIR, involve statistical elements called 'chemometrics" so as to relate absorption spectra to chemically relevant information. Even if it does not go much further the PAT initiative should create a culture which is more receptive to statistical modelling than is currently the case in bioprocess manufacturing.

Appendix A2: A brief note on business issues.

Potential commercial exploitation.

The hybrid modelling techniques developed have the potential to boost process profitability and save development time. If when used as part of a process development program they save just one day of a new biologic's period of exclusivity then this represents a saving of $1 million. Unfortunately, until it is proven, in a business setting, that savings can be made companies will be reluctant to invest time and money in trialing new techniques.

The most likely route for commercial exploitation is therefore some kind of joint research agreement between bioprocess companies and either Academia or a broader technological consultancy firm.

Intellectual property.

The situation regarding IP is somewhat complex. At the start of the EngD program a standard IP agreement was drawn up between myself, UCL and the then sponsoring company 'Adaptive Biosystems'. The agreement essentially stated that any software was the property of the sponsoring company, any non-software IP the property of UCL in consideration of a 10% stake going to the researcher.

The core architecture for the hybrid modelling software used in this thesis, was written by myself in collaboration with David Sweeny and Chris Taylor who were at that time employees of Adaptive Biosystems. However Adaptive Biosystems ceased trading in 2003 and hence stopped sponsoring this research program. The adaptive code is now public domain. Since that time I have added substantially to the program and made use of 3 public class libraries:

'NewMAT', a matrix algebra class library. (http://www.robertnz.net/)

'Mersenne Twister', a pseudo random number generator, (http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html' Matsumoto and Nishimura(1998)).

'LibSVM', a library for training support vector machines. ('http://www.csie.ntu.edu.tw/~cjlin/libsvm/ ' Chang, C. C. et al(2001))

All parties have given permission for the non commercial use of code. However any release of the software as commercial package would require the agreement of the various parties. If no permission was given, code could be rewritten to exclude any or all proprietary components, but this would require the investment of a several man months of programming effort.

Appendix B1: The Runge Kutta numerical integration method.

Since the hybrid model representation is simply one of a system of ordinary differential equations in matrix form standard numerical integration schemes can used without modification. For conceptual simplicity, and to avoid focusing too much on numerical integration schemes, a standard 4th order Runge Kutta method without adaptive step sizes as shown in Figure 103.was chosen.

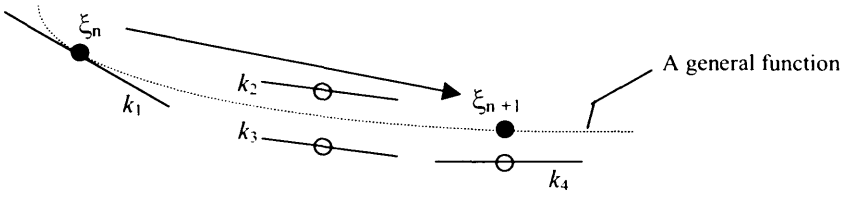


Figure 103 Operation of the fourth order Runge Kutta method: For every step, the derivative is calculated four times – once at the n[th] step (k_1), twice at the midpoints (k_2 & k_3) and once at a trial endpoint (k_4). • = a function point; o = trial point (Flannery, B. P. et al(1999))

For a function $\frac{d\xi}{dt} = Kr(\xi, v)$

$$\begin{aligned} k_1 &= \Delta t \times Kr(\xi_{t=t_n}, v_{t_n}) \\ k_2 &= \Delta t \times Kr(\xi_{t=t_n} + \frac{k_1}{2}, v_{t_n+\frac{\Delta t}{2}}) \\ k_3 &= \Delta t \times Kr(\xi_{t=t_n} + \frac{k_2}{2}, v_{t_n+\frac{\Delta t}{2}}) \\ k_4 &= \Delta t \times Kr(\xi_{t=t_n} + k_3, v_{t_n+\Delta t}) \end{aligned} \qquad (12.1)$$

Hence giving rise to the fourth order Runge Kutta formula:

$$\xi_{t-n+\Delta t} = \xi_{t-n} + \left[\frac{(k_1 + 2k_2 + 2k_3 + k_4)}{6}\right] + O(\Delta t^5) \qquad (12.2)$$

The $O(\Delta t^5)$ term refers to the $5-1=4^{th}$ order error induced in the actual answer by virtue of using an approximate algorithm. The series of k expressions in brackets refers to the weighted average of the values that forms the final term added to the current (step n) value of the function to evaluate the (n+1)[th] position.

Appendix B2: Randomly generating dynamic systems for testing purposes.

Clearly any number of published models can be found in literature that could be used to simulate experimental data and thus to evaluate the feasibility of different modelling approaches. These systems, and indeed experimental test systems, are useful as illustrative examples and indicative of likely performance on similar problems but it is questionable what conclusions this allows us to draw about the performance of modelling methodologies in general.

It is therefore proposed that the relative performance of different modelling methodologies is compared on a (reasonably) large number of randomly generated systems.

The proposed procedure is as follows.

1. Create a $K \in \Re^{N_s \times N_r}$ constraint matrix and vector of functions $r(\xi,\nu) = (r_1(\xi,\nu), r_2(\xi,\nu), ..., r_{N_{reac}}(\xi,\nu))^T \in \Re^{N_r}$.

2. Obtain the 'concentration' time profiles by integrating the system given some initial conditions and control actions.

3. Simulate experimental data by sampling from this continuous profile and adding noise.

4. Produce a model of the system from the simulated experimental data using all the methodologies to be compared.

5. Record the performance of the methodologies according to some measure.

It is immediately clear that the pseudo stoichiometric matrix can be produced simply[64] by drawing $N_s \times N_r$ random numbers from some predefined distribution. What is less clear is how functions can be created at random. This is achieved as follows.

The two sets are defined.

[64] The comprehensively studied Mersenne Twister method, Matsumoto, M. et al(1998), was used for generating pseudorandom numbers.

The first consists of terminals. $Terminals = \xi \cup v \cup w$ These are the state variables, the environmental variables and parameter values consisting of randomly created floating point numbers $w \in \Re$.

The second set consists of basic operators. $Operators = \{+, -, *, \%, mm, sig\}$ where % denotes protected division, $mm(x) = \frac{x}{x+1}, sig(x) = \frac{1}{1+e^{-x}}$.

The set of basic operators can take as arguments either a terminal or the result of another operator. In this way equations can be represented as hierarchical tree structure where the nodes are elements from either of the above sets.

$$3a + \frac{(b-a)}{c}$$

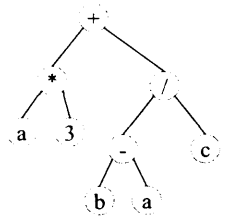


Figure 104 Parse tree.

A random equation can be generated by the following pseudocode:-

```
Create_random_tree()
{
        Root node~ ∈(Operators)
        Generate_subnodes(root_node)
}

Generate_subnodes(parent_node)
{
        for(i=1;to parent_node->number_of_arguments; i++)
        {
        new_node=← ∈(Operators ∪ Terminals)
        parent_node.argument(i)=new_node
        if(new_node is an operator) Generate_subnodes(new_node)
        }
}
```

Appendix B3: The model of Jang, J. D. et al(2000).

Introduction.

This appendix gives the details of the published unstructured model of a fed-batch murine hybridoma cell culture, one of the toy systems used to evaluate the various modelling approaches developed in this thesis. The model is presented as a series of differential equations for cell concentration and dry cell weight, glucose and glutamine (both substrates), ammonia and lactate (both by-products), macromolecules; DNA, RNA, polysaccharides, lipids and proteins and also product formation of monoclonal antibodies.

A key feature of the true system is that hybridoma cells in the culture must pass through as series of phases in order to divide the gap 1 phase (G1-phase), the DNA synthesising phase (S-phase), the gap 2 phase (G2-phase) or the mitotic phase (M-phase). Therefore, in the model, cells are grouped into three categories: viable cycling cells, viable arrested cells, and dead cells (Figure 105).

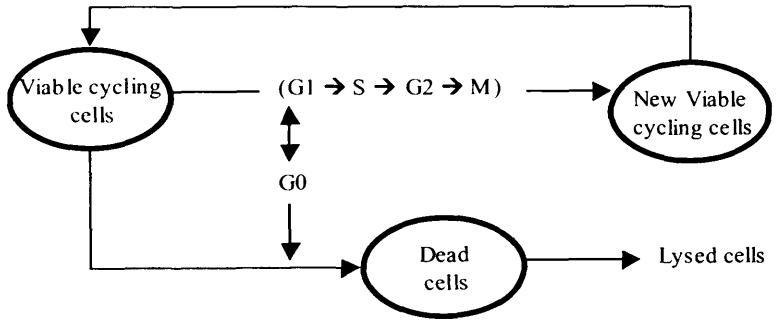


Figure 105 Cell cycling in Jang, J. D. et al(2000).

This is not explicitly modelled by Jang and Barford as a stratified population with cells moving between pools but rather the proportions in each phase are estimated by a linear function and this ratio enters into subsequent rate equations.

The presented model.

Hybridoma growth.

The growth of hybridoma cells, describing the change of viable cell mass, X_v, with respect to time, can be described by the following equation:

$$\frac{dX_v}{dt} = \mu X_v - \mu_d X_v - \frac{F_0}{V} X_v \qquad (12.3)$$

With the equivalent equation for dead cell mass being given as:

$$\frac{dX_d}{dt} = \mu_d X_v - K_{lys} X_d - \frac{F_0}{V} X_d \qquad (12.4)$$

Where μ is the maximal specific growth rate given as a function of substrate and inhibitor concentrations by equation (12.5), and f_{go} that fraction of cells that have arrested growth and remain in the G0 phase estimated as a linear function of μ_d by equation (12.7).

In equation (12.7) 'a', 'b' are the death rate constants taken from literature, μ_d is the maximal specific death rate which increases with ammonia concentration (12.6), F_0 is the volumetric flow rate out of the reactor due to sampling or harvesting and V is the culture volume. K_{lys} is a cell lysis specific rate constant.

$$\mu = \mu_{\max}\left[\frac{[Glc]}{K_{glc}+[Glc]}\right]\left[\frac{[Gln]}{K_{gln}+[Gln]}\right]\left[\frac{K_{i,amm}}{K_{i,amm}+[Amm]}\right]\left[\frac{K_{i,lac}}{K_{i,lac}+[Lac]}\right](1-f_{go}) \qquad (12.5)$$

$$\mu_d = \left[\frac{\mu_{dm}}{1+\left[\frac{K_{d,Amm}}{[Amm]}\right]^n}\right] \qquad n>1 \qquad (12.6)$$

$$f_{GO} = \frac{\mu_d - a}{b} \qquad (12.7)$$

Kinetics of antibody production.

Jang and Barford noted that the specific antibody production rate was higher for cells in the G1 phase than other phases. Assuming Glutamine to be the limiting substrate upon which antibody production was dependent they expressed the differential equations for antibody production over time as:

$$\frac{d[Ab]}{dt} = Q_{ab}\left(\frac{[Gln]}{[Gln]+K_{gln}}\right)X_v - \frac{F_0}{V}[Ab] \qquad (12.8)$$

with Q_{ab} is given by:

$$Q_{ab} = Q_c\left(1-f_{go}\right) + Q_{go} f_{go} \qquad (12.9)$$

Where Q_c is the specific antibody production rate for cycling cells and Q_{go} is the specific antibody production rate for cells arrested in the G0 phase.

DNA production.

In the unstructured model, it was assumed that only glucose and glutamine were major limiting substrates of the system and other nutrients were assumed not to limit cell growth, DNA synthesis or product formation.

$$\frac{d[DNA]}{dt} = \mu K_{dna}[DNA]\left(\frac{X_v}{X_t}\right) - K_{ddna}[DNA] - \frac{F_0}{V}[DNA] \qquad (12.10)$$

where K_{dna} is a conversion factor for the specific DNA production rate and K_{ddna} is a DNA degradation rate constant X_t.

RNA production.

RNA molecules are synthesised only by translation of DNA, the rate of which is dictated by the amount of DNA available as a template and the concentration of ribonucleotides available. The model is assumed to be of the same form as for DNA synthesis. In addition, Jang and Barford assumed that for the mammalian cells in question, the majority of the RNA was present as stable ribosomal or transfer RNA – hence the term Q_{mrna} refers to the specific mRNA production rate, with K_{deg1} and K_{deg2} being degradation rate constants:

$$\frac{d[RNA]}{dt} = Q_{mrna} f_{lim} f_{inh}[DNA]\left(\frac{X_v}{X_t}\right) - (K_{deg1} f_m + K_{deg2}(1 - f_m))[RNA] - \frac{F_0}{V}[RNA] \qquad (12.11)$$

for limitation and inhibition respectively.

f_m is the fraction of messenger RNA in the total RNA pool. such that:

$$f_{lim} = \prod_{i=1}^{n}\left(\frac{[S_i]}{KS_i + [S_i]}\right) \text{ and } f_{inh} = \prod_{i=1}^{n}\left(\frac{KI_i}{KI_i + [I_i]}\right) \qquad (12.12)$$

Where S and I refer to the concentration of the i^{th} limiting substrate and inhibitor respectively. f_{lim} and f_{inh} represent functions of nutrient limitation and by-product inhibition respectively. The KS_i & KI_i values are constants.

Protein production.

Protein synthesis is directly linked to the available quantity of RNA for peptide chain elongation. For amino acids, the model presented glutamine as the only factor that limited the rate of protein synthesis, i.e. all other amino acids were assumed to be in excess.

$$\frac{d[PRT]}{dt} = Q_{mprt} f_{lim} f_{inh} [RNA] \left(\frac{X_v}{X_t} \right) - K_{dprt} [PRT] - \frac{F_0}{V} [PRT]. \qquad (12.13)$$

Q_{mprt} represents the specific protein production rate, while K_{dprt} is the protein degradation rate constant.

Lipid synthesis.

Lipids are synthesised by some proteins (enzymes) using palmitate and glycerol as precursors. Hence, letting Q_{mlpd} represent the specific cellular lipid production rate and k_{dlpd} the degradation rate constant of cellular lipid:

$$\frac{d[LPD]}{dt} = Q_{mlpd} f_{lim} f_{inh} [PRT] \left(\frac{X_v}{X_t} \right) - K_{dlpd} [LPD] - \frac{F_0}{V} [LPD]. \qquad (12.14)$$

Polysaccharide synthesis.

Polysaccharides, such as glycogen, are biosynthesised by proteins, (in the form of enzymes), using glucose as the starting substrate. Polysaccharide degradation may occur depending upon the conditions in the cell, using different enzymes from those seen in the synthesis pathway. Letting Q_{mpsd} represent the specific polysaccharide production rate inside the cell and K_{dpsd} represent the polysaccharide degradation rate constant, the following equation is presented:

$$\frac{d[PSD]}{dt} = Q_{mpsd} f_{lim} f_{inh} [PRT] \left(\frac{X_v}{X_t} \right) - K_{dpsd} [PSD] - \frac{F_0}{V} [PSD] \qquad (12.15)$$

Substrate metabolism.

Consumption of glucose and glutamine is described as a function of specific growth rate and substrate concentration. In the case of glucose, Jang and Barford also state a dependency in the relationship on the maintenance energy. Hence the equations for glucose and glutamine may be presented as:

$$\frac{d[Glc]}{dt} = Q_{glc} X_v - \frac{F_0}{V} [Glc]$$
$$where\ Q_{glc} = \frac{\mu}{Y_{x,glc}} + m_{glc} + Q_{exglc} \left(\frac{[Glc]}{[Glc] + k_{exglc}} \right) \qquad (12.16)$$

$$\frac{d[Gln]}{dt} = Q_{gln} X_v - K_{dgln} [Gln] - \frac{F_0}{V} [Gln]$$
$$where\ Q_{gln} = \frac{\mu}{Y_{x,gln}} \qquad (12.17)$$

Where Q_{glc} and Q_{gln} represent specific consumption rates of glucose and glutamine respectively. $Y_{x,glc}$ and $Y_{x,gln}$ are cell yields from glucose and glutamine. m_{glc} is the maintenance energy for glucose and K_{dgln} is the rate of spontaneous glutamine degradation. Glucose consumption is dependent on the enzyme hexokinase at low glucose concentrations and the enzyme glucokinase at reasonably high glucose concentrations, (for example, 20 mM).

To this end Q_{exglc} represents the specific glucose consumption rate by glucokinase and k_{exglc} is the Monod constant for glucokinase.

Lactate and ammonia production.

Jang and Barford state that, for the animal cell culture under discussion, the rate of lactate production is related to the consumption of glucose. Hence, the following equations were formulated:

$$\frac{d[Lac]}{dt} = Q_{lac}X_v - \frac{F_0}{V}[Lac] \qquad (12.18)$$
$$where\ Q_{lac} = Y_{lac,glc}Q_{glc}$$

Where Q_{lac} is the specific lactate production rate and $Y_{lac,glc}$ represents the yield of lactate from glucose.

For the purposes of the current model, ammonia production was related to glutamine consumption (Ozturk and Palsson, 1991). Q_{amm} stands for the specific ammonia production rate, K_{dgln} is the rate of spontaneous glutamine degradation and $Y_{amm,gln}$ is the yield of ammonia from glutamine:

$$\frac{d[Amm]}{dt} = Q_{amm}X_v + K_{dgln}[Gln] - \frac{F_0}{V}[Amm] \qquad (12.19)$$
$$where\ Q_{amm} = Y_{amm,gln}Q_{gln}$$

The values of coefficients are:

Table 15. Summary of constants used in Jang, J. D. et al(2000).

Constant	Value	Constant	Value
μ_{max}	$0.065\ h^{-1}$	$Y_{lac,glc}$	$2.0\ mol\ mol^{-1}$
μ_{dmax}	$0.075\ h^{-1}$	$Y_{amm,gln}$	$0.7\ mol\ mol^{-1}$
K_{glc}	0.75 mM	K_{dns}	0.9 dimensionless
K_{gln}	0.075 mM	Q_{mns}	$0.8\ mg\ cell^{-1}\ h^{-1}$
KI_{lac}	90 mM	Q_{mpn}	$0.5\ mg\ cell^{-1}\ h^{-1}$
KI_{amm}	15 mM	Q_{mlpd}	$0.012\ mg\ cell^{-1}\ h^{-1}$
K_{dAmm}	4.5 mM	Q_{mpsd}	$0.4\ mg\ cell^{-1}\ h^{-1}$
Q_c	$0.70\ pg\ cell^{-1}\ h^{-1}$	K_{ddns}	$1.0\times10^{-12}\ h^{-1}$
Q_{GG}	$1.00\ pg\ cell^{-1}\ h^{-1}$	K_{deg1}	$0.03\ h^{-1}$
$Y_{x,glc}$	$2.37\times10^{8}\ cells\ mol^{-1}$	K_{deg2}	$0.0001\ h^{-1}$
m_{glc}	$2.0\times10^{-12}\ mmol\ cell^{-1}\ h^{-1}$	K_{dpn}	$0.02\ h^{-1}$
$Y_{x,gln}$	$8.0\times10^{8}\ cells\ mol^{-1}$	K_{dlpd}	$0.005\ h^{-1}$
Q_{exglc}	$2.0\times10^{-10}\ mmol\ cell^{-1}\ h^{-1}$	K_{dpsd}	$0.15\ h^{-1}$
K_{exglc}	10 mM		

Writing in matrix form.

The model for the media components, biomass and product can be written in state space form as:

$$\frac{d}{dt}\begin{bmatrix} X_v \\ X_d \\ [Ab] \\ [Glc] \\ [Gln] \\ [Lac] \\ [Amm] \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1/Y_{x,glc} & 0 & 0 & 0 & -1 & 0 & -1 \\ -1/Y_{x,gln} & 0 & 0 & 0 & 0 & -1 & 0 \\ \frac{Y_{lac,glc}}{Y_{x,glc}} & 0 & 0 & 0 & Y_{lac,glc} & 0 & -1 \\ \frac{Y_{amm,gln}}{Y_{x,gln}} & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \end{bmatrix} + \frac{F_{in}}{V}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \\ [Glc]_{in} \\ [Gln]_{in} \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} X_v \\ X_d \\ [Ab] \\ [Glc] \\ [Gln] \\ [Lac] \\ [Amm] \end{bmatrix} \right)$$

where,

$$r_1 = \mu, r_2 = \mu_d\ r_3 = K_{lys} X_d, r_4 = Q_{ab}\left(\frac{[Gln]}{[Gln]+K_{gln}}\right)X_v, r_5 = Q_{exglc}\left(\frac{[Glc]}{[Glc]+k_{exglc}}\right)X_v$$

$r_6 = K_{dgln}[Gln], r_7 = m_{glc} X_v$ With the various sub terms as defined earlier.

In Jang and Barford's model each rate of production/degradation of intracellular macromolecules is independent of any other rate. The kinetics of each reaction have

the sub-unit $f_{\lim} f_{inh}\left(\frac{X_v}{X_t}\right)$ in common, but this is just a similarity of kinetic functions, they are not linked by any mass balance or enthalpy conservation constraints in the model. This is apparent from writing the model for intracellular macromolecules in matrix form where no column element appears in more than one row.

$$\frac{d}{dt}\begin{pmatrix}[DNA]\\[RNA]\\[PROT]\\[LPD]\\[PSD]\end{pmatrix}=\begin{pmatrix}1&0&0&0&0&-1&0&0&0&0\\0&1&0&0&0&0&-1&0&0&0\\0&0&1&0&0&0&0&-1&0&0\\0&0&0&1&0&0&0&0&-1&0\\0&0&0&0&1&0&0&0&0&-1\end{pmatrix}\begin{pmatrix}r_8\\r_9\\r_{10}\\r_{11}\\r_{12}\\r_{13}\\r_{14}\\r_{15}\\r_{16}\\r_{17}\end{pmatrix}+u$$

where,

$$r_8=\mu K_{dna}[DNA]\left(\frac{X_v}{X_t}\right), r_9=Q_{mrna} f_{\lim} f_{inh}[DNA]\left(\frac{X_v}{X_t}\right), r_{10}=Q_{mprt} f_{\lim} f_{inh}[RNA]\left(\frac{X_v}{X_t}\right),$$

$$r_{11}=Q_{mlpd} f_{\lim} f_{inh}[PRT]\left(\frac{X_v}{X_t}\right), r_{12}=Q_{mpsd} f_{\lim} f_{inh}[PRT]\left(\frac{X_v}{X_t}\right), r_{13}=K_{ddna}[DNA],$$

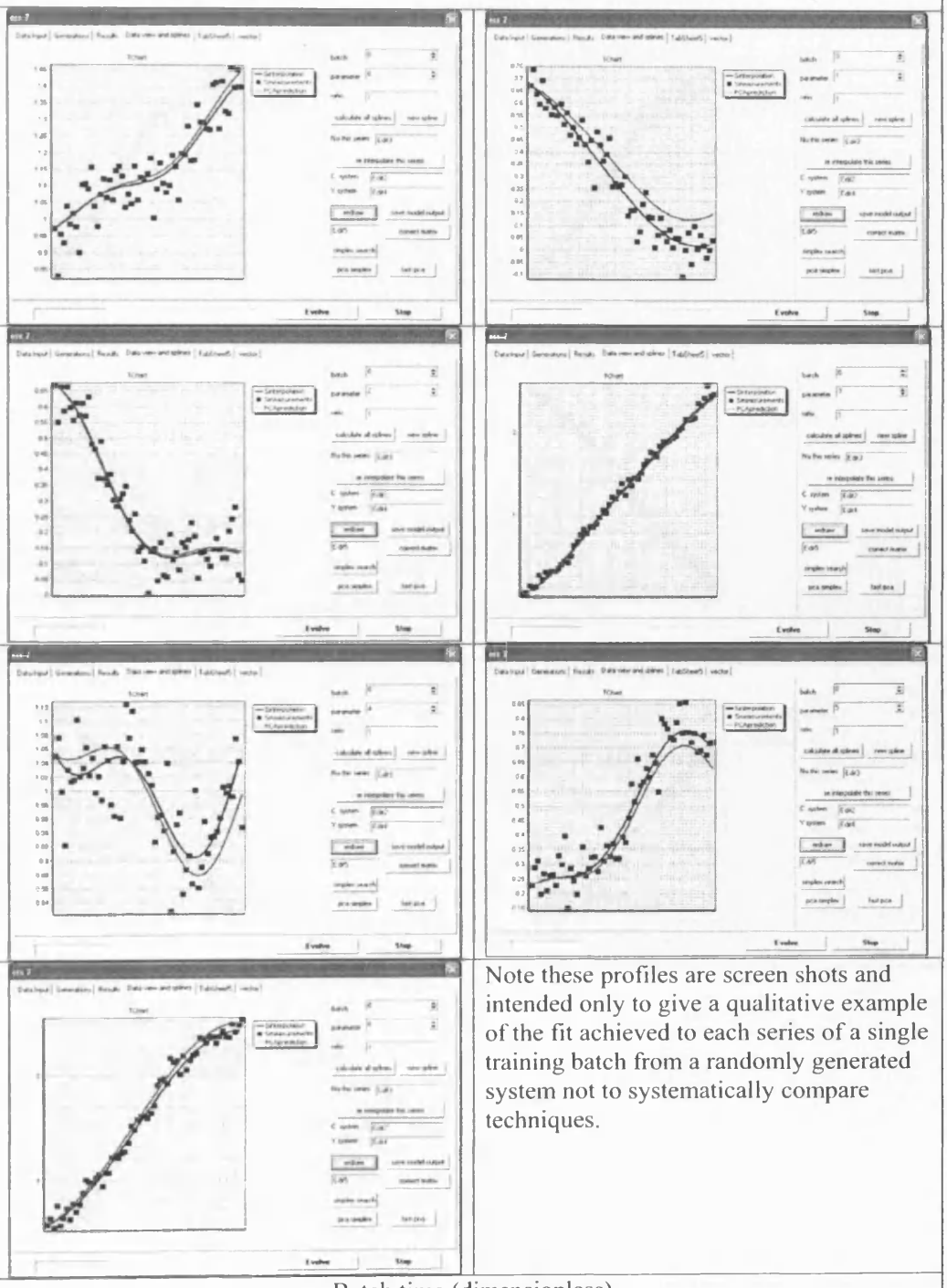
$$r_{14}=(K_{\deg 1} f_m+K_{\deg 2}(1-f_m))[RNA], r_{15}=K_{dprt}[PRT], r_{16}=K_{dlpd}[LPD]$$

$$r_{17}=K_{dpsd}[PSD]$$

Further note that while the macromolecule kinetics depend on metabolite concentrations the metabolite kinetics, (as stated in the above model), in no way depend on the macromolecules.

Appendix C1: Example profiles obtained for kinetic models built using the PCA method.

Table 16. Screen shots of the performance of PCA models with perfect kinetic modelling.

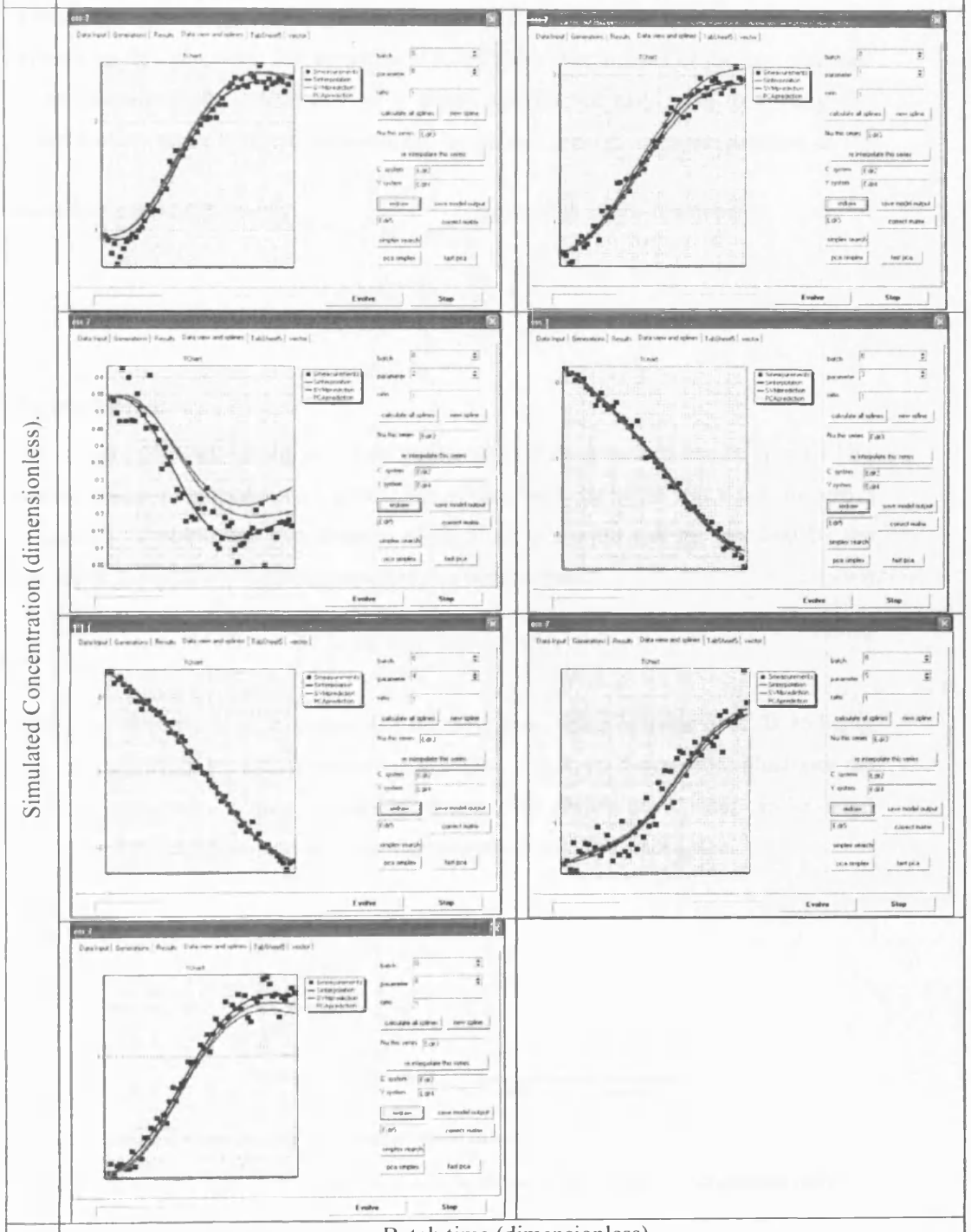


Simulated Concentration (dimensionless).

Note these profiles are screen shots and intended only to give a qualitative example of the fit achieved to each series of a single training batch from a randomly generated system not to systematically compare techniques.

Batch time (dimensionless)

Table 17. Screen shots of the performance of Kinetic models built using PCA method and knowing the constraint matrix.



Appendix C2: Transforming principal components back into phase space.

While the system is restricted to the manifold defined by the principal components, in theory any valid coordinate system can be used to represent the evolution of the system on this manifold. For example, if it is known that the rate of the key reactions is the measured rate of change of a single species we may wish to specify our constraints in terms of these components. In general form given some partition of the state space and PCA matrix $\eta^T = [\eta_a^T, \eta_b^T]$ $M^T = [M_a^T, M_b^T]$ the model can be rewritten as:

$$\begin{aligned} \eta &= MM_a^{-1}(\eta_a - \bar{\eta}_a) + \bar{\eta} \\ \eta &= MM_a^{-1}\eta_a - MM_a^{-1}\bar{\eta}_a - \bar{\eta} \\ \eta &= MM_a^{-1}\eta_a \end{aligned} \tag{13.1}$$

In more detail.

Any point which lies on the manifold defined by the PCA vectors can be specified by adding principal components together and adding back the mean. For a system with 3 measured variables and two degrees of freedom any point can be specified by the values of A and B (the rates in principle component space).

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = A \times \begin{pmatrix} PCA_1^X \\ PCA_1^Y \\ PCA_1^Z \end{pmatrix} + B \times \begin{pmatrix} PCA_2^X \\ PCA_2^Y \\ PCA_2^Z \end{pmatrix} + \begin{pmatrix} \bar{X} \\ \bar{Y} \\ \bar{Z} \end{pmatrix} \tag{13.2}$$

Thus our objective is to transform the equation from being in terms of A, B, to being in terms of two independent chosen coordinates, e.g. X,Y. Figure 106. illustrates the process graphically. A point specified in terms of the chosen coordinates can be found in terms of the principal components then projected into full state space

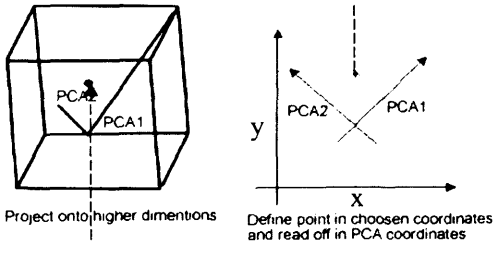


Figure 106 transforming from PCA space to phase space.

First equation (13.2) is partitioned so that it is terms of the chosen coordinates only

$$\begin{pmatrix} X \\ Y \end{pmatrix} = A \times \begin{pmatrix} PCA_1^X \\ PCA_1^Y \end{pmatrix} + B \times \begin{pmatrix} PCA_1^X \\ PCA_1^Y \end{pmatrix} + \begin{pmatrix} \bar{X} \\ \bar{Y} \end{pmatrix}. \tag{13.3}$$

Since the reduced PCA matrix is by definition square and non singular it is invertible and so (13.3) can be easily solved for A,B.

$$\begin{pmatrix} PCA_1^X & PCA_2^X \\ PCA_1^Y & PCA_2^Y \end{pmatrix}^{-1} \begin{pmatrix} X - \bar{X} \\ Y - \bar{Y} \end{pmatrix} = \begin{pmatrix} A \\ B \end{pmatrix}. \tag{13.4}$$

Substituting this into (13.2) we obtain:-

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} PCA_1^X & PCA_2^X \\ PCA_1^Y & PCA_2^Y \\ PCA_1^Z & PCA_2^Z \end{pmatrix} \begin{pmatrix} PCA_1^X & PCA_2^X \\ PCA_1^Y & PCA_2^Y \end{pmatrix}^{-1} \begin{pmatrix} X - \bar{X} \\ Y - \bar{Y} \end{pmatrix} + \begin{pmatrix} \bar{X} \\ \bar{Y} \\ \bar{Z} \end{pmatrix} \tag{13.5}$$

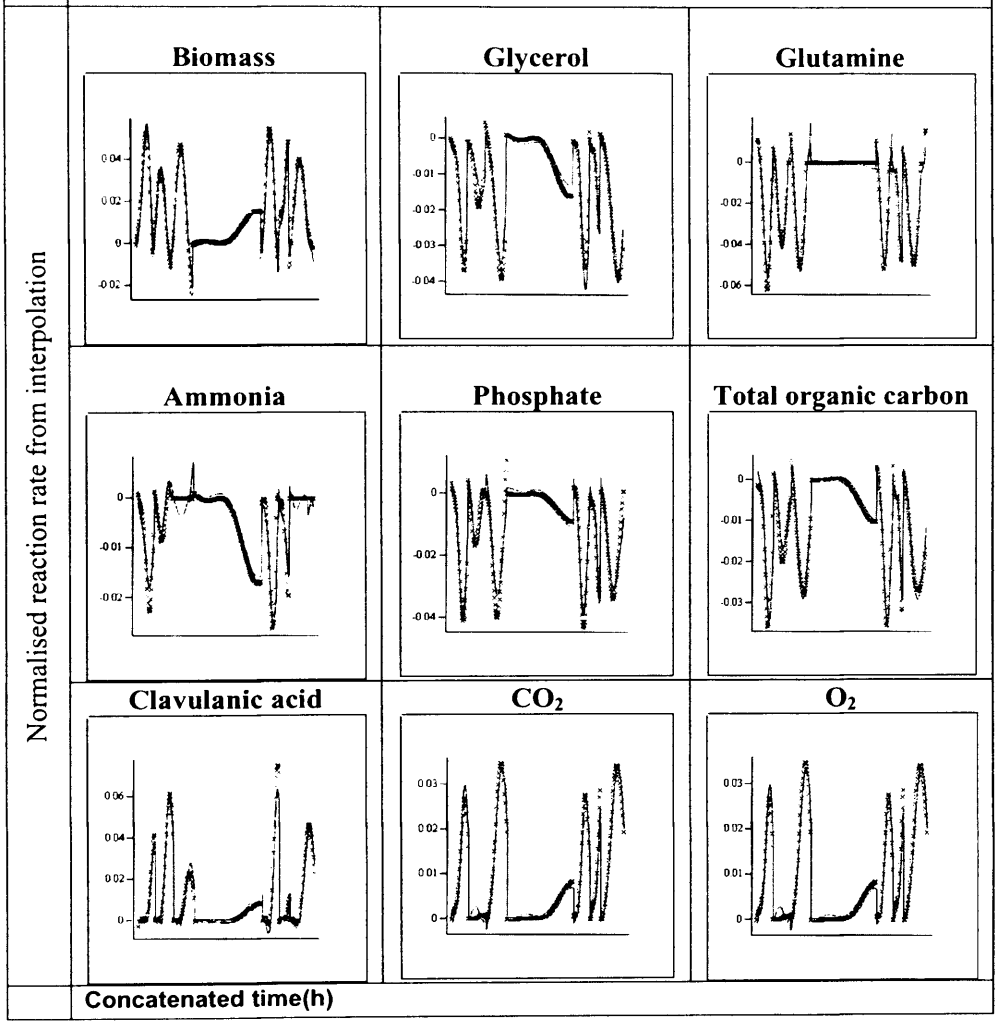
Finally the means cancel

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} PCA_1^X & PCA_2^X \\ PCA_1^Y & PCA_2^Y \\ PCA_1^Z & PCA_2^Z \end{pmatrix} \begin{pmatrix} PCA_1^X & PCA_2^X \\ PCA_1^Y & PCA_2^Y \end{pmatrix}^{-1} \begin{pmatrix} X \\ Y \end{pmatrix}. \tag{13.6}$$

The above process transforms a vector specified in terms of principal components back into the space of measured rates of change. This may be useful in understanding the constraints imposed by the PCA process. However, there may well be other useful coordinate transforms. For example, independent component analysis takes principal components as an input and separates the signals to obtain vectors where the mutual information between signals is minimised. This may prove to be an interesting approach to minimising the redundancy of kinetic models but is not pursued in this thesis.

Appendix C3: Principal components fits to Streptomyces clavulingerus *batch cultivation data.*

Figure 107. Normalised reaction rates implied from free rates by constraint matrix inferred using principal component analysis(line) vs. normalised reaction rates obtained by differentiating the interpolated measurements (points).



Appendix D1: A brief tutorial on Lagrange multipliers.

Lagrange multipliers are a tool for constrained optimisation. The objective is to find the extreme points, that is maxima, minima or saddle points, given a differentiable function and some constraints. Consider maximising a function:-

$$f(x) = 1 - x_0^2 - 2x_1^2 \qquad x \in \mathrm{R}^2 \tag{14.1}$$

subject to the equality constraint:-

$$g(x) = x_0^2 + x_1^2 - 1 = 0 \tag{14.2}$$

Figure 108 shows a surface plot of the function to be optimised. The thin ellipses are iso-contour lines corresponding to points which have the same value of f. The thick line represents all those points where the constraint is satisfied and are thus allowed solutions.

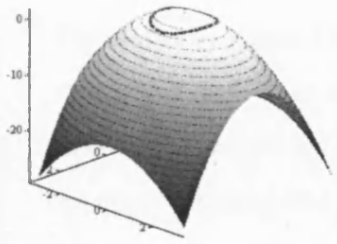


Figure 108. Function to optimise subject to constraint shown as dark circle.

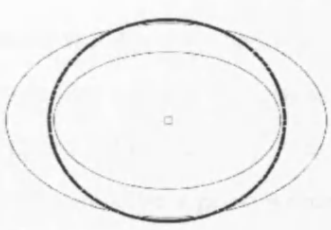


Figure 109. Intersections between constraint and contour lines.

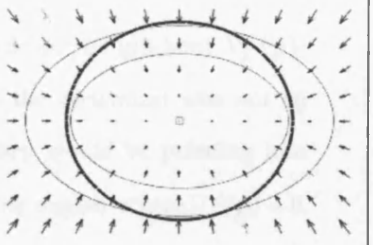


Figure 110. Arrows indicate direction in which function increases (normal vectors). Adapted from Klein(2005).

Ignore the fact that this simple example can easily be solved by substitution and consider the following geometric intuition. The constrained maximum can be found by starting with a very small ellipse at the unconstrained maximum (0,0) and slowly expanding it until it touches the constraint boundary. At this point p_1 (Figure 109.) the constraint and the (iso-contour) ellipse are tangential to each other. This means there is no direction along the constraint that will increase the value of f. The normal vectors are parallel to each other, formally written as:-

$$\nabla f(p) = \lambda \nabla g(p). \tag{14.3}$$

The new variable λ is a scaling factor called a Lagrange multiplier and simply accounts for differences in the magnitude of the gradients at this point. The problem can then be written as a Lagranian in the general form for multiple constrains.

$$L(x,\lambda) = f(x) - \sum_i \lambda_i g_i(x) \tag{14.4}$$

For the example problem the Lagranian is:

$$L(x,\lambda) = 1 - x_0^2 - 2x_1^2 - \lambda(x_0^2 + x_1^2 - 1) \tag{14.5}$$

The extreme points are those points where the partial derivatives are zero with respect to all variables. $\nabla L(x,\lambda) = 0$

$$\frac{\delta L}{\delta x_0} = -2x_0 - 2\lambda x_0 = 0, \frac{\delta L}{\delta x_1} = -4x_1 - 2\lambda x_1 = 0, \frac{\delta L}{\delta \lambda} = -x_0^2 + x_1^2 - 1 = 0$$

This has the following solutions:

$\lambda = -1$ in which case $(x_0 = \pm 1, x_1 = 0)$ a minimum.

$\lambda = -2$ in which case $(x_0 = 0, x_1 = \pm 1)$ a maximum.

The arrows in Figure 110 show the direction and magnitude of the gradient $\nabla f(p)$. Notice that the arrows are pointing towards the centre. If the constraint was not an equality but instead the inequality $g(x) = x_0^2 + x_1^2 - 1 \le 0$ they would be pointing into the allowed region. The solution must be a point within this region where $\nabla f(p) = 0$ which can be enforced by setting $\lambda = 0$ and letting $g(x) \neq 0$. But moves in the opposite direction are not allowed. The boundary may be an extreme point if the normal vectors point in opposite directions. i.e. Either $\lambda = 0$ or $g(x) = 0$, but not both and if λ is not zero it is negative.

In general multiple inequalities $g_i(x) \le 0$ can be enforced by the following conditions, known as the first order Karsh Kuhn Tucker conditions.

$$\forall i \ \lambda_i \le 0 \tag{14.6}$$

$$\sum_i \lambda_i g_i(x) = 0, \prod_i (\lambda_i + g_i(x)) \neq 0 \tag{14.7}$$

Appendix D2: Hyper parameter selection by cross validation.

This appendix gives further details of the cross validation approach that was used to select the values of the hyper-parameters C and σ_k, which determine the regularisation and Gaussian kernel width respectively.

The concept of cross validation.

The error of a model on a representative sample of unseen data is likely to be a good predictor of the error on unseen data in general. Cross validation works by splitting the available data into two sets:

Training data. – Used to determine the model.

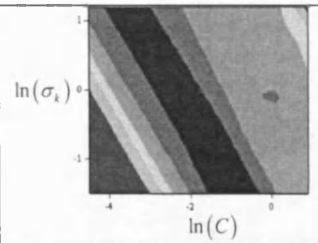
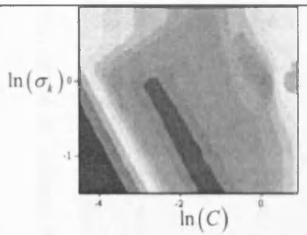
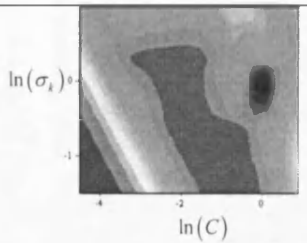
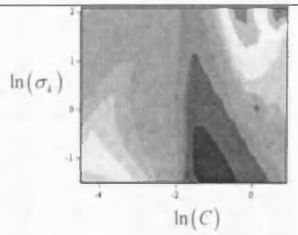
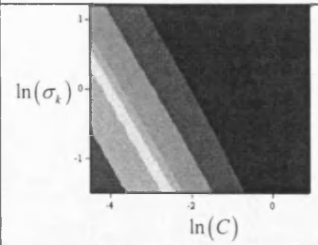
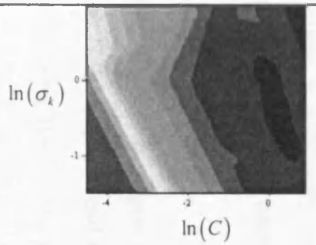
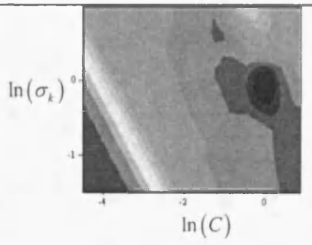
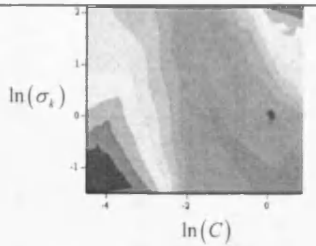
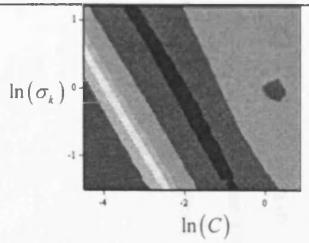
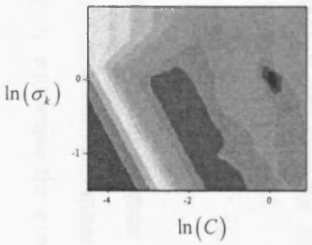
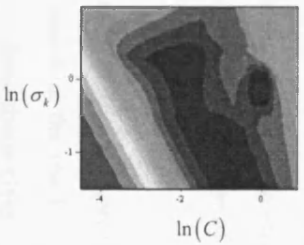
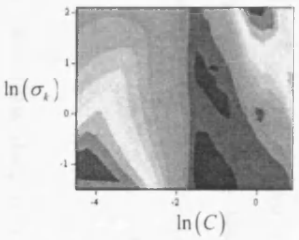
Validation data. – Used to predict the likely error of the model on unseen data.

For hybrid modelling, batches provide a natural basis for dividing data into these sets since decisions will ultimately be taken on the basis of predicted batch profiles. For support vector machines the complexity of the model is determined by the hyperparameters C and σ_k. If these are set so as to minimize the error on validation data then, assuming the validation batches are representative, this should correspond to minimum error on testing batches.

Demonstration.

Support vector machine models of the hybridoma system were trained using different hyper-parameter values forming an equally spaced grid in log space C and σ_k. The average error of the resulting models, on 5 training batches, 2 validation data and 5 testing batches was then plotted as contour plots (Figure 111). This was repeated for 4 different noise levels.

ffFigure 111 Model error as a function of hyperparameter values.
Model errors vary from 0.032(white) to 0.007(black).

	No measurement error	5% measurement error	10% measurement error	15% measurement error
Validation data				
Training data				
Testing data				

Discussion.

Figure 111 shows a set of contour plots of error of models built using the support vector machine methodology. Each column represents a set of training, validation and testing data generated by running the hybridoma model with a specific level of measurement noise. The first row shows the error on training data. The second row shows the error measured on training data. The third row shows the error on testing data.

From Figure 111. it can be seen that the performance on validation data is a good predictor of performance on the unseen data. Since at each noise level the contour plots for the error on validation data are very similar to contour plots for the error on testing data. Performance on training data, by contrast, is not a good predictor of error on testing data.

It is also noteworthy that on high levels of noise high values of C do not improve the fit to that same training data. This is because the model is trained on the reaction rates calculated from the interpolation rather than to the data values themselves. The contour plots also show that the surface is relatively uncomplicated without serious local optima and so most optimisation algorithms are likely to be effective.

Simplex method.

Searching for the set of hyperparameters which minimise the validation error by a grid search is inefficient therefore the Nelder and Mead simplex method Flannery, B. P. et al(1999),was used to adjust the hyper-parameters so as to minimise the error on validation batches.

The simplex method works by constructing a geometrical figure consisting in N dimensions of N+1 vertices, with all of the corresponding interconnecting lines and faces present, so as to enclose a finite inner N-dimensional volume that represents the localised search space.

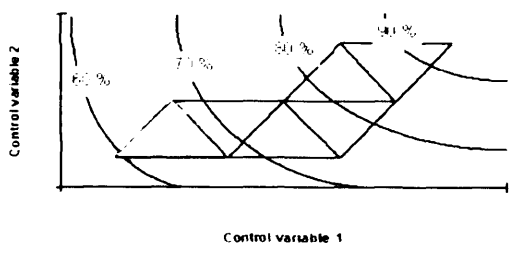


Figure 112. Simplex reflections.

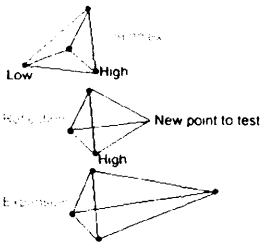


Figure 113. Simplex geometry and reflection/expansion moves.

The simplex algorithm navigates through the N-dimensional topography, by moving the point of the simplex, where the fitness is lowest, through the opposite face of the simplex to a higher point. These steps are called reflections, and are so constructed as to conserve the volume of the simplex and thus maintain its non-degeneracy. The method can expand the simplex when the search space is easy in one or more directions to take larger steps. When it reaches a ridge the method contracts itself in the transverse direction and tries to follow the ridge. The algorithm is as follows, Flannery, B. P. et al(1999):

1. Create a n-dimensional simplex where each of the (n+1) points is a vector holding the values of the 2 hyper parameters for each of the SVMs in the model.

2. For each point on the simplex, train the model using the values of the hyper parameters at this point and return the average RMS error that the resulting model has on the validation batches.

3. Reflect the point with the highest error through the centroid of the simplex and evaluate the error at this new point.

4. If the new point is the lowest in the simplex expand the simplex in this direction
   If the new point is the highest in the simplex contract the simplex towards the centroid
   else reflect.

Data reuse.

Having determined the optimum values of parameters C and σ_k using the simplex method. The whole data set, including cross validation(CV) can be used to train the model. The validation strategy can be viewed as setting the capacity of the model to

be trained rather than the settings of a trained model. As Table 18. confirms that when this extra training data is used the fit to unseen data improves.

Table 18. Comparison of SVM models where validation data is discarded after capacity is set and where validation data is reused as training data.

noise level	0%	5%	10%	15%
	Average error on testing batches			
Trained on training only	0.010	0.016	0.007	0.012
Trained on training and validation	0.008	0.015	0.007	0.012

Appendix D3: Hybrid modelling of a Streptomyces clavulingerus *batch cultivation - the results obtained by Hans Roubos.*

The method of hybrid modelling by Hans Roubos and the results obtained by him for the *Streptomyces clavulingerus* batch cultivation are reproduced here to facilitate comparison with the SVM methodology.

Methodology used by Hans Roubos.

The general hybrid modelling formulism was applied:

$$\frac{d\xi}{dt} = \underbrace{Kr(\xi,t)}_{kinetics} - \underbrace{D\xi - Q(\xi) + F}_{transport} \tag{14.8}$$

The matrix K was determined from a detailed metabolic model consisting of 85 reactions. An additional level of detail was that reactions were added and removed from the metabolic model on the basis of extra cellular metabolite concentrations in order to approximate the different regulation of enzymes in different metabolic phases.

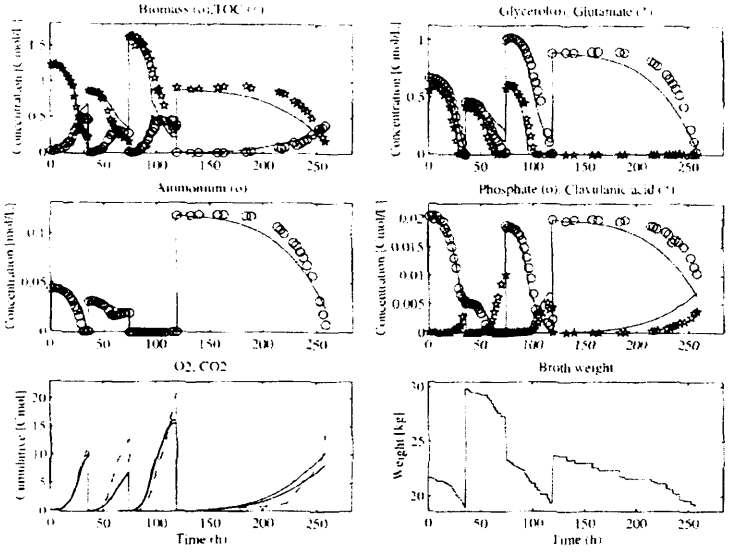
This left 5 independent kinetic functions; Glycerol uptake, Glutamate uptake, CA production, CA degradation and cell lysis.

These kinetic functions were modelled using 3 techniques:

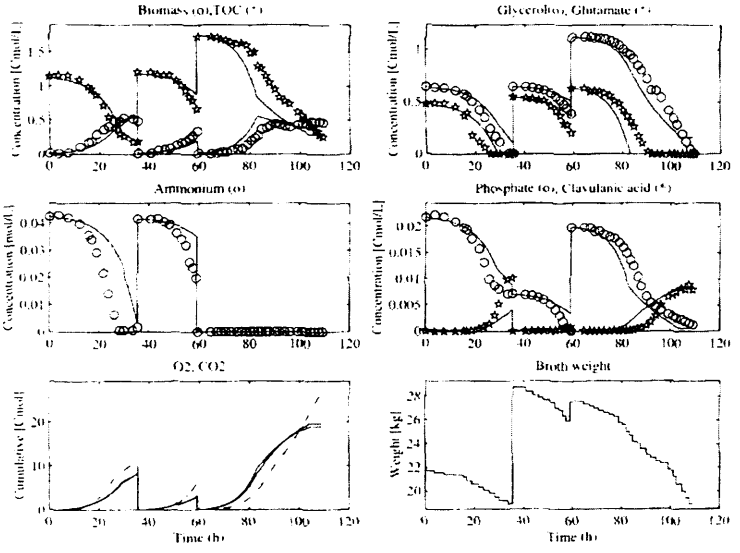
Conventional equations.

Feed forward Neural network models.

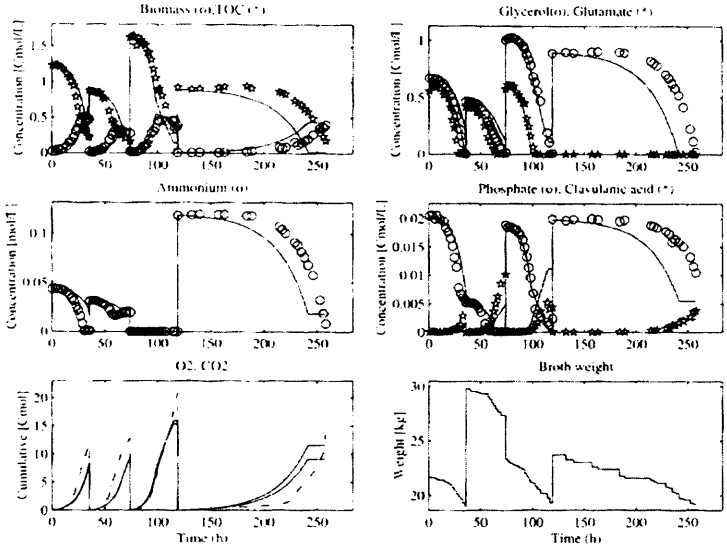
Takagi-Sugeno Fuzzy logic models.



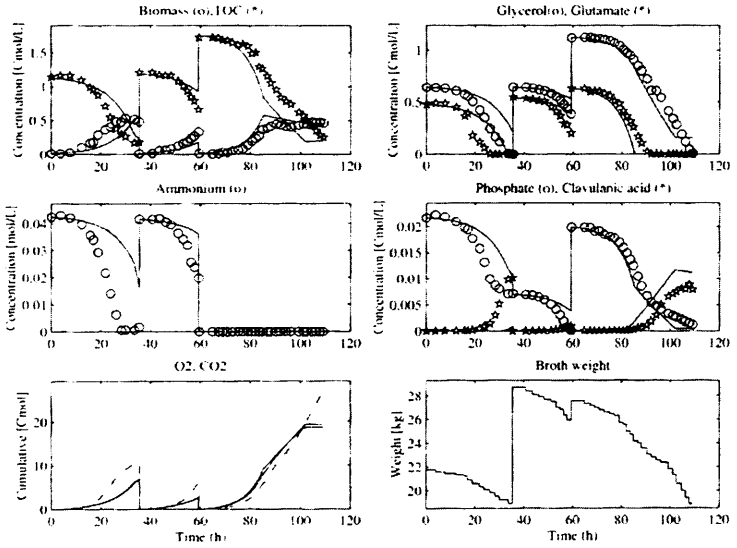
(a) Identification: model with conventional kinetic model.



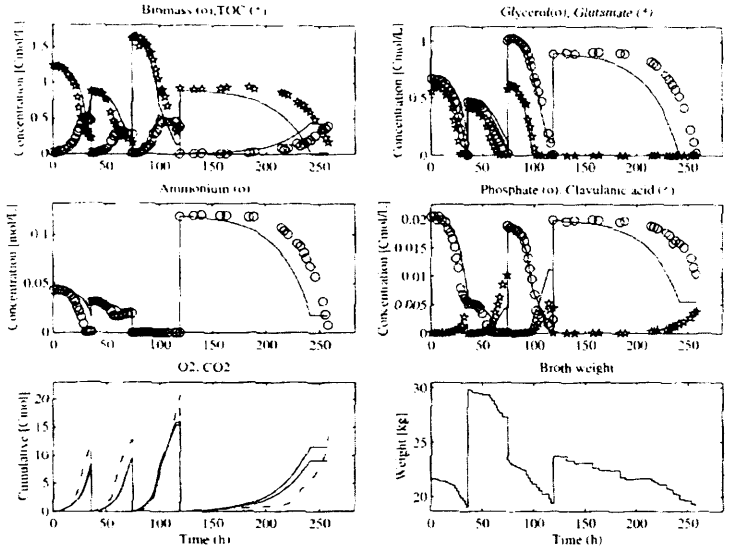
(b) Validation; model with conventional kinetic model.



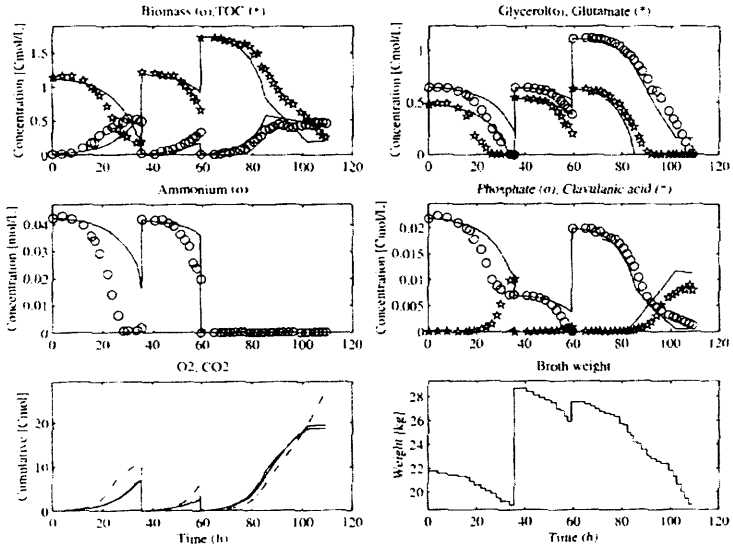
(c) Identification; model with neural network model.



(d) Validation; model with neural network model.



(e) Identification: model with fuzzy kinetic model.



(f) Validation: model with fuzzy kinetic model.

Figure 4.5. *Simulation using the training data – batch experiments B1, B3, B5 and B6 (left column) and the validation data – batch experiments B2, B7 and B8 (right column). Note that the individual batches are appended after each other.*

Figure 114. Results of hybrid modelling by Hans Roubos using 3 techniques. (Taken from Roubos, J. A.(2002)

Appendix E1: Gaussian processes.

A Gaussian process (MacKay, D. J. C.(2004); Rasmussen, C. E.(2003)) defines a joint distribution over a data set.

$$\begin{aligned} &inputs \quad X_{N\cdot 1} = \left[x^{(0)}, x^{(1)} \ldots x^{(N-1)}\right] \\ &outputs \quad \mathrm{t}_{N\cdot 1} = \left[t\left(x^{(0)}\right), t\left(x^{(1)}\right) \ldots t\left(x^{(N-1)}\right)\right] \end{aligned} \tag{15.1}$$

in the form of a multidimensional Gaussian:-

$$P\left(\mathrm{t}|C,\{X\}\right) = \frac{1}{Z}\exp\left(\frac{-1}{2}\left(\mathrm{t}-\mu\right)^T C^{-1}\left(\mathrm{t}-\mu\right)\right) \tag{15.2}$$

where the elements of C are calculated from a covariance function $C_{i,j} = Cf\left(x^i, x^j\right)$ such as:-

$$Cf\left(x^a, x^b\right) = \theta_1 \exp\left(\frac{-1}{2}\sum_{i=0}^{d-1}\frac{\left(x^a{}_i - x^b{}_i\right)^2}{r_i}\right) + \theta_2 \tag{15.3}$$

θ_1, θ_2 *and* $r \in \Re^d$ are hyper-parameters.

The probability of a new point output value y given inputs $x^{(N)}$ can then be found from Bayes theorem as:-

$$P\left(y|D, X^{(N)}\right) = \frac{P\left(\mathrm{t}_N|\{X_n\}\right)}{P\left(\mathrm{t}|\{X_{n-1}\}\right)} \tag{15.4}$$

Here t_N and X_n are the data set vector defined in (15.1) with the new $x^{(N)}$ and y values appended. i.e:

$$\begin{aligned} X_n &= \left[x^{(0)}, x^{(1)} \ldots x^{(N-1)}, x^N\right] \\ \mathrm{t}_N &= \left[t\left(x^{(0)}\right), t\left(x^{(1)}\right) \ldots t\left(x^{(N-1)}\right), y\right] \end{aligned} \tag{15.5}$$

By substituting (15.2) into (15.4) the conditional probability is obtained as:-

$$P\left(\mathrm{t}_N|D, x^{(N)}\right) = \frac{Z_{N-1}}{Z_N}\exp\left(-\frac{1}{2}\left(\mathrm{t}^T{}_N C_N^{-1}\mathrm{t}_N - \mathrm{t}^T{}_{N-1} C_{N-1}^{-1}\mathrm{t}_{N-1}\right)\right) \tag{15.6}$$

The expectation of a Gaussian is its mean and so this is given by:

$$\hat{y} = \hat{\mathrm{t}}_N = k_N^T C_{N-1}^{-1}\mathrm{t}_{N-1} \tag{15.7}$$

where $k_N^T = Cf\left(x^0, x^N\right) \ldots Cf\left(x^{N-1}, x^N\right)$. i.e. the covariance between the new point and the existing data points. The variance of the Gaussian is given by:-

$$\sigma_{t_N} = \kappa - k_N^T C_{N-1}^{-1} k_N \tag{15.8}$$

where $\kappa_N^T = Cf\left(x^N, x^N\right)$.

An example of a 2 dimensional Gaussian process regression is shown in Figure 115. below. Expected value is shown as dark surface, training data as dots and variance of the estimate as light surfaces.

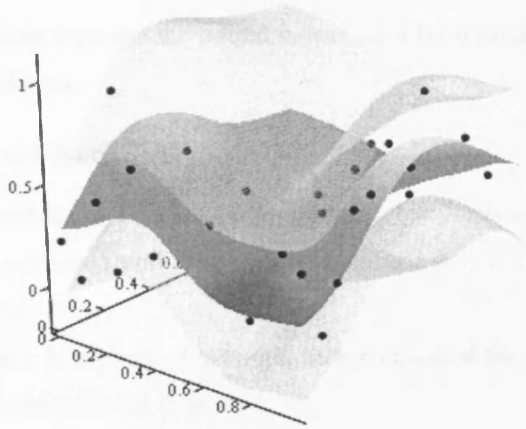


Figure 115. Fitting a Gaussian process to 2 dimensional data.
Dots are training data, dark surface is expected value, light surfaces are 'error bars'.

Appendix E2: Statistical tests.

Paired T-test

The paired t–test can be used to test whether two distributions consisting of matched pairs (X_i, Y_i) differ from each other in a significant way by giving the probability of the data given the null hypothesis that there is no difference between the two distributions. The t statistic is calculated as:

$$t = \left(\overline{X} - \overline{Y}\right) \sqrt{\frac{n(n-1)}{\sum_{i=1}^{n}\left(\hat{X}_i - \hat{Y}_i\right)^2}} \tag{15.9}$$

where

$$\hat{X}_i = X_i - \overline{X}$$

$$\hat{Y}_i = Y_i - \overline{Y}$$

The probability of that the data would have been obtained had null hypothesis held can be calculated from the student's t distribution with degrees of freedom equal to $\nu = N - 1$ or found in statistical tables. For the paired T test to be a valid test of statistical significance the following assumptions must hold:

1. The differences between the pairs must be approximately normally distributed.
2. The scale of the measurement of (X_i, Y_i) must have properties of an equal interval scale
3. The differences between the paired values have been randomly drawn from the source population;

The Wilcoxon Signed-Rank Test

The Wilcoxon Signed-Rank Test is a non parametric test for paired data which can be applied when assumptions 1 and 2 do not hold and the paired T-test cannot be used. The test is as follows:

1. The difference is calculated between each pair and if the difference is zero the pair is removed from the data set.

$$d_i = X_i - Y_i \tag{15.10}$$

2. Each pair is then ranked by the absolute value of the difference $rank(|d_i|)$
3. The sum is then calculated of the signed ranks.

$$w = \sum_{i=0}^{N-1} rank(|d_i|) \times sign(d_i) \tag{15.11}$$

4. The z statistic is calculated as:

$$z = \frac{w + c}{\sigma_w}$$

$$where\ c = \begin{cases} +0.5\ if(w < 0) \\ -0.5\ if(w > 0) \end{cases} \tag{15.12}$$

$$and\ \sigma_w = \sqrt{\frac{N(N+1)(2N+1)}{6}}$$

For N<10 the probability of the null hypothesis can be calculated by enumeration of all the possibilities. For example as shown in for N=3 there are 2^3=8 possible combinations depending on whether each pair is positive or negative. The right most

column shows the probability of obtaining a value of w greater than or equal to the w of that row value by pure chance.

Table 19 W values for a sample size of three

Rank				
1	*2*	*3*	*W*	*P(D≥W)*
+	+	+	+6	1/8
-	+	+	+4	1/4
+	-	+	+2	3/8
+	+	-	0	1
-	-	+	0	
-	+	-	-2	3/8
+	-	-	-4	1/4
-	-	-	-6	1/8

For N>10 the z statistic is distributed according to the standard normal distribution so the significance level for the null hypothesis can then be calculated from this or from statistical tables.

The assumptions of the Wilcoxon signed rank test are very weak:

- The differences between the paired values have been randomly drawn from the source population;
- That the variables (X_i, Y_i) can in principle be measured on a continuous scale so that the differences can be meaningfully ordered.

The Kolmogorov-Smirnov test for normality.

The Kolmogorov-Smirnov test measures the maximum absolute difference between two cumulative distributions. To test for normality the cumulative empirical distribution of the data (Y_i) as estimated from equation (15.13) below:

$$F_{Model\ type}(x) = \frac{1}{n}\sum_{i=0}^{n-1}\begin{cases}1 & if(Y_i \le x \\ 0 & otherwise\end{cases} \qquad (15.13)$$

Is compared with the cumulative normal distribution where mean (μ) and standard deviation (σ) of the normal distribution are equal to the sample mean and standard deviation of the data (Y_i):

$$c_{norm}(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right) da \qquad (15.14)$$

The Kolmogorov-Smirnov test measures the maximum difference between the two distributions. The two one-sided Kolmogorov-Smirnov test statistics are given by

$$\begin{aligned} D_n^+ &= \max\left(F(x) - c_{norm}(x)\right) \\ D_n^- &= \max\left(c_{norm}(x) - F(x)\right) \end{aligned} \tag{15.15}$$

The Kolmogorov-Smirnov statistic has an asymptotic null distribution given by equation (15.16) where n is the number of data points. An algorithm for calculating the significance(of the null hypothesis that the distributions are the same) can be found in Flannery, B. P. et al(1999) .

$$\lim_{n\to\infty} \mathrm{P}\left(\sqrt{\frac{\mathrm{n}^2}{2\mathrm{n}}} D_n \le L(D)\right) = L(D)$$

where (15.16)

$$L(D) = 1 - 2\sum_{i=1}^{\infty} (-1)^{i-1} e^{-2i^2D^2}$$

A large value of D, and therefore a small value of $L(D)$, indicates that the null hypothesis is unlikely to be true, whereas small D values support the null hypothesis.

Appendix E2: MathCAD code for Schnabel, R. B. et al(1990) Generalised Cholsky decomposition.

```
bnabel(A) :=
    n ← rows(A)
    m ← cols(A)
    L ← identity(n)·identity(n)
    gamm ← max(|diagV(|A|)|)
    tau ← macheps^(1/3)
    norm_A ← max(sumc(|A|))
    delta ← max(macheps·norm_A, macheps)
    Pprod ← identity(n)
    deltaprev ← 0
    if n > 2
        for k ∈ 0..n - 3
            if min((diagV(submatrix(A, k + 1, n - 1, k + 1, n - 1)))^T - sq(submatrix(A, k, k, k + 1, n - 1)) / A_{k,k}) < tau·gamm·min(|eigenvals(submatrix(A, k + 1, n - 1, k + 1, n - 1))|)
                dmax ← max(indd(diagV(submatrix(A, k, n - 1, k, n - 1))))_{0,0}
                if A_{k+dmax,k+dmax} > A_{k,k}
                    P ← identity(n)
                    P ← (P)^T
                    Ptemp ← P^(k)
                    P^(k) ← P^(k+dmax-1)
                    P^(k+dmax-1) ← Ptemp
                    P ← (P)^T
                    A ← P·A·P
                    L ← P·L·P
                    Pprod ← P·Pprod
                g ← zeros[n - (k - 1), 1]
                for i ∈ k..n - 1
                    sum1 ← 0 if i = 0
                    sum1 ← sumc((|submatrix(A, i, i, k, i - 1)|)^T)_{0,0} otherwise
                    sum2 ← 0 if i = n - 1
                    sum2 ← sumc((|(|submatrix(A, i + 1, n - 1, i, i)|)|)^T)_{0,0} otherwise
                    g_{i-(k-1)} ← A_{i,i} - sum1 - sum2
                gmax ← max(indd(g))
                if gmax ≠ k
                    P ← identity(n)
                    P ← (P)^T
                    Ptemp ← P^(k)
                    P^(k) ← P^(k+dmax-1)
                    P^(k+dmax-1) ← Ptemp
                    P ← (P)^T
                    A ← P·A·P
                    L ← P·L·P
                    Pprod ← P·Pprod
                normj ← sumc(|submatrix(A, k + 1, n - 1, k, k)|)
                delta ← max(0, deltaprev - A_{k,k} + max(normj, tau·gamm))
                if delta > 0
                    A_{k,k} ← A_{k,k} + delta
                    deltaprev ← delta
```

```
A_{k,k} ← sqrt(A_{k,k})
L_{k,k} ← A_{k,k}
for i ∈ k + 1 .. n - 1
    A_{i,k} ← A_{i,k} / L_{k,k}  if L_{k,k} > machep
    L_{i,k} ← A_{i,k}
    temp ← submatrix(A, i, i, k + 1, i) - L_{i,k} (submatrix(L, k + 1, i, k, k))^T
    for index ∈ 0 .. cols(temp) - 1
        A_{i, index + k + 1} ← temp_{0, index}
    A_{i,i} ← 0  if A_{i,i} < 0
A_{n-2, n-1} ← A_{n-1, n-2}
eig ← eigenvals(submatrix(A, n - 2, n - 1, n - 2, n - 1))
dlist ← augment[0, deltaprev, -min(eig) + tau·max[(1/(1 - tau)) max(eig) - min(eig), gamm]]^T
delta ← (dlist_{0,0})  if dlist_{0,0} > dlist_{1,0}
delta ← dlist_{1,0}  otherwise
delta ← dlist_{2,0}  if delta < dlist_{2,0}
if delta > 0
    A_{n-2,n-2} ← A_{n-2,n-2} + delta
    A_{n-1,n-1} ← A_{n-1,n-1} + delta
A_{n-2,n-2} ← sqrt(A_{n-2,n-2})
L_{n-2,n-2} ← A_{n-2,n-2}
A_{n-1,n-2} ← A_{n-1,n-2} / L_{n-2,n-2}
L_{n-1,n-2} ← A_{n-1,n-2}
A_{n-1,n-1} ← sqrt(A_{n-1,n-1} - (L_{n-1,n-2})^2)
L_{n-1,n-1} ← A_{n-1,n-1}
(Pprod^T L^T Pprod^T)^T
```

Appendix E3: MathCAD code for Gill, P. E. et al(1981) Generalised Cholsky decomposition.

$$
\begin{aligned}
&\text{gill}(A):= \\
&\quad n \leftarrow \text{rows}(A) \\
&\quad R \leftarrow \text{identity}(n) \\
&\quad L \leftarrow R \quad R \\
&\quad \text{norm\_A} \leftarrow \max\left(\text{sumc}\left(\left|A^{\langle}\right|\right)\right) \\
&\quad \text{gamm} \leftarrow \max\left[\text{diag}\left(\left|A^{\langle}\right|\right)\right] \\
&\quad \text{delta} \leftarrow \max(\text{macheps}\cdot\text{norm\_A},\text{macheps}) \\
&\quad \text{for } j \in 0..n-1 \\
&\qquad \text{theta\_j} \leftarrow 0 \\
&\qquad \text{for } i \in 0..n-1 \\
&\qquad\quad \text{sum} \leftarrow 0 \\
&\qquad\quad \text{for } k \in 0..i-1 \qquad \text{if } i>0 \\
&\qquad\qquad \text{sum} \leftarrow \text{sum} + R_{k,i}\cdot R_{k,j} \\
&\qquad\quad R_{i,j} \leftarrow \frac{\left(A_{i,j}-\text{sum}\right)}{R_{i,i}} \\
&\qquad\quad \text{theta\_j} \leftarrow \left(A_{i,j}-\text{sum}\right) \text{ if } \left(A_{i,j}-\text{sum}\right) > \text{theta\_j} \\
&\qquad\quad R_{i,j} \leftarrow 0 \text{ if } i>j \\
&\qquad \text{sum} \leftarrow 0 \\
&\qquad \text{for } k \in 0..j-1 \qquad \text{if } j>0 \\
&\qquad\quad \text{sum} \leftarrow \text{sum} + \left(R_{k,j}\right)^2 \\
&\qquad \text{phi\_j} \leftarrow A_{j,j}-\text{sum} \\
&\qquad \text{xi\_j} \leftarrow \max\left(\left(\left|\text{submatrix}(A,j+1,n-1,j,j)\right|^{\langle}\right)\right) \text{ if } (j+1)\le n-1 \\
&\qquad \text{xi\_j} \leftarrow \left|A_{n-1,j}\right| \text{ otherwise} \\
&\qquad \text{beta\_j} \leftarrow \sqrt{\max\left(\text{gamm},\frac{\text{xi\_j}}{n},\text{macheps}\right)} \\
&\qquad E_{j,j} \leftarrow \text{delta}-\text{phi\_j} \text{ if } \text{delta} \ge \max\left(\left|\text{phi\_j}\right|,\frac{\text{theta\_j}^2}{\text{beta\_j}^2}\right) \\
&\qquad \text{otherwise} \\
&\qquad\quad E_{j,j} \leftarrow \left|\text{phi\_j}\right|-\text{phi\_j} \text{ if } \left|\text{phi\_j}\right| \ge \max\left(\frac{\text{theta\_j}^2}{\text{beta\_j}^2},\text{delta}\right) \\
&\qquad\quad E_{j,j} \leftarrow \frac{\text{theta\_j}^2}{\text{beta\_j}^2}-\text{phi\_j} \text{ if } \left(\frac{\text{theta\_j}^2}{\text{beta\_j}^2}\right) \ge \max\left(\text{delta},\left|\text{phi\_j}\right|\right) \text{ otherwise} \\
&\qquad R_{j,j} \leftarrow \sqrt{A_{j,j}-\text{sum}+E_{j,j}} \\
&\quad R^T
\end{aligned}
$$

Appendix E4: MathCAD code for Hamiltonian Monte Carlo (Duane et al(1987)).

```
Hamiltonian(x_, p, L, τmax, ε) :=
    M ← x
    k ← 0
    g ← gradE(x_)
    E ← findE(x_)
    for i ∈ 1, 2.. L
        p ← rnorm(rows(x_), 0, 1)
        H ← (p^T · p) / 2 + E
        xnew ← x_
        gnew ← g
        τ ← ceil(rnd(τmax))
        for tau ∈ 1, 2.. τ
            p ← p − ε · gnew / 2
            xnew ← xnew + ε · p
            gnew ← gradE(xnew)
            p ← p − ε · gnew / 2
        Enew ← findE(xnew)
        Hnew ← (p^T · p) / 2 + Enew
        dH ← Hnew − H
        accept ← 1 if (dH < 0)
        otherwise
            accept ← 1 if (rnd(1) < exp(−dH))
            accept ← 0 otherwise
        if accept = 1
            g ← gnew
            x_ ← xnew
            E ← Enew
            k ← k + 1
            M^⟨k⟩ ← x_
    M
```

findE(x) := fo(x) should return the Hessian of the distribution of interest at point x

Appendix E5: Reversible jump Markov chain Monte Carlo.

This method uses a Markov chain to sample from an extended 'supermodel', where model structure becomes part of the state space of the Markov chain and thus marginalise with respect to model structure. A Markov chain is constructed which can 'jump' between models with parameter spaces of different dimension in a flexible way while retaining detailed balance, which ensures the correct limiting distribution provided the chain is irreducible and a-periodic. However the computational burdens and design issues involved are considerably greater than those of simple Monte Carlo with a fixed model structure.

Pseudocode for MJMCMC is as follows:-

1. Propose moving to a new model k' by drawing an integer from some arbitrary distribution $j(k',k)$
2. *if* $(k'=k)$ do a standard metropolis step
   otherwise

   a. Generate a random vector u of length $\max(0,\dim(w^{k'})-\dim(w^{k}))$ from an arbitrary distribution $q_k(u)$

   b. Generate a new state $\psi'=(k',w^{k'})$ from some invertible deterministic function of the current state and the random numbers $\psi'=g_{k\to k'}(w^k,u)$ where $g_{k'\to k}=g^{-1}_{k\to k'}$ and $\dim(w^{k'})+\dim(u^{k'})=\dim(w^{k})+\dim(u^{k})$

   c. Accept the new state with probability $\min(1,A)$ where A is given by the following.

$$A=\frac{P(\psi')}{P(\psi)}\frac{j(k,\psi')}{j(k',\psi)}\frac{q_{k'}(u')}{q_k(u)}\left|\det J\right|$$

Comments on the algorithm.

The first term $\frac{P(\psi')}{P(\psi)}$ of the expression for A is the ratio of the probability according to the distribution of interest of the proposed point to the current point. In the current context that would be the un normalised posterior $P(D|w^k, H_k)P(w^k|H_k)P(H_k)$.

The second term $\frac{j(k,\psi')}{j(k',\psi)}$ is the probability ratio of the chosen move from model $k \to k'$ to the reverse move.

The third term is the ratio of proposal distributions. Note here that if $\dim(w^{k'}) > \dim(w^k)$ no random variables u need to be generated and hence the reverse move is deterministic and so $q_{k'}(u') = 1$.

The final term is the absolute value of the determinant of the Jacobian matrix of the mapping function. $J = \frac{\partial g_{k\to k'}(w^k,u)}{\partial(w^k,u)}$ and is required due to the change in variables due to the mapping function.[65] That is since a change in variable of an n dimensional integral is given by

$$\int_\Omega f(x)d^n x = \int_{u(\Omega)} f(u^{-1}(u))\left|\det\frac{dx}{du}\right| d^n u$$

$$\frac{\partial g_{k\to k'}(w^k,u)}{\partial(w^k,u)} = \begin{pmatrix} \frac{\partial w_1^{k'}}{\partial w_1^k} & \cdots & \frac{\partial w_1^{k'}}{\partial w_n^k} & \frac{\partial w_1^{k'}}{\partial u_1} & \cdots & \frac{\partial w_1^{k'}}{\partial u_n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial w_n^{k'}}{\partial w_1^k} & \cdots & \frac{\partial w_n^{k'}}{\partial w_n^k} & \frac{\partial w_n^{k'}}{\partial u_1} & \cdots & \frac{\partial w_n^{k'}}{\partial u_n} \\ \frac{\partial u_1'}{\partial w_1} & \cdots & \frac{\partial u_1'}{\partial w_n^k} & \frac{\partial u_1'}{\partial u_1} & \cdots & \frac{\partial u_1'}{\partial u_n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial u_n'}{\partial w_1} & \cdots & \frac{\partial u_n'}{\partial w_n^k} & \frac{\partial u_n'}{\partial u_1} & \cdots & \frac{\partial u_n'}{\partial u_n} \end{pmatrix}.$$

[65] "Change of Variables in Integrals." Kaplan(1984)

Bibliography.

Reference List

1. Abdel-Fatta, Y. R., Saeed, H. M., Gohar, Y. M., and El-Baz, M. A.(2004).Improved production of Pseudomonas aeruginosa uricase by optimization of process parameters through statistical experimental designs. Process Biochemistry.

2. Akaike, H.(1973).Information Theory and an Extension of the Maximum Likelihood Principle. 267-281.

3. Altýntas,, M. M., Kýrdar, B., O¨nsan, Z. I., and Ulgen, K.(2001).Plasmid stability in a recombinant S. Cerevisia strain secreting a bifunctional fusion protein. Journal of Chemical Technology and Biotechnology. 76:612-618.

4. Ando, Shin, Sakamoto, Erina, and Iba, Hitoshi(2002).Evolutionary modeling and inference of gene network. Information Sciences. 145:237-259.

5. Bailey, J. E.(1998).Mathematical modeling and analysis in biochemical engineering: past accomplishments and future opportunities. Biotechnology Progress. 14:8-20.

6. Balboni, M. L(2003).Process Analytical Technology. Concepts and Principles. Pharmaceutical Technology.

7. Baltes, M., Schneider, R., Sturm, C., and Reuss, M.(2005).Optimal Experimental Design for Parameter Estimation in Unstructured Growth Models. Biotechnology Progress. 10:480-488.

8. Basheer, I. A. and Hajmeer, M.(2000).Artificial neural networks: fundamentals, computing, design and application. Journal of Microbiology Methods. 43:3-31.

9. Baughman, D. and Liu, Y.(1995).Neural networks in bioprocessing and chemical engineering. New York Academic Press,

10. Berger, J. O(1980).Statistical Decision Theory and Bayesian Analysis. 2 0-387-96098-8

11. Bernard, Olivier and Bastin, Georges(2005).On the estimation of the pseudo-stoichiometric matrix for macroscopic mass balance modelling of biotechnological processes. Mathematical Biosciences. 193:51-77.

12. Binder, T., Cruse, C. A., Cruz, V., and Marquardt, W.(2000).Dynamic optimization using a wavelet based adaptive control vector

parameterization stratergy. Computers & Chemical Engineering. 24:1201-1207.

13. Bogaerts, Ph, Delcoux, J.-L., and Hanus, R.(2003).Maximum likelihood estimation of pseudo-stoichiometry in macroscopic biological reaction schemes. Chemical Engineering Science. 58:1545-1563.

14. Bogaerts, Ph and Wouwer, A. V. A.(2004).Parameter identification for state estimation--application to bioprocess software sensors. Chemical Engineering Science. 59:2465-2476.

15. Bortz, D. M. and Nelson, P. W.(2004).Sensitivity analysis of a nonlinear lumped parameter model of HIV infection dynamics. Bulletin of Mathematical Biology. 66:1009-1026.

16. Broomhead, D. S and Lowe, D(1988).Multivariable functional interpolation and adaptive networks. Complex Systems. 2:321-355.

17. Butler, Michael J., Bruheim, Per, Jovetic, Srdjan, Marinelli, Flavia, Postma, Pieter W., and Bibb, Mervyn J.(1-10-2002).Engineering of Primary Carbon Metabolism for Improved Antibiotic Production in Streptomyces lividans. Applied and Environmental Microbiology. 68:4731-4739.

18. Cai, C. Z., Wang, W. L., Sun, L. Z., and Chen, Y. Z.(2003).Protein function classification via support vector machine approach. Mathematical Biosciences. 185:111-122.

19. Cannizzaro, C, Valentinotti, S., and Von Stockar, U.(2004).Control of yeast fed-batch process through regulation of extracellular ethanol concentration. Bioprocess and Biosystems Engineering. 26:377-383.

20. Cao, Hongqing, Yu, Jingxian, Kang, Lishan, Chen, Yuping, and Chen, Yongyan(30-3-1999).The kinetic evolutionary modeling of complex systems of chemical reactions. Computers & Chemistry. 23:143-152.

21. Carrasco, E. F. and Banga, J. R.(1997).Dynamic optimizeation of batch reactors using adaptive stochastic algorithms. Industrial Engineering Chemical research. 36:2252-2261.

22. Chang, C. C. and Lin, C.(2001).LIBSVM : a library for support vector machines.

23. Chen, L. and Bastin, G.(1996).Structural identifiability of the yield coefficients in bioprocess models when the reaction rates are unknown. Mathematical Biosciences. 132:35-67.

24. Chen, L., Bernard, O., Bastin, G., and Angelov, P.(2000).Hybrid modelling of biotechnological processes using neural networks. Control Engineering Practice. 8:821-827.

25. Clyde, M. A(2001).Experimental design: Bayesian designs. International Encyclopedia of the Social & Behavioral Sciences. 5075-5081.

26. Cockshott, A. R. and Hartman, B. E.(2001). Improving the fermentation medium for Echinocandin B production part II: Particle swarm optimization. Process Biochemistry. 36:661-669.

27. Cooney, M. J. and McDonald, K. A.(1995).Optimal dynamic experiments for bioreactor model discrimination. Applied Microbiology and Biotechnology. 43:826-837.

28. Cortes, C. and Vapnik, V.(1995).Support-vector networks. Machine Learning. 20:273-297.

29. Cramer, Nichael Lynn(1985). A Representation for the Adaptive Generation of Simple Sequential Programs.

30. Dickmanns, D., Winklhofer, A, and Schmidhuber, J(1987). Der genetische Algorithmus: Eine Implementierung in Prolog. Fortgeschrittenenpraktikum.

31. Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D.(1987).Hybrid Monte Carlo. Physics Letters. 55:2774-2777.

32. Dunn, J. I, Jiarcaron, E., Parcaron, E., Heinzle, E., and Ingham, J.(2003).Biological Reaction Engineering: Dynamic Modelling Fundamentals with Simulation Examples. 2.Wiley, 3-527-30759-1

33. Eberhart, R. B., Shi, Y., and Kennedy, J.(2001).Swarm Intelligence. Morgan Kaufmann, 1558605959

34. Federov, V.(1972).Theory of optimal experiments. Academic Press,

35. Feyo de Azevedo, S., Dahm, B., and Oliveira, F. R.(20-5-1997).Hybrid modelling of biochemical processes: A comparison with the conventional approach. Computers & Chemical Engineering. 21:S751-S756.

36. Flannery, B. P, Press, W. H, Vetterling, W. T, Press, W. B, and Teukolsky, W. A.(1-1-1999).Numerical Recipes in C : The Art of Scientific Computing. 2nd edition.Cambridge University Press, 0521431085

37. Foss, B. A., Johansen, T. A., and Sorensen, A. V.(1995).Nonlinear predictive control using local models -- applied to a batch fermentation process. Control Engineering Practice. 3:389-396.

38. Galvanauskas, V., Simutis, R., and Lubbert, Andreas(1997).Model-based design of biochemical processes: simulation studies and experimental tests. Biotechnology Letters. 19:1043-1047.

39. Galvanauskas, V., Simutis, R., and Lubbert, Andreas(2004).Hybrid process models for process optimisation, monitoring and control. Bioprocess and Biosystems Engineering. 26:393-400.

40. Galvanauskas, V., Simutis, R., Volk, N., and Lubbert, Andreas(1998).Model based design of a biochemical cultivation process. Bioprocess and Biosystems Engineering. 18:227-234.

41. Garcia, Carlos E., Prett, David M., and Morari, Manfred(1989).Model predictive control: Theory and practice--A survey. Automatica. 25:335-348.

42. Gawthrop, P. J., Ballance, D., and Chen, W. H(2003).Optimal control of nonlinear systems: a predictive control approach. Automatica. 39:633-641.

43. Gelman, A. and Meng, X. L(1998).Simulating normalizing constants: from importance sampling to bridge sampling to path sampling. Statistical science. 13:163-185.

44. Gill, P. E., Murray, W., and Wright, M. H.(1981).Modified Cholesky Algorithm. 108-111.

45. Gill.J. and King.G.(1-8-2004).What to do When Your Hessian is Not Invertible: Alternatives to Model Respecification in Nonlinear Estimation. Sociological Methods and Research.

46. Goldberg, D. E(1989).Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Professional, 0201157675

47. Green, P(1995).Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Biometrica. 82:711-732.

48. Grosman, Benyamin and Lewin, Daniel R.(15-11-2004).Adaptive genetic programming for steady-state process modeling. Computers & Chemical Engineering. 28:2779-2790.

49. Haag, Jens E., Wouwer, Alain Vande, and Bogaerts, Philippe(2005).Systematic procedure for the reduction of complex biological reaction pathways and the generation of macroscopic equivalents. Chemical Engineering Science. 60:459-465.

50. Hastings, W. K(1970).Monte Carlo sampling methods using Markov chains and their applications. Biometrika. 57:97-109.

51. Haykin, S.(1999).Neural Networks - A Comprehensive Foundation. Prentice Hall,

52. Hinchliffe, Mark P. and Willis, Mark J.(15-12-2003).Dynamic systems modelling using genetic programming. Computers & Chemical Engineering. 27:1841-1854.

53. Hinton, G. A. and Sejnowski, T. J.(1986).Learning and relearning in boltzmann machines. Parallel Distributed Processing. 1:

54. Hodge, D. B. and Karim, M. N(2002).Modeling and Advanced Control of Recombinant Zymomonas mobilis Fed-Batch Fermentation. Biotechnology Progress. 18:572-579.

55. Hodgson, B. J., Taylor, C. N, Ushio, M, Leigh, J. R., Kalganova, T., and Baganz, F.(2004).Intelligent modelling of bioprocesses: a comparison of structured and unstructured approaches. Bioprocess and Biosystems Engineering. 26:353-359.

56. Hopfield, J. J.(1982).Neural networks and physical systems with emergent collective computational abilities. Proceedings of the national acadamy of science. 79:2554-2558.

57. Hornik, K.(1991).Approximation capabilities of multilayer feedforward networks. NeuralNetworks. 2:359-366.

58. Howard Anton(1-1-2000).Elementary Linear Algebra. 8.John Wiley & Sons Inc, 0471170550

59. James, Scott, Legge, Raymond, and Budman, Hector(2002).Comparative study of black-box and hybrid estimation methods in fed-batch fermentation. Journal of Process Control. 12:113-121.

60. Jang, Jae Deog and Barford, John P.(2000).An unstructured kinetic model of macromolecular metabolism in batch and fed-batch cultures of hybridoma cells producing monoclonal antibody. Biochemical Engineering Journal. 4:153-168.

61. Jaynes E.T.(2003).Probability Theory : The Logic of Science. Cambridge University Press, 0521592712

62. Johansen, Tor A. and Foss, Bjarne A.(1998).ORBIT - operating-regime-based modeling and identification toolkit. Control Engineering Practice. 6:1277-1286.

63. Julier, Simon, Uhlmann, Jeffrey, and Durrant-Whyte, Hugh F.(2000).New method for the nonlinear transformation of means and covariances in filters and estimators. IEEE Transactions on Automatic Control. 45:477-482.

64. Kaiser, H. F. and Dickman, K.(1-1-1962).Sample and Population Score Matrices and Sample Correlation Matrices From an Arbitrary Population Correlation Matrix". Psychomerika. 27:179-182.

65. Kalil, S. J, Maugeri, F., and Rodrigues, M. I.(2000).Response surface analysis and simulation as a tool for bioprocess design and optimization. Process Biochemistry.

66. Kalman, R. E.(1960).A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME--Journal of Basic Engineering. 82:35-45.

67. Kaplan, W(1984).Change of Variables in Integrals. 3:238-245.

68. Karnin, E. D(1990).A Simple Procedure for Pruning Back-Propagation Trained Neural Networks. IEEE Transactions on Neural Networks. 1:239-245.

69. Kassam, S. A. and Cha, I.(1995).Interference cancellation using radial basis function networks. Signal Processing. 47:247-268.

70. Kennedy, M. J and Spooner, N. R.(1996).Using fuzzy logic to design fermentation media: a comparison to neural networks and factorial design. Biotechnology Techniques. 10:47-52.

71. King, R. and Budenbender, C.(1997).A structured mathematical model for a class of organisms: II. Application of the model to other strains. Journal of Biotechnology. 52:235-244.

72. Kirkpatrick, S, Gelat, C. D., and Vecchi, M. P(1983).Optimization by Simulated Annealing. Science. 220:671-680.

73. Klein, D(2005). Lagrange Multipliers without Permanent Scarring.

74. Knightes C.D and Peters C.A(20-7-2000).Statistical Analysis of Nonlinear Parameter Estimation for Monod Biodegradation Kinetics Using Bivariate Data. Biotechnology and Bioengineering. 69:

75. Knuth, D. E.(1981).Seminumerical Algorithms. The art of Computer Programming. Addison-Wesley,

76. Koza, J. R.(1992).Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems). MIT Press, 0262111705

77. Kullback, S. and Leibler, R. A.(1951).On information and sufficiency. Annals of MathematicalStatistics. 22:79-86.

78. Kwok, J. T(2000).The evidence framework applied to support vector machines. IEEE Transactions on Neural Networks. 11:1162-1173.

79. Lee, D. S., Jeon, C. O, Park, J. M., and Chang, K. S.(2002).Hybrid neural network modeling of a full-scale industrial wastewater treatment process. Biotechnology and Bioengineering. 78:670-682.

80. Luenberger, D G.(1966).Observers for multivariable systems. Transactions of automatic Control. 11:190-197.

81. Luke, S. and Spector, L.(1997).A Comparison of Crossover and Mutation in Genetic Programming.

82. MacKay, David. J. C(29-6-2004).Information Theory, Inference, and Learning Algorithms. Cambridge University Press,

83. Marcus Hutter(2004).Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability. Springer,

84. Marenbach, P.(1998).Using prior knowledge and obtaining process insight in data based modelling of bioprocesses. Systems analysis modeling and simulation. 31:39-59.

85. Matsumoto, M. and Nishimura, T.(1998).Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Trans.on Modeling and Computer Simulation. 8:3-30.

86. McKay, Ben, Willis, Mark, and Barton, Geoffrey(13-6-1997).Steady-state modelling of chemical process systems using genetic programming. Computers & Chemical Engineering. 21:981-996.

87. Mekarapiruk, W. and Luus, R.(2000).Optimal control by iterative dynamic programming with deterministic and random candidates for control. Computers & Chemical Engineering. 39:84-91.

88. Meng, X. L and Wong, W. H.(1996).Simulating ratios of normalizing constants via a simple identity: A theoretical exploration. Statistical science. 6:831-860.

89. Metropolis, Rosenbluth, A. W., Rosenbluth, M. N, Teller, A. H, and Teller, E(1953).Equations of state calculations by fast computing machines. Journal of Chemical Physics. 21:1087-1092.

90. Montgomery, DC(1991).Design and analysis of experiments. Wiley, New York.

91. Moody, J. E. and Darken, C. J.(1989).Fast learning in networks of locally-tuned processing units. Neural Computing. 1:281-294.

92. Moriyama, H. and Shimizu, K(2005).On line optimisation of culture temperature for ethanol fermentation using genetic programming. Journal of Chemical Technology and Biotechnology. 66:222-

93. Muehlenbein, H(1991).Evolution in time and space - the parallel genetic algorithms. 316-338.

94. Muller, K. R., Smola, A. J., Ratsch, G., Scholkopf, B., Kohlmorgen, J., and Vapnik, V.(1997).Predicting time series with support vector machines. In Proceedings of the International Conference on Articial Neural Networks.

95. Nakashima, N., Mitani, Y., and Tamura, T.(2005).Actinomycetes as host cells for production of recombinant proteins. Microbial Cell Factories. 4:1-7.

96. Nandi, Somnath, Badhe, Yogesh, Lonari, Jayaram, Sridevi, U., Rao, B. S., Tambe, Sanjeev S., and Kulkarni, Bhaskar D.(15-2-2004).Hybrid process modeling and optimization strategies integrating neural networks/support vector regression and genetic algorithms: study of benzene isopropylation on Hbeta catalyst. Chemical Engineering Journal. 97:115-129.

97. Neal, R. M(1996a).Bayesian Learning for Neural Networks. Springer,

98. Neal, R. M(2001).Annealed importance sampling. Statistics and Computing. 11:125-139.

99. Neal, R. M.(1992).Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical Report CRG-TR-92-1:

100. Neal, R. M.(1996b).Sampling from multimodal distributions using tempered transition. Statistics and Computing. 6:353-366.

101. Nelder, J. A. and Mead, R.(1965).The downhill simplex method. Computer Journal. 7:308-319.

102. Nielsen, J.(2001).Metabolic Engineering. Applied Microbiology and Biotechnology. 55:263-283.

103. Nielsen, J., Pedersen, A. G., Strudsholm, K., and Villadsen, J.(1990).Modeling fermentations with recombinant microorganisms: Formulation of a structured model. Biotechnology and Bioengineering. 37:802-808.

104. Oliveira, M. F.(1998).Supervision, Control and Optimization of Biotechnological Processes Based on Hybrid Models.

105. Oliveira, R.(15-5-2004).Combining first principles modelling and artificial neural networks: a general framework. Computers & Chemical Engineering. 28:755-766.

106. Orderud, F.(2005).Autonomous Helicopter Navigation.

107. Painton, L. A and Diwekar, U. M(1995).Stochastic annealing for synthesis under uncertainty. European Journal of Operational Research. 83:489-502.

108. Palsson, B.(2002).In silico biology through 'omics'. Nature Biotechnology. 20:649-650.

109. Pitt, M. A. and Myung, I. J(2002).When a good fit can be bad. Trends in Cognitive Science. 6:421-425.

110. Plackett, R. L. and Burman, J. P.(1946).The design of optimum multifactorial experiments. Biometrika. 33:305-325.

111. Platt, J.(1999).Fast training of support vector machines using sequential minimal optimisation. 185-208.

112. Poggio, T. and Girosi, F.(1989).A theory of networks for approximation and learning. Technical report AI. 1140-

113. Poggio, T. and Girosi, F.(1990).Regularization algorithms for learning that are equivalent to multilayer networks. Science. 247:978-982.

114. Prechelt, L.(1998).Automatic early stopping using cross validation: quantifying the criteria. Neural Networks. 4:761-767.

115. Preuss, K., Le Mann, M. V., Proth, J., and Pingaud, H.(2000).Modelling and predictive control of fed-batch yeast growth on industrial plant scale. Chemical Engineering Journal. 78:53-59.

116. Rasmussen, C. E.(2003).Gaussian Processes to Speed up Hybrid Monte Carlo for Expensive Bayesian Integrals. 7:651-659.

117. Rasmussen, C. E. and Ghahramani, Z.(2003).Bayesian Monte Carlo. Advances in Neural Information Processing Systems. 15:

118. Roels, J. A.(1983).Energetics and kinetics in biotechnology. Elsevier,

119. Romanenko, Andrei and Castro, Jose A. A. M.(15-3-2004).The unscented filter as an alternative to the EKF for nonlinear state estimation: a simulation case study. Computers & Chemical Engineering. 28:347-355.

120. Roubos, J. A.(2002).Bioprocess modeling and optimization - Fed batch clavulanic acid production by Streptomyces clavuligerus. Delft University of Technology, Delft, The Netherlands, 90-6464-868-9

121. Roubos, J. A., Krabben, P., Luiten, R., Babuska, R., and Heijnen, J. J.(2001).A semi-stoichiometric model for a *Streptomyces* fed-batch cultivation with multiple feeds. 299-304.

122. Roubos, J. A., van Straten, G., and van Boxtel, A. J. B.(22-1-1999).An evolutionary strategy for fed-batch bioreactor optimization; concepts and performance. Journal of Biotechnology. 67:173-187.

123. Royce, P. N.(1993).A discussion of recent developments in fermentation monitoring and control from a practical perspective. Critical reviews in Biotechnology. 13:117-149.

124. Rumelhart, D. E., Hinton, G. E., and Williams, R. J.(1986).Learning internal representations by error propagation. Parallel Distributed Processing. 1:318-362.

125. Sanchez, D. V(2003).Advanced support vector machines and kernel methods. Neurocomputing. 55:5-20.

126. Schnabel, R. B. and Eskow, E.(1990).A New Modified Cholesky Factorization. SIAM Journal of Scientific Statistical Computing. 11:1136-1158.

127. Schölkopf, B., Burges, J. C., and Smola, A.(1999).Advances in Kernel Methods--Support Vector Learning. MIT Press,

128. Schubert, J., Simutis, R., Havlik, D., and Lübbert, A.(1994).Hybrid Modelling of Yeast Production Processes – Combination of a priori knowledge on Different levels of Sophistication. Chemical engineering technology. 17:10-20.

129. Schwarz, G.(1978).Estimating the Dimension of a Model. Annals of Statistics. 6:461-464.

130. Seeger, M.(2003).Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations.

131. Sen, R. and Swaminathan, T.(2004).Response surface modeling and optimization to elucidate and analyze the effects of inoculum age and size on surfactin production. Biochemical Engineering Journal. 21:141-148.

132. Solomonoff, R. J.(1997).The Discovery of Algorithmic Probability. Journal of Computer and System Sciences. 55:73-88.

133. Stephanopoulos, G. N, Aristidou, A. A, and Nielsen, J.(1998).Metabolic Engineering Principles and Methodologies. Elsevier Science, 0-12-666260-6

134. Takors, R W, Wiechert, D, and Weuster-Botz, D(1998).Experimental design for the indentification of macrokinetic models and model discrimination. Biotechnology Bioengineering. 56:564-576.

135. Thompson, M. L. and Kramer, M. A.(1994).Modeling chemical processes using prior knowledge and neural networks. AIChE Journal. 40:1328-1340.

136. Tremblay, M., Perrier, M., Chavarie, C., and Archambault, J.(1993).Fed-batch culture of hybridoma cells: comparison of optimal control approach and closed loop stratergies. bioprocess engineering. 7:229-234.

137. Tsuchiya, H. M., Fredrickson, A. G., and Aris, R.(2005).Dynamics of Microbial Cell Populations. Advances in Chemical Engineering. 6:125-132.

138. Ushio, M(2003).A metabolic engineering approach to examine polyketide production by Saccharopolyspora erythraea.

139. V Vapnik(1995).The Nature of Statistical Learning Theory. Springer-Verlag, 0387987800

140. van Can, H., Braake, H., Hellinga, K, Luyben, K, and Heijnen, J.(1996).Strategy for Dynamic Process Modeling Based on Neural Networks in Macroscopic Balances. AIChE Journal. 42:3403-3418.

141. Vapnik, V.(1995).The Nature of Statistical Learning Theory. Springer-Verlag, 0387987800

142. Versyck, K. J., Claes, J. E., and Van Impe, J. F.(1-6-1998).Optimal experimental design for practical identification of unstructured growth models. Mathematics and Computers in Simulation. 46:621-629.

143. Warnes, M. R, Glassey, J., Montague, G. A., and Kara, B(1998).Application of Radial Basis Function and Feedforward Artificial Neural Networks to the Escherichia coli Fermentation Process. Neurocomputing. 20:67-82.

144. Weuster-Botz, D(2000). Experimental Design for Fermentation Media Development: Statistical Design or Global Random Search? Journal of Bioscience and Bioengineering. 90:473-483.

145. Whitehead, B. A. and Choate, T. D.(1994).Evolving space-filling curves to distribute radial basis functions over an input space. IEEE Transactions on Neural Networks. 5:15-23.

146. Willis, M. J., Montague, G. A., Di Massimo, C., Tham, M. T., and Morris, A. J.(1992).Artificial neural networks in process estimation and control*1. Automatica. 28:1181-1187.

147. Wilson, D. I., Agarwal, M., and Rippin, D. W. T.(15-10-1998).Experiences implementing the extended Kalman filter on an industrial batch reactor. Computers & Chemical Engineering. 22:1653-1672.

148. Xin, Yao(1999).Universal Approximation by Genetic Programming. 66-67.

149. Yang, Zhanwen, Liu, Mingzhu, and Song, Minghui(4-3-2005).Stability of Runge-Kutta methods in the numerical solution of equation u'(t)=au(t)+a0u([t])+a1u([t-1]). Applied Mathematics and Computation. 162:37-50.

150. Zadeh, L. A.(1965).Fuzzy Sets. Information and Control. 8:338-353.

151. Zorzetto, L. F. M. and Wilson, J. A.(1996).Monitoring bioprocesses using hybrid models and an extended Kalman filter. Computers & Chemical Engineering. 20:S689-S694.