MODELLING TECHNIQUES FOR BIOLOGICAL SYSTEMS

by

Alison Emslie Billing, BSc (Chem Eng) (Cape Town)

Thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in the Faculty of Engineering, University of Cape Town.

Department of Chemical Engineering
University of Cape Town

August 1987

DECLARATION BY CANDIDATE

I hereby declare that this thesis is my own work and has not been submitted for a degree at any other University.

A E Billing

August 1987

SYNOPSIS

The objective of this investigation has been to develop and evaluate techniques which are appropriate to the modelling and simulation of biological reaction system behaviour.

The model used as the basis for analysis of modelling and simulation techniques is a reduced version of the biological model proposed by the IAWPRC Task Group for mathematical modelling in wastewater treatment design. This limited model has the advantage of being easily manageable in terms of analysis and presentation of the simulation techniques whilst at the same time incorporating a range of features encountered with biological growth applications in general. Because a model may incorporate a number of different components and large number of biological conversion processes, a convenient method of presentation was found to be a matrix format. The matrix representation ensures clarity as to what compounds, processes and reaction terms are to be incorporated and allows easy comparison of different models. In addition, it facilitates transforming the model into a computer program.

Simulation of the system response first involves specifying the reactor configuration and flow patterns. With this information fixed, mass balances for each compound in each reactor can be completed. These mass balances constitute a set of simultaneous non-linear differential and algebraic equations which, when solved, characterise the system behaviour. Two situations were considered for the purposes of simulation:

(i) steady state conditions, where the system operates under conditions of constant influent flow and load;

(ii) dynamic conditions, where the influent to the system varies with time, usually in a cyclic pattern.

Modelling of steady state conditions

Under constant input conditions, the response of each compound in each reactor is described by a single concentration value which does not vary with time. For the steady state case, the derivative terms in the mass balance equations fall away and the problem is reduced to one of solving a set of algebraic equations which contain non-linear terms. Because of these terms, which are introduced into the equations through the biological kinetic expressions, iterative solution procedures must be employed.

Some insight into appropriate numerical solution procedures for this problem is gained by representing the equations in a matrix format. The matrix representation gives a concise summary of the steady state situation as well as providing a graphical illustration of the salient features of the system under consideration. It also indicates how the biological reaction processes and system configuration influence the choice of suitable numerical solution procedures.

A number of different approaches for computing the solution to the set of non-linear algebraic equations were evaluated. These were the five methods generally used in chemical engineering flowsheeting applications. The performance of each method was evaluated and compared in application to a range of specific steady state biological system problems. These problems incorporated the characteristics of the various types of flowsheet encountered in practice.

The most straightforward numerical technique examined was the direct linearisation approach. This involves linearising the set of non-linear equations at each iteration and solving the resultant system of linear equations by Gaussian elimination. Although this approach performed surprisingly efficiently for all the case studies, the extensive prior

mathematical manipulation required before the method could be implemented was seen as a major drawback.

The commonly encountered method of successive substitution was also evaluated. This method requires rearrangement of the non-linear equations into a form that allows fixed point iteration. This approach was refined still further by using an acceleration technique proposed by Wegstein in an attempt to improve the convergence properties. When applied to the case studies, the performance of both methods was found to be unsatisfactory. Although both techniques offered the advantages of being simple, slow convergence rates and potential instability problems in their implementation rendered them inappropriate for general use.

The most successful technique for the case studies was found to be Newton's method. This is an approach based on the idea of constructing a local linear approximation to the non-linear functions by using the Jacobian matrix of partial derivatives. In this case, a finite difference approximation to the Jacobian was successfully implemented, thus rendering the simulation program generally applicable. For all the case studies, Newton's method was always the fastest to converge and required significantly fewer iterations than any of the other methods to reach a solution.

A quasi-Newton method evaluated was Broyden's method, based on the idea of approximating the Jacobian in order to avoid the computational effort required to repeatedly evaluate it. Although the effort required to set up the Jacobian was half that of Newton's method, overall the approach did not improve on Newton's method. The increased number of iterations and the increased effort to solve the linear equations at each step outweighed the savings.

Modelling of the dynamic response

For the dynamic situation, the change in concentration of each compound in each reactor with time subject to variations in the input pattern is described by a set of coupled non-linear ordinary differential

equations. Solving the set of simultaneous equations constitutes an initial value problem. In this study, initial conditions in each reactor were taken as those produced by the solution to the steady state problem. Thereafter, the changes in concentration of each compound in each reactor are tracked using a stepping technique, which approximates the solution at a series of discrete points.

Although the differential equations describing the dynamic response are, in fact, coupled, it was found that the degree of coupling between certain compounds was not strong. This meant that a multirate integration technique could be applied where groups of compounds with differing dynamics are integrated separately.

Two groups of compounds in the biological system were identified; those with "slow" dynamics (generally the particulate compounds) and those with "fast" dynamics (generally the soluble compounds). The compounds exhibiting "slow" dynamics were integrated using long timesteps whilst the group of compounds exhibiting "fast" dynamics was integrated using short timesteps. This results in considerable savings in the computational effort compared to methods based on a single step length. With those methods, the steplength for all compounds would be constrained to the shorter step of the multirate groups.

The multirate technique also incorporated a variable steplength facility. This made further savings in the required computational energy for the integration method. The size of each step in the integration routine was based on an evaluation of the magnitude of the integration error generated at the previous step. The mechanism for adjusting step size was based on the approach of Dahlquist and Bjorck (1974). This method offers distinct advantages over the method proposed by Gear (1984).

The technique used to carry out the integration is a simple predictor-corrector approach corresponding to a second-order Runge-Kutta method. The explicit Euler formula is used to predict an initial estimate of the solution, which is then improved upon by the application of the implicit

trapezoidal rule as the corrector. This predictor-corrector pair was used to integrate both the "fast" and the "slow" groups of compounds, although different stepsizes were used to integrate these two groups. To account for coupling and simultaneous integration, straight line interpolation is used to obtain values for the "slow" components at intermediate points in the long integration steps.

The use of a multirate technique in combination with variable stepsize for the integration was found to be a most successful approach for biological system simulation.

ACKNOWLEDGEMENTS

It would be impossible to thank and acknowledge the many individuals who have contributed to the completion of this project.

I would like, however, to express my sincere appreciation to Peter Dold for his guidance and support throughout the project.

The CSIR is thanked for their financial support.

Many thanks to Cathy, Dave, Christopher, Murray, Glynnis and the many others for their assistance in times of dire need. Also to Caron Park, who helped me put it all together when it was falling apart.

Finally, I would especially like to acknowledge the support and understanding of my mother, without which this project would never have reached completion.

DEDICATION

During the years that I have spent at UCT, I have received an education that has hopefully equipped me to play a useful role in the society in which I live. I would like to take this opportunity to dedicate my skills to working for peace and change in this country and to express the hope that I can contribute to building a society based on the principles of peace and justice.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

% acc	-	percentage accuracy
b	-	specific decay rate of biomass (d^{-1})
B(x)	-	an approximation to the Jacobian matrix
C	-	concentration (ML^{-3})
c_n	-	constant specific to the order of a multistep method
c'	-	constant specific to the order of the predictor
c	-	constant specific to the order of the corrector
COD	-	chemical oxygen demand (g COD m^{-3})
f	-	endogenous residue fraction (g COD.$gCOD^{-1}$)
	-	also used to denote functions and their derivatives
g(x)	-	a function of x in the form x = g(x)
$\frac{\partial g}{\partial x}\Big\vert_{(X_0,Y_0)}$	-	the partial derivative of g with respect to x evaluated at the point (X_0, Y_0)
h	-	change in successive estimates of x for Newton's method
h	-	small timestep or stepsize used in integration method
H	-	large timestep or stepsize used in integration method
J(x)	-	The Jacobian matrix
K_h	-	maximum specific particulate COD utilisation rate under aerobic conditions ($gCOD.gCOD^{-1} \cdot d^{-1}$)
K_s	-	half saturation coefficient for readily biodegradable COD utilisation (g $COD.m^{-3}$)
K_x	-	half saturation coefficient for particulate COD utilisation under aerobic conditions ($gCOD.gCOD^{-1}$)
l_n	-	local error of a multistep method
m	-	number of applications of the corrector
m	-	slope for Wegstein's method
O_c	-	carbonaceous oxygen demand ($gO.m^{-3}.day^{-1}$)
O_t	-	total oxygen demand ($gO.m^{-3}.day^{-1}$)

OUR	-	oxygen utilisation rate ($gO.m^{-3}.day^{-1}$)
p	-	order of the integration method
q_w	-	volume of mixed liquor wasted per day ($l.d^{-1}$)
Q_i	-	influent wastewater flowrate (L^3T^{-1})
Q_r	-	settling tank underflow flowrate (L^3T^{-1})
r	-	number of small timesteps in large timestep H = rh
r_i	-	rate of reaction of compound i ($ML^{-3}T^{-1}$)
R_S	-	system sludge age (d)
s	-	change in successive estimates of x for Broyden's method
S_O	-	concentration of dissolved oxygen ($gO.m^{-3}$)
S_{Si}	-	concentration of soluble substrate in the influent ($gCOD.m^{-3}$)
S_{Sr}	-	concentration of soluble substrate in the underflow ($gCOD.m^{-3}$)
t	-	acceleration factor for Wegstein's method
t_o	-	beginning of time interval
V	-	Volume of reactor (l)
VSS	-	concentration of volatile suspended solids ($gVSS.m^{-3}$)
X_{Bi}	-	concentration of particulate biomass in the influent ($gCOD.m^{-3}$)
X_{Br}	-	concentration of particulate biomass in the underflow ($gCOD.m^{-3}$)
X_{Ei}	-	concentration of endogenous residue in the influent ($gCOD.m^{-3}$)
X_{Er}	-	concentration of endogenous residue in the underflow ($gCOD.m^{-3}$)
X_{Si}	-	concentration of particulate substrate in the influent ($gCOD.m^{-3}$)
X_{Sr}	-	concentration of particulate substrate in the underflow ($gCOD.m^{-3}$)
x	-	vector of x values
$x_{n(0)}$	-	initial estimate of the vector x_n
y	-	change in successive function values for Broyden's method
Y	-	true growth yield
y_n	-	value of y at time t_n [= $y(x_n)$]

y'_n	-	first derivative of y at time t_n [= $y'(x_n)$]
y''_n	-	second derivative of y at time t_n [= $y''(x_n)$]
y^P	-	predicted value of y
y^C	-	corrected value of y

GREEK CHARACTERS

ρ	-	process rate expression ($ML^{-3}T^{-1}$)
$\hat{\mu}$	-	maximum specific growth rate [(g COD cell growth).(g COD utilised)$^{-1}$ (day)$^{-1}$]
υ	-	stoichiometric coefficient
Δ	-	change
∂	-	partial derivative
ϵ	-	error tolerance
θ	-	safety factor for integration method

SUPERSCRIPTS

*	-	denotes solution
+	-	denotes scaled variable

SUBSCRIPTS

a	-	index denotes (a) recycle stream
b	-	index denotes (b) recycle stream
i	-	index denotes the i^{th} reactor in a series
j	-	index denotes the j^{th} compound in a series
k	-	index denotes the k^{th} reactor in a series
(p)	-	index denotes the p^{th} iteration
r	-	index denotes the underflow recycle stream from the gravity settler
(0)	-	index denotes the initial estimate

CHAPTER ONE

INTRODUCTION

The focus of this study is an evaluation of techniques for modelling and simulation of biological system behaviour. Consideration is given to both (1) the manner in which the mathematical model for a biological system is presented, and (2) comparison of various numerical simulation techniques.

The phenomenon of biological growth is harnessed in a wide variety of applications. These may range from, for example, a laboratory fermentation for the production of a pharmaceutical compound to the treatment of municipal wastewater in a full scale activated sludge process. The common feature in the various systems is biological growth, even though the scale of operation and the final objectives of the growth process may be very different. For example, in a fermentation process, the objective is to maximise certain soluble products of growth whereas in sewerage treatment an objective is to minimise the residual soluble material. Whatever the objective, it is useful to be able quantify system behaviour on the basis of a model of the process. Because biological growth is the central feature in all of these applications, it is likely that very similar considerations will be necessary in setting up and solving a mathematical model for any of the systems.

A simulation program which can predict the response of a biological system on the basis of a mathematical model is useful for a number of reasons. For example:

Model development: A mathematical model incorporates a number of kinetic and stoichiometric expressions which represent the biological interactions. These expressions are based on hypotheses which are proposed for the biological processes occuring within the system. In order to test these hypotheses, specific experiments are designed and data on the system response is accumulated. This data can then be

compared with the predictions obtained from the model. In turn, the biological model can be altered with the objective of improving the predictive capacity. A simulation program is thus an indispensable tool in facilitating the development and sophistication of a biological model.

System evaluation and optimisation: A simulation program can be a useful aid in analysing the operation of existing biological systems. If a system model can provide accurate predictions of response behaviour, then these predictions can be compared to observed responses. Any discrepancies can be useful in identifying problems in system operation. An accurate and representative computer model can also be used to optimise the performance of existing systems. Various operating strategies can be proposed and rapidly tested without having to resort to potentially difficult practical evaluation.

System design: A simulation program can be a useful tool for the design engineer. With the aid of an accurate and representative computer model, proposed system designs and configurations can be evaluated rapidly. In addition, a dynamic model can provide valuable design information which is usually only available through empirical estimates. For example, a parameter such as peak oxygen utilisation rate in an activated sludge system could be obtained directly from the simulation program run under time-varying input patterns. This means that the peak aeration capacity can be quantified accurately - traditional design methods rely on empirical estimates.

Control strategy development: A simulation program allows the rapid and efficient evaluation of control strategies in a manner similar to evaluation of system designs. Strategies can be tested and compared in an economical way that reduces the need for field evaluation.

Having identified some of the reasons why it is useful to model biological system behaviour, it is now possible to define certain of the requirements in a computer program for simulating system behaviour. Amongst these are the following:

- The program should be able to simulate the response behaviour in the types of system configuration encountered in practice. A typical biological reaction system configuration would consist of a series of interconnected reactors. In certain applications, the last reactor in the series would be followed by a sedimentation tank. Mixed liquor will usually flow sequentially by gravity from reactor to reactor, and internal recycles may convey liqour to upstream reactors. Influent to the system may be distributed to any of the different reactors. If a sedimentation tank is included, the underflow may be recycled to any of the reactors.

  In modelling the system described above, the general approach is to consider each reactor as a completely-mixed stirred tank (CSTR), with individual units being connected to the others by streams. These interconnected modules form the flowsheet which is used as the basis for the simulation program. [1]

- The simulation program should be capable of analysing both steady state and dynamic behaviour. In a steady state situation, the system operates under constant flow and load conditions. Under a dynamic regime, influent flow and/or concentration will vary with time. In certain situations such as a wastewater treatment plant, the inputs will vary with time in a cyclic pattern which is repeated closely from day to day. In this case, a useful facility of a computer simulation program is the ability to predict the steady state cyclic response under the expected cyclic input pattern.

- The computer program should be structured in such a way that refinements to the biological model can be made with a minimum of

---

[1] Plug flow reactors, such as oxidation-ditch type activated sludge systems, can also be modelled using this approach. The plug flow reactor is considered to be made up of a number of small CSTR's in series. This is a standard approach adopted in chemical engineering process simulation.

disruption of the program code. This is to facilitate efficient model evaluation and development.

- The numerical techniques utilised by the computer program should provide accurate solutions to both the steady state and the dynamic problem. The numerical methods should be efficient and economical in terms of computer time, as well as being stable and robust enough to handle a wide variety of system configurations and kinetic models.

Some mathematical model is required as a basis for the analysis of modelling and simulation techniques appropriate for biological systems. In this investigation, a model of limited scope has been selected. This has the advantage of being easily manageable in terms of both analysis and presentation of the techniques. Nevertheless, it has been attempted to select a model which is fully representative of the range of biological growth applications. The model which has been adopted is a simplified version of the activated sludge system model of the International Association for Water Pollution Research and Control (IAWPRC). Therefore, to a certain extent, the analysis is specific to the activated sludge system. Nevertheless, because this model incorporates features common to a range of biological processes, the results can be extended readily to other systems.

CHAPTER TWO

MATHEMATICAL DESCRIPTION OF BIOLOGICAL REACTION SYSTEMS

2.1 INTRODUCTION

A comprehensive mathematical model for the simulation of biological system behaviour must account for a large number of reactions between a large number of components (compounds). In this presentation, the reactions will be referred to as **processes**, where processes act on certain **compounds** in the **system**, and convert these to other compounds. The set of distinct biological processes and the manner in which these act on the group of compounds constitute the biological model. The model should quantify, for each process, both the kinetics (rate-concentration dependence) and the stoichiometry (effect on the masses of compounds involved) (Henze et al, 1987).

Once a model has been formulated for a biological system, simulation of the system response involves two principal steps. Firstly, the reactor configuration and the flow patterns need to be specified. Once this information is fixed, it is possible to complete mass balances over each reactor for each compound. Assuming that the system operates at constant temperature, this quantifies the behaviour of each compound in the system. The concentrations of these "compounds" constitute the state variables (dependent variables). The mass balances make up the state equations which relate the dependent variables to the independent variables such as reactor volume. The mass balances form a set of simultaneous non-linear equations which, when solved, characterise the system behaviour.[1] The simultaneous solution of these equations thus provides values of the state variables at points in space (different reactors) and time (where there is a time-varying input to the system). In this way, the change of state of the system is related to the

---

[1] The equations are usually non-linear because the kinetic expressions for biological systems generally are non-linear.

transport (input and output) and conversion (reaction) processes occurring within the system.

At this point it is worth noting certain characteristics of biological reaction systems which distinguish these from most other applications in the chemical process industry:

- A feature commonly encountered with biological systems is that the process occurs in a series of completely mixed stirred tank reactors.[2]

- An identical set of reactions often takes place in each reactor in the system. For example, in a series of aerobic activated sludge reactors, the behaviour in each reactor is governed by the same kinetic and stoichiometric expressions. The only difference between the reactors would be the values of the state variables, the reactor volumes and the flow terms.

- The response of biological systems is often governed largely by the effect of recycles.

These features of biological systems are not usually encountered in operations in the chemical process industry. Those systems are generally made up of a distinct set of unit operations. Therefore, each reactor unit is governed by a different collection of reaction equations i.e. a different model. Also, the magnitude of the recycles and feedbacks is generally small. These distinguishing features of biological systems, demand that specific consideration be given to their simulation.

---

[2] As noted earlier, certain reactor configurations such as oxidation ditch type systems for wastewater treatment may not appear to fit the description given here, as these are essentially plug flow reactors with recycle, and are not divided into distinct zones. However, these systems can be modelled as tanks-in-series systems by considering the plug flow zones to be made up of a number of small CSTR's in series. This is a standard approach adopted in chemical engineering process simulation.

2.2 MODEL REPRESENTATION

An important part of the simulation process is a clear and flexible representation of the model itself. Because a model may incorporate a number of different components and a large number of biological conversion processes, one convenient method of presentation is a matrix format.

The matrix method for model presentation described here is based on the approach to chemical kinetic modelling of Petersen (1965). In the context of biological systems, the method has been utilised by the IAWPRC Task Group on mathematical modelling in wastewater treatment design (Henze et al, 1987). The matrix representation ensures clarity as to what compounds, processes and reaction terms are to be incorporated and allows easy comparison of different models. In addition, the method facilitates transforming the model into a computer program.

2.2.1 Setting up the matrix

Table 2.1 presents, in matrix format, the essential components of a simple Monod-Herbert model for aerobic microbial growth on a soluble substrate, accompanied by organism death.

The first step in setting up the matrix is to identify the compounds of relevance in the model. The Monod-Herbert model quantifies the growth of the biomass component (X_B) at the expense of the soluble substrate component (S_B). By keeping track of X_B and S_B, it is possible to calculate the oxygen requirement, so oxygen (S_O) can be included as a third component. The compounds are presented as symbols across the top of the table, and are defined (with units) at the bottom of the corresponding matrix columns. The index "i" is assigned to the range of compounds. In this case, "i" ranges from 1 to 3 for the three compounds considered in this simple model. [3]

[3] The recommended symbol notation of the IAWPRC has been followed: namely, X for particulate matter and S for soluble materials. (Grau et al, 1982)

	Component i	1	2	3	Rate expressions
j	Process	X_B	S_B	S_O	
1	Growth	1	$\frac{-1}{Y}$	$-\frac{(1-Y)}{Y}$	$\hat{\mu} X_B \frac{S_B}{(K_B+S_B)}$
2	Decay	-1		1	$b X_B$
Observed conversion rates ($ML^{-3}T^{-1}$)		$r_i = \Sigma \nu_{ij} \rho_j$			

Kinetic parameters:
Maximum specific growth rate: $\hat{\mu}$

Half saturation constants: K_B, K_O
Specific decay rate: b

Stoichiometric parameters:
True growth yield: Y

BIOMASS M (COD) L^{-3}	SUBSTRATE M (COD) L^{-3}	OXYGEN (NEGATIVE COD) M (-COD) L^{-3}

Table 2.1 The Monod-Herbert model showing process kinetics and stoichiometry for heterotrophic bacterial growth in an aerobic environment

The second step in developing the matrix is to identify the biological processes occurring in the system. These are conversions or transformations which affect the compounds considered in the model. Only two processes take place in this simple model - aerobic growth of organisms at the expense of soluble substrate, and organism decay. These are itemised one above the other at the left of the matrix. The index "j" is assigned to the range of processes; in this case "j" can only assume a value of 1 or 2.

The kinetic expressions (rate equations) for each process are recorded down the right hand side of the matrix in the appropriate row. These are given the symbol ρ_j with j denoting the index of the biological process. The kinetic parameters incorporated in the rate expressions are defined at the lower right corner of the matrix.

The elements within the matrix comprise the stoichiometric coefficients, $\nu_{i,j}$, which define the mass action relationship between the components in the individual processes. For example, aerobic growth of heterotrophs (+1) occurs at the expense of soluble substrate (-1/Y); oxygen is utilised in the metabolic process (-(1-Y)/Y). The stoichiometric parameters are defined at the lower left of the table.

The stoichiometric coefficients $\nu_{i,j}$ are greatly simplified by working in consistent units; in this case, all concentrations are expressed as COD equivalents. Provided consistent units have been used, continuity may be checked from the stoichiometric parameters by moving across any row of the matrix. With consistent units, the sum of the stoichiometric coefficients must be zero (noting that oxygen uptake is equivalent to negative COD).

The sign convention used in the matrix is "negative for consumption" and "positive for production". Cognisance must be taken of the units used in the rate equation. For example, the rate equation for aerobic growth of biomass, ρ_1, is written as a biomass growth rate (not as a substrate utilisation rate) and has units of (mg cell COD growth)(mg substrate COD

utilised)$^{-1}$(day)$^{-1}$. The stoichiometric values are thus normalised with respect to the biomass concentration i.e. for growth, the stoichiometric coefficients for X_B and S_B are 1 and -1/Y respectively, and not Y and -1.

2.2.2 Use in mass balances

Within a system boundary, the concentration of a single compound may be affected by a number of different processes. An important benefit of the matrix representation is that it allows rapid and easy recognition of the fate of each component, which aids in the preparation of mass balance equations.

The fundamental equation for a mass balance within any defined system boundary is:

$$\begin{bmatrix}\text{Rate of Accumulation}\end{bmatrix} = \begin{bmatrix}\text{Rate of Input}\end{bmatrix} - \begin{bmatrix}\text{Rate of Output}\end{bmatrix} + \begin{bmatrix}\text{Rate of Production by Reaction}\end{bmatrix} \qquad (2.1)$$

The input and output terms are transport terms and depend upon the physical characteristics of the system being modelled. The incorporation of these is discussed later. The system reaction term (usually denoted by r_i for compound i) must often account for the combined effect of a number of processes. In the matrix format, this information is obtained by summing the products of the stoichiometric coefficients, ν_{ij}, times the process rate expression, ρ_j, for the component i being considered in the mass balance i.e. moving down the column for the specific component i and accumulating the product of ν_{ij} and ρ_j:

$$r_i = \sum_j \nu_{ij}\ \rho_j \qquad (2.2)$$

For example, from Table 2.1, the rate of reaction for the compound biomass (X_B) at a point in the system would be:

$$r_{XB} = \frac{\hat{\mu}\, S_B}{(K_B + S_B)} \cdot X_B - b\, X_B \qquad (2.3)$$

Similarly for the component soluble substrate (S_B):

$$r_{SB} = \frac{-1}{Y} \cdot \frac{\hat{\mu}\, S_B}{(K_B + S_B)} \cdot X_B \qquad (2.4)$$

and for dissolved oxygen (S_O):

$$r_{SO} = -\frac{(1-Y)}{Y} \cdot \frac{\hat{\mu}\, S_B}{(K_B + S_B)} \cdot X_B - b\, X_B \qquad (2.5)$$

To create the mass balance for any component within a given system boundary (e.g. a completely mixed reactor) the conversion rate, r_i, would be combined with the appropriate advective terms (input and output flow) for the particular system: this is not shown here as the system is not yet defined.[4]

2.2.3 Switching functions

At this point, it is worth introducing an aspect of the kinetic expressions which is often useful - namely "switching functions". Consider the aerobic growth of biomass. In Table 2.1, the Monod growth rate equation has been utilised:

$$\rho_1 = \frac{\hat{\mu}\, S_B}{(K_B + S_B)} \cdot X_B \qquad (2.6)$$

In an environment where the dissolved oxygen concentration (S_O) is zero (or perhaps close to zero), the rate of this aerobic process should also decrease to zero. Mathematically, this can be achieved by multiplying the Monod rate expression by a "switching" factor which is zero when S_O

---

[4] The system reaction rate or conversion rate, r_i, may be of interest on its own. For example, Eq (2.5) defines the "rate of production" of S_O: therefore $-r_{SO}$ defines the oxygen utilisation rate at a point within the system. This parameter is often of interest in aerobic systems.

is zero, and unity when the environment is aerobic. In this case, it is convenient to write the switching function in the form:

$$\frac{S_O}{(K_O + S_O)} \qquad (2.7)$$

where K_O = switching constant of small magnitude
(say 0.1 mgO/l)

The process rate equation then becomes:

$$\rho_1 = \frac{\hat{\mu}\, S_S}{(K_S + S_S)} \cdot \frac{S_O}{(K_O + S_O)} X_B \qquad (2.8)$$

With this "switching function" operating on the growth rate equation, when S_O is zero the value of the function is zero, and the process rate, ρ_1, will be zero. However, if S_O is say 1 mgO/l then the value of the switching function is close to unity and the process rate will then be that given by the Monod equation. In this way, the process of aerobic growth is switched "on" or "off" automatically by the model depending on the dissolved oxygen concentration. The selection of a small value for K_O means that the value of the switching function decreases from near-unity to zero only at very low S_O values i.e. when the D.O. value decreases below, say 0.2 mgO/l. However, the function is mathematically continuous, which helps to eliminate problems of numerical instability in simulating system behaviour; such problems can arise if the rate is switched "on" and "off" discontinuously.

In certain situations, the switching "off" of one process may be linked to the switching "on" of another. If, for example, the oxygen input to a nitrifying activated sludge system were terminated periodically, there would be a switch from aerobic to anoxic growth. The latter process is governed by kinetic and stoichiometric expressions which differ from those for the aerobic growth process. To account for this phenomenon in a single model, the rate equations for aerobic and anoxic growth can be multiplied by the appropriate switching functions as follows:

$$\text{Observed } \rho_{aerobic} = \rho_{aerobic} \cdot \frac{S_o}{(K_O + S_O)} \qquad (2.9)$$

$$\text{Observed } \rho_{anoxic} = \rho_{anoxic} \cdot \left[1 - \frac{S_O}{(K_O + S_O)} \right]$$

$$= \rho_{anoxic} \cdot \frac{K_O}{(K_O + S_O)} \qquad (2.10)$$

In this instance, it is apparent that the selection of K_O will influence the point at which there is a switch from aerobic to anoxic growth, and vice versa. That is, K_O now influences the model predictions and is not only serving a mathematical objective. Therefore, whenever switching functions are utilised, care should be taken in the selection of the magnitude of the switching constant (K_O here) to ensure that the model predictions are not incorrectly biased.

The consequence of using switching functions to switch between processes within a model should be highlighted. The example of anoxic and aerobic growth illustrates how switching functions enable incorporation of qualitative changes in system behaviour within a single model. Without switching functions, different models would be required to simulate the behaviour either in an aerobic or an anoxic environment.

2.3 BIOLOGICAL MODEL USED IN THIS STUDY

A biological model of limited complexity has been selected as the "demonstration" model in this study. That is, only a limited number of compounds and processes have been incorporated. The objective of limiting model size has been to enable rapid evaluation of numerical modelling techniques. Despite its limited size, however, the model nevertheless incorporates a range of characteristics encountered in biological systems.

Table 2.2 presents the limited model in matrix format. This model is a reduced version of that proposed by the IAWPRC Task Group for mathe-

	Component i	1	2	3	4	5	Rate expressions
j	Process	X_B	X_E	X_S	S_S	S_O	
1	Growth	1			$\frac{-1}{Y}$	$-\frac{(1-Y)}{Y}$	$\hat{\mu}\, X_B \frac{S_S}{(K_S+S_S)} \cdot \frac{S_O}{(K_O+S_O)}$
2	Decay	-1	f	(1-f)			$b\, X_B$
3	Solubilisation			-1	1		$\frac{K_H\ (X_S/X_B)}{(K_X + (X_S/X_B))} X_B$
Stoichiometric parameters: True growth yield: Y Endogenous residue fraction : f		PARTICULATE BIOMASS M (COD) L^{-3}	ENDOGENOUS RESIDUE M (COD) L^{-3}	PARTICULATE SUBSTRATE M (COD) L^{-3}	SOLUBLE SUBSTRATE M (COD) L^{-3}	OXYGEN (NEGATIVE COD) M (-COD) L^{-3}	Kinetic parameters: Maximum specific growth rate: $\hat{\mu}$ Maximum solubilisation rate: K_H Half saturation constants: K_S, K_O, K_X Specific decay rate: b

Table 2.2 The reduced IAWPRC model for utilisation of carbonaceous material in an aerobic activated sludge system

matical modelling in wastewater treatment design (Dold and Marais, 1985) (Henze et al, 1987). The model incorporates only those features which relate to the utilisation of carbonaceous material in an aerobic activated sludge system.

Five compounds are identified in the demonstration model. These are:

- heterotrophic organism mass (X_B)
- endogenous residue (X_E)
- particulate biodegradable substrate (X_S)
- soluble biodegradable substrate (S_S)
- dissolved oxygen (S_O)

Three processes operate on the compounds in a manner defined by the stoichiometry and the process rate equations:

Aerobic growth of heterotrophs: Soluble substrate (S_S) is utilised for growth by the heterotrophic organisms (X_B). There is an associated utilisation of oxygen (S_O). The process is modelled by the Monod expression together with a switching function which reduces the rate to zero in the absence of oxygen.

Death of heterotrophs: Organism decay is modelled according to the "death-regeneration" hypothesis. The heterotrophic organism mass dies at a certain rate: a portion of the material from death is non-degradable (f) and adds to the endogenous residue (X_E) while the remainder (1-f) adds to the pool of biodegradable particulate COD (X_S).

Hydrolysis of particulate COD: Biodegradable particulate COD in the influent is assumed to be enmeshed in the sludge mass within the system. The enmeshed material is broken down extracellularly, with the products of breakdown adding to the pool of readily biodegradable substrate (S_S) available to the organisms for synthesis purposes. This "hydrolysis/solubilisation" process is modelled on the basis of Levenspiel's surface reaction kinetics (Levenspiel, 1972).

A number of features incorporated in this model, and which may be encountered with other biological systems, should be noted. These are:

Dual substrate: The model distinguishes between soluble and particulate biodegradable influent material, and the manner in which these are removed in the system.

Non-linear expressions: The non-linear nature of certain of the process rate equations introduces non-linear terms into the mass balance equations. This aspect influences the numerical techniques for solution of the simulation problem.

Single and series reactions: Utilisation of soluble substrate directly by the organism is modelled as a single reaction. However, utilisation of particulate material occurs in two steps: hydrolysis to soluble substrate followed by utilisation of the soluble substrate. This sequence constitutes a series reaction.

Bulk versus surface concentration terms: Generally, process rate expressions are formulated in terms of the bulk concentration of certain species in the system (i.e. the mass per unit system volume). For example, the concentration of soluble substrate (S_S) as used in the Monod growth rate expression is given by the mass of S_S in the system divided by the volume of the system. However, in certain cases, the basis for quoting concentration is some parameter other than the system volume. For example, hydrolysis of particulate substrate is modelled as being dependent on the concentration of particulate material adsorbed onto the organism mass i.e. the surface concentration. The ratio of two bulk concentrations (X_S/X_S) is used to approximate the surface concentration, and this term appears in the rate expression.

2.4 SETTING UP THE MASS BALANCE EQUATIONS

In a system consisting of a series of completely mixed reactors, the set of equations defining the state of the system is obtained by performing a separate mass balance over each reactor for each compound. Where a

solids/liquid separator, such as a gravity settling tank, is included in the configuration, an additional set of mass balance equations is required.

2.4.1 The reactor

Consider a single component in the i^{th} reactor in a series of n completely mixed reactors (Fig 2.1):

The inputs to the reactor could comprise some or all of the following:

(i) an influent feed stream at a flow rate Q_{feed} and a concentration C_{feed};
(ii) flow from the previous reactor [$(i-1)^{th}$] in the series, at a flow rate Q_{i-1} and a concentration C_{i-1};
(iii) a mixed liquor recycle (a) from the k^{th} reactor in the series, at a flow rate Q_a and a concentration C_k;
(iv) underflow from the settling tank at a flow rate Q_r and a concentration C_r.

Output streams from the i^{th} reactor could comprise some or all of the following:

(i) flow from this reactor to the next reactor [$(i+1)^{th}$] in the series at a flow rate Q_i and a concentration C_i;
(ii) a mixed liquor recycle (b) out of this reactor at a flow rate Q_b and a concentration C_i;
(iii) a sludge wastage stream may be withdrawn from the reactor at a rate q_w and a concentration C_i [5] ;

The reaction terms are obtained as described previously, by summing the products of the stoichiometric coefficients and the process rate expressions for the particular component being considered. These

[5] Biological sludge is withdrawn to prevent a build-up of solids in systems incorporating a solids/liquid separator. In this presentation, waste liquor will be withdrawn only from the last reactor in the series.

conversion terms are combined with the flow terms to create the mass balance equations.

Substituting in Eq (2.1), the mass balance for a single compound in the i^{th} reactor in a series is:

$$V_i \frac{dC_i}{dt} = Q_{feed}C_{feed} + Q_{i-1}C_{i-1} + Q_a C_x + Q_r C_r - Q_i C_i - Q_b C_i - q_w C_i + rV_i \qquad (2.11)$$

where V_i = volume of the i^{th} reactor (L^3)

C = concentration (ML^{-3})

Q = flow rate (L^3T^{-1})

q_w = wastage rate (L^3T^{-1})

r = rate of reaction or conversion rate of the compound (positive for production) ($ML^{-3}T^{-1}$)

$= \sum_j \nu \rho$

2.4.2. The solids/liquid separator

In certain circumstances, the output from the last reactor in the biological system passes to a solids/liquid separation device (often a gravity settler). This is usually with the intention of being able to maintain an organism retention time in excess of the hydraulic retention time and for maintaining a solids-free effluent. In this presentation, it has been assumed that the process which occurs in the settling tank is merely one of physical concentration i.e. no reaction takes place. In this way, the settling tank is treated as a separation point with no hold-up. Also, the settling tank is considered to operate at 100 percent efficiency. This means that the overflow from the settling tank comprises only soluble material and all particulate compounds entering the vessel are recycled back to the chain of reactors. Mass balances over the settler must therefore distinguish between particulate and soluble compounds.

Figure 2.2 illustrates the flow terms associated with a settling tank situated at the end of a series of n reactors. These are:

(i) flow from the last (n^{th}) reactor at a flow rate ($Q_{feed} + Q_r - q_w$) and a concentration C_n;

(ii) overflow from the settling tank at a rate of ($Q_{feed} - q_w$) and a concentration of C_n for soluble material and C = 0 for particulate material;

(iii) underflow from the settling tank, at a flowrate of Q_r and a concentration C_r.

Mass balances for the particulate and soluble compounds are as follows:

Particulate:

$$(Q_{feed} + Q_r - q_w)\, C_n = Q_r C_r \qquad (2.12)$$

Soluble:

$$(Q_{feed} + Q_r - q_w)\, C_n = Q_r C_r + (Q_{feed} - q_w)\, C_n$$

With $C_n = C_r$ for soluble compounds, this yields the trivial mass balance:

$$C_n - C_r = 0 \qquad (2.13)$$

2.4.3 Dissolved oxygen mass balance

Although dissolved oxygen (S_O) is included in the matrix, a mass balance[1] for S_O will not usually be required. This is because the oxygen input to a reactor is generally regulated externally in order to maintain the dissolved oxygen concentration at some constant value. The reason for including S_O is that it allows computation of the oxygen utilisaiton rate ($-r_{SO}$), an important parameter in modelling aerobic behaviour.

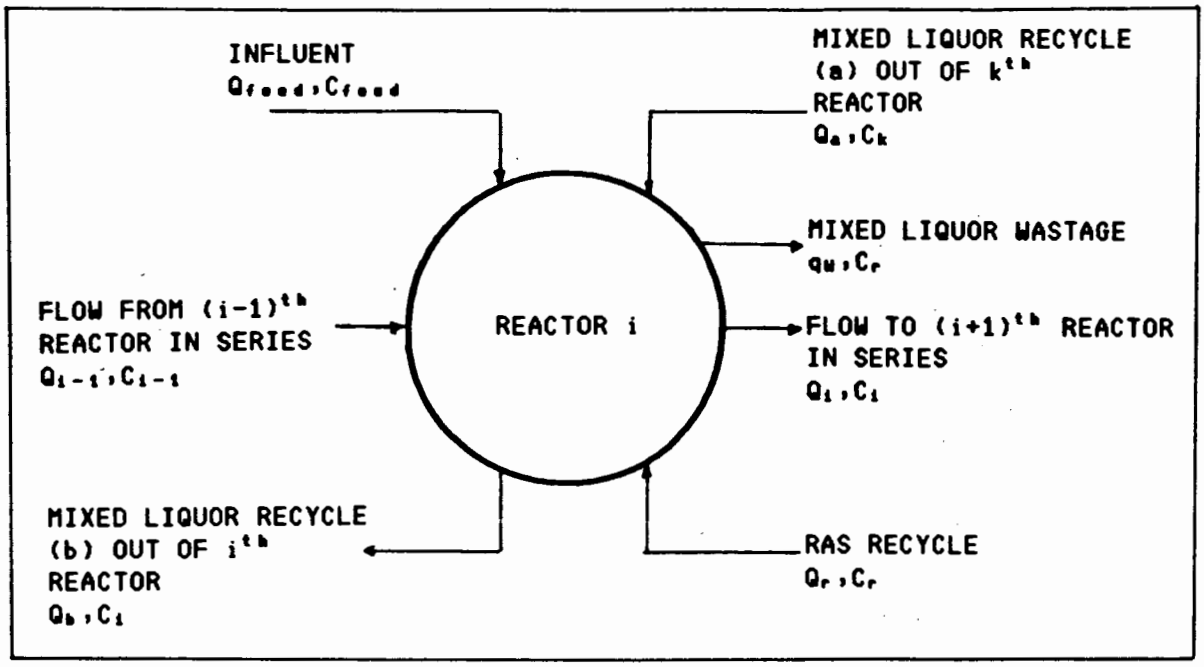


Figure 2.1 Schematic representation of the i^{th} reactor in a series of n completely mixed reactors

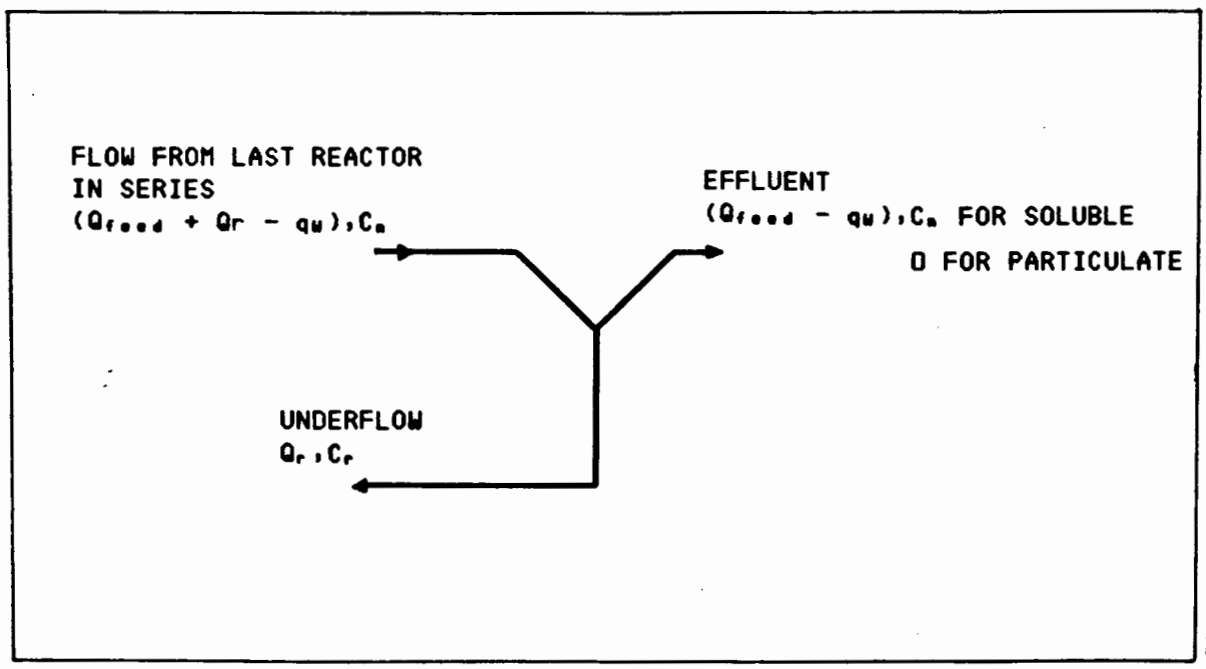


Figure 2.2 Schematic representation of a solids/liquid separator at the end of a series of reactors

2.5. A CASE STUDY

Consider the system of Fig 2.3 comprising a single aerobic reactor and a settling tank. Underflow from the settling tank is returned to the reactor. The system is described by eight mass balance equations, one for each of the compounds, X_B, X_E, X_S, and S_S in the reactor and in the settling tank underflow, respectively. The eight simultaneous equations comprise a set of four non-linear ordinary differential equations for the reactor and four algebraic equations for the solids/liquid separator.

Reactor:

$$V \frac{dX_B}{dt} = Q_i X_{B,i} + Q_r X_{B,r} - (Q_i+Q_r)X_B - b X_B V + \frac{\hat{\mu} S_S X_B}{(K_S+S_S)} V \qquad (2.14)$$

$$V \frac{dX_E}{dt} = Q_i X_{E,i} + Q_r X_{E,r} - (Q_i+Q_r)X_E + f b X_B V \qquad (2.15)$$

$$V \frac{dX_S}{dt} = Q_i X_{S,i} + Q_r X_{S,r} - (Q_i+Q_r)X_S + (1-f) b X_B V - \frac{K_H X_S}{(K_X + X_S/X_B)} V \qquad (2.16)$$

$$V \frac{dS_S}{dt} = Q_i S_{S,i} + Q_r S_{S,r} - (Q_i+Q_r)S_S - \frac{\hat{\mu}}{Y} \cdot \frac{X_B S_S}{(K_S+S_S)} V + \frac{K_H X_S}{(K_X + X_S/X_B)} V \qquad (2.17)$$

Solids/liquid separator:

$$(Q_i + Q_r - q_W) X_B = Q_r X_{B,r} \qquad (2.18)$$

$$(Q_i + Q_r - q_W) X_E = Q_r X_{E,r} \qquad (2.19)$$

$$(Q_i + Q_r - q_W) X_S = Q_r X_{S,r} \qquad (2.20)$$

$$S_S = S_{S,r} \qquad (2.21)$$

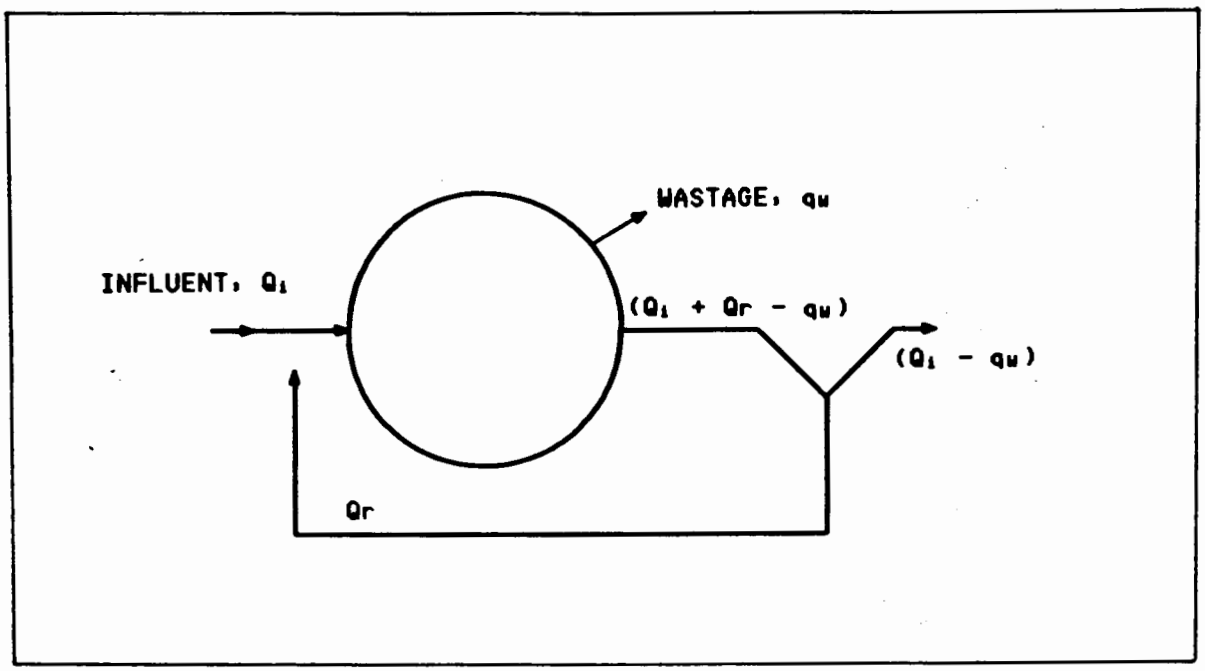


Figure 2.3 A case study: a single aerobic reactor with settling tank

2.6 CLOSURE

For the steady state situation, where the system operates under conditions of constant influent flow and load, the derivative terms in the reactor mass balance equations fall away. This reduces the single-reactor-plus-settler example problem to a set of eight simultaneous non-linear algebraic equations. In the dynamic situation, the problem remains one of solving the system of four non-linear ordinary differential equations and four algebraic equations. Solution procedures for solving the sets of equations resulting from the steady state and dynamic situations necessitate specific considerations in each case.

The steady state problem involves finding a single value for the concentration of each compound in each reactor and in the underflow recycle stream which satisfies the set of algebraic equations. Because the biological reactions introduce non-linear terms into the equations, the solution cannot be found directly and iterative techniques must be employed. These techniques range in complexity from simple successive substitution (with or without acceleration) to the various Newton-type methods. The success and efficiency of the different techniques is determined principally by the degree of non-linearity in the equations.

Under dynamic conditions, a set of coupled ordinary differential and algebraic equations describe the change in concentration of each compound in each reactor with time subject to variations in the input pattern. Because the biological system incorporates reactions involving both soluble and particulate compounds at a range of concentrations, the system will exhibit dynamics varying from fast to slow for different compounds. Therefore, utilisation of an integration technique that exploits the differing dynamics exhibited by the compounds in a biological system is indicated.

In both the steady state and dynamic situations, the objective has been to identify numerical techniques which take advantage of the particular characteristics of the equations describing the system. Through exploring the nature of their non-linearity, and exploiting the specific

dynamics of the biological reaction behaviour, it has been possible to identify techniques appropriate for either the steady or the dynamic situation.

CHAPTER THREE

MODELLING OF THE STEADY STATE CASE

3.1 INTRODUCTION

"Steady state" conditions are defined as those where the biological reaction system operates under conditions of constant input flow rate and load. The problem in modelling is one of predicting the state of the system for different system configurations and operating conditions. That is, under these constant input conditions, the response of each compound in each reactor is described by a single concentration value which does not vary with time. It is these concentration values that provide the solution to what has been termed the "steady state" problem.

A system which operates under steady state conditions as described above can be characterised by a set of simultaneous mass balance equations which include non-linear terms. Any time- dependent or derivative terms will be zero, and the set of equations will therefore be algebraic. The solution to the system of non-linear equations cannot be expressed in closed form, so "exact" or direct methods cannot be applied. Instead, iterative procedures must be employed. These require an initial estimate of the solution which is updated via a linear approximation of the relevant mass balance functions. The updating procedure is repeated until convergence is achieved. The main concern is the selection of a solution technique that will guarantee convergence. Additional considerations in the choice of a suitable numerical method would be its computational efficiency, robustness and stability.

3.2 A CASE STUDY: CONTINUED

Consider the single aerobic reactor plus settling tank problem of Chapter 2, Section 2.5 (Fig 2.3). Under steady state conditions, any derivative terms in Eqs (2.14) to (2.21) fall away, and the resultant eight steady state mass balances become:

Reactor:

$$Q_r X_{B,r} - (Q_i + Q_r) X_B - b\, X_B\, V + \frac{\hat{\mu}\, S_S\, X_B}{(K_S + S_S)} V = - Q_i\, X_{B,i} \qquad (3.1)$$

$$Q_r X_{E,r} - (Q_i + Q_r) X_E + f\, b\, X_B\, V = - Q_i\, X_{E,i} \qquad (3.2)$$

$$Q_r X_{S,r} - (Q_i + Q_r) X_S + (1-f)\, b\, X_B\, V - \frac{K_H\, X_S}{(K_X + X_S / X_B)} V = - Q_i\, X_{S,i} \qquad (3.3)$$

$$Q_r S_{S,r} - (Q_i + Q_r) S_S - \frac{\hat{\mu}}{Y} \cdot \frac{S_S\, X_B}{(K_S + S_S)} V + \frac{K_H\, X_S}{(K_X + X_S / X_B)} V = - Q_i\, S_{S,i} \qquad (3.4)$$

Solids/liquid separator:

$$(Q_i + Q_r - q_W)\, X_B - Q_r\, X_{B,r} = 0 \qquad (3.5)$$

$$(Q_i + Q_r - q_W)\, X_E - Q_r\, X_{E,r} = 0 \qquad (3.6)$$

$$(Q_i + Q_r - q_W)\, X_S - Q_r\, X_{S,r} = 0 \qquad (3.7)$$

$$S_S - S_{S,r} = 0 \qquad (3.8)$$

These equations may be written in the form $f(x) = 0$ where x is the vector of state variables:

$$x = \begin{bmatrix} X_B \\ X_E \\ \vdots \\ X_{S,r} \\ S_{S,r} \end{bmatrix}$$

Some insight into appropriate numerical solution procedures for the steady state problem may be gained by representing the equations in a matrix format..

3.3 THE STEADY STATE MATRIX

The matrix representation is used here because it gives a concise summary of the steady state problem. It shows, amongst others, features such as feed distribution, the flow links between reactors and the conversion processes, in a "graphical" manner.

Consider how the eight simultaneous steady state mass balance equations [Eqs (3.1) to (3.8)] of the case study are transformed into the matrix format in Fig 3.1. The equations are expressed in the form:

$$A\,X = B \qquad (3.9)$$

The X Vector: Equations (3.1) to (3.8) are mass balances for the eight state variables X_B, X_E,....,$X_{S,r}$ and $S_{S,r}$. These state variables form the X vector, which is the solution to the steady state problem.

The B Vector: This is the "feed vector". It contains the elements of the right hand sides of Eqs (3.1) to (3.8). Each term is the influent mass input rate of the corresponding compound into the particular zone (negative value). In this case, the first four values are the influent mass input rates of X_B, X_E, X_S and S_S into the reactor. For example, $-Q_i X_{B,i}$ is the input rate of X_B into the reactor. The last four values are the influent inputs into the settler (zero here).

The A Matrix: The A matrix contains the reaction and flow terms which characterise the particular activated sludge system configuration. It is of interest to note how the non-linear terms are handled. Consider how Eq (3.1) is inserted in the top row of the matrix. The linear terms can only be placed in one location. These are $-(Q_i+Q_r)$, Q_r and $-bV$. However, the non-linear term, $(\hat{\mu}\ S_S\ X_B\ V/\ (K_S+S_S))$ can be handled in two ways:

- $(\hat{\mu}\ S_S\ V/\ (K_S+S_S))$ in the X_B location

or

- $(\hat{\mu}\ X_B\ V/\ (K_S+S_S))$ in the S_S location

In this case, the second option has been used.

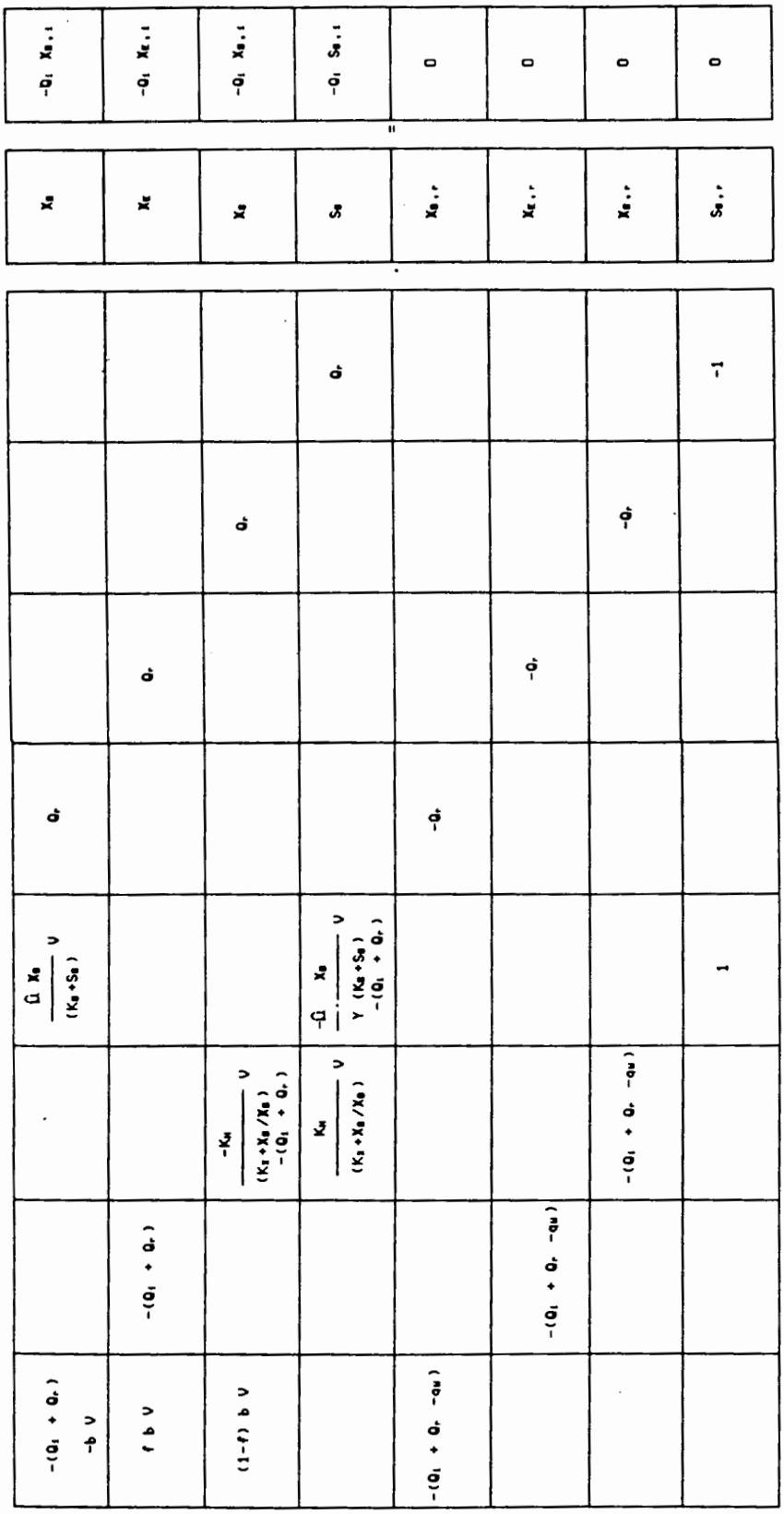


Figure 3.1 A matrix representation of the mass balance equations for a single reactor and settling tank system.

The A matrix is always square and has dimension (number of compounds * (number of reactors + 1)). In this case, the system contains four different compounds and one reactor (plus settler). Hence, the size of the A matrix will be eight by eight. Each four by four "block" of the matrix contains specific information about the nature of the system being analysed.

- The top left hand "block" contains terms for the reaction processes occurring in the reactor. Also, on the diagonal, flow-related terms appear. These represent the total flow out of the reactor [equal to the sum of the flows into the reactor i.e. $-(Q_i + Q_r)$].

- The bottom right hand block represents the settler. Because no reaction takes place in the settler, only flow-related terms appear. These represent flow out of the settler which is recycled within the system. ($-Q_r$ for particulate and -1 for soluble compounds).

- The diagonal vector, Q_r, in the top right hand block of the matrix represents the underflow recycle from the settler to the first reactor. Flows directed upstream or "backwards" such as recycles will always lie above the diagonal of the matrix.

- The diagonal vector ($Q_i + Q_r$), (with +1 for the soluble compound) in the lower left hand block of the matrix represents the flow from the reactor into the settling tank. Downstream or "forward" flows will always lie below the diagonal of the matrix.

Let us now extend the example to a system consisting of n reactors in series, followed by a settling tank. The system can be represented in general matrix format as shown in Fig 3.2.

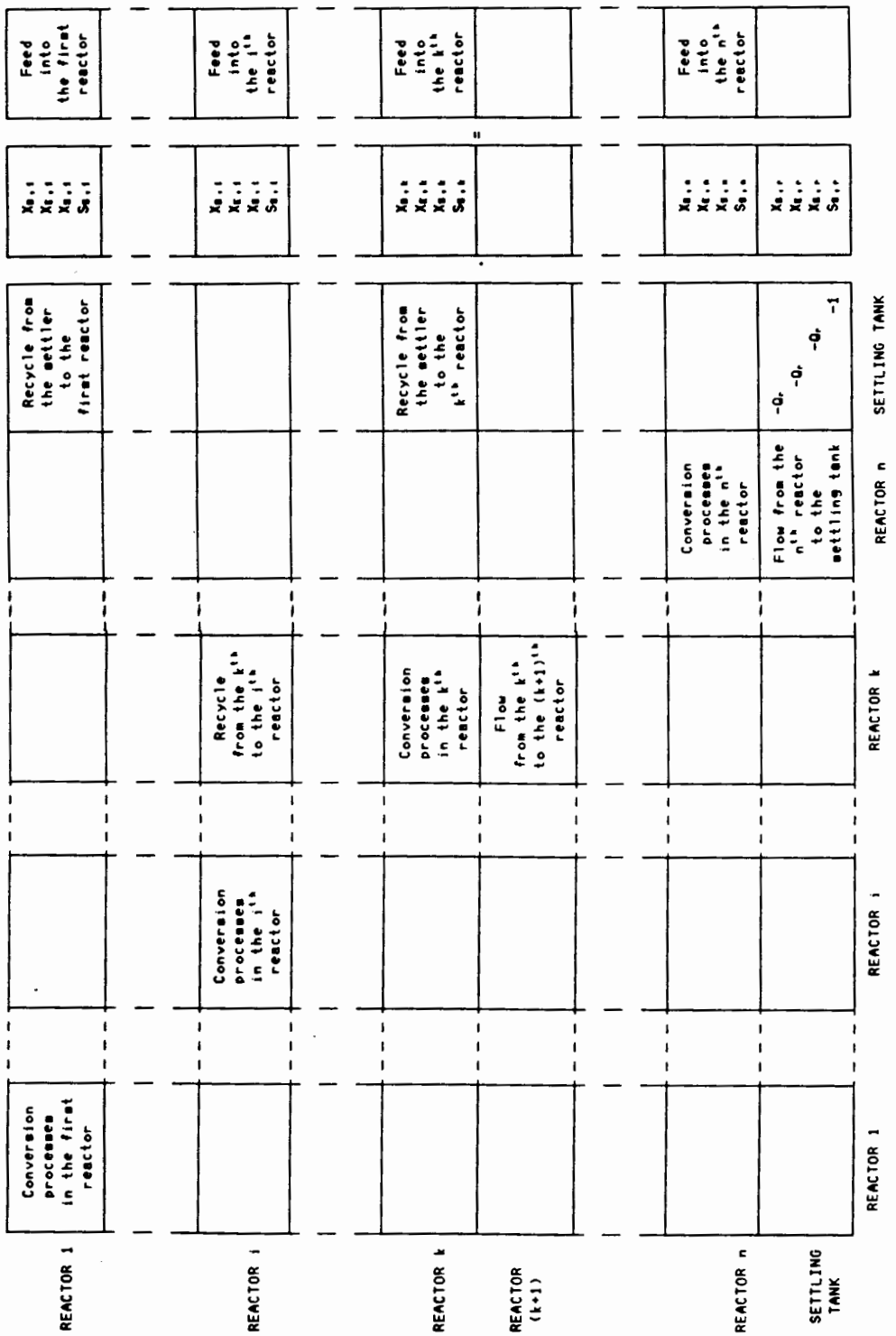


Figure 3.2 The steady state matrix representation of an n reactor system. Each block in the matrix corresponds to a sub-matrix of dimension (number of compounds).

The X vector: The X vector contains the terms $X_{B,1}$, $X_{E,1}$,....,$X_{S,r}$, $S_{S,r}$. These are the concentrations of the compounds X_B, X_E, X_S and S_S in reactors 1,2,...,n and in the underflow from the settler, r. These state variables form the solution to the steady state problem.

The B vector: The B vector contains the feed terms which are the influent input rates of the corresponding compounds into each reactor. In situations where all the feed enters the first reactor, only the first four terms will appear in the vector; all other terms will be zero. If the feed to the system is split, with a portion of the feed entering the k^{th} reactor, then the corresponding locations in the B vector will accordingly be filled with non-zero terms.

The A matrix: This is a square matrix of dimension ((n+1) * no. of compounds). The large matrix can be subdivided into (n+1) by (n+1) sub-matrices. Each sub-matrix is square with dimension equal to the number of compounds.

Consider the k^{th} reactor in the series. The terms representing the conversion processes occuring in the k^{th} reactor will be situated in the k^{th} reactor "block" on the diagonal of the A matrix as indicated in Fig 3.2. In addition, the diagonal within the k^{th} reactor block will contain terms representing flow out of the k^{th} reactor. Flow from the k^{th} reactor to the $(k+1)^{th}$ reactor in the series will be represented by a diagonal vector containing the relevant flow terms in a block situated directly "below" the k^{th} block on the diagonal. That is, the vertical location of the block will be fixed opposite the column representing the k^{th} reactor. The horizontal location of the block will be fixed by the column representing the $(k+1)^{th}$ reactor.

Recycle flows from one reactor to another in the series are handled in a similar fashion. A recycle from the k^{th} to the i^{th} reactor in the series will be represented by a diagonal vector containing the relevant flow terms in a block situated above the diagonal of the A matrix. The vertical location of the block will be fixed by the column representing the k^{th} reactor and the horizontal location of the block will be fixed

by the column representing the i^{th} reactor. In general, the vertical position of the sub-matrix represents flow "out of" that reactor. The horizontal position of the sub-matrix represents flows "into" that reactor.

3.4 SOLUTION TO THE STEADY STATE PROBLEM

The topography of the steady state matrix, besides providing a graphical illustration of the salient features of the system under consideration, also has specific implications for the nature of a suitable solution procedure. The case study has illustrated that the numerical problem has a very definite structure. This is dictated by the biological reaction processes as well as the system configuration, particularly the manner in which the series of reactors in the system are interlinked. A significant part of any solution technique is to convert all this structural information into a form in which it can be exploited to reduce computational effort in finding the solution.

The matrices resulting from flowsheeting problems for systems comprising a number of units are often solved using techniques such as partitioning with precedence ordering and tearing (Westerberg et al, 1979). These techniques involve considering each unit separately, and partitioning the matrix into a number of smaller sub-matrices which are then solved individually. The most appropriate sequence in which to solve the individual units can be determined by a process of precedence ordering. In solving the individual units, we may require estimates of the values of the concentrations in streams from other units yet to be solved. Estimation of these concentrations is termed tearing of the system. As a result of this process of estimation, the solution procedure for the complete system of interlinked units is an iterative one. If the recycle flows are not particularly significant, then this approach is a suitable one. However, with biological systems, the recycle terms can be large, exerting a strong and often dominating influence on the system. Therefore, partitioning is not suitable. An appropriate solution procedure should handle the matrix as a single entity.

One of the significant features of the biological flowsheeting problem is the fact that the steady state matrix is usually sparse. Although many solution methods have been developed which exploit the sparsity of a matrix, most of these rely on the matrix being symmetrical and diagonally dominant, for example, in analysis of structures. In our situation, this is not usually the case, and many of these approaches are therefore not suitable.

A number of different approaches have been evaluated for computing the solution to the set of non-linear algebraic equations of the form encountered within biological reaction systems. These are the five methods generally used in chemical engineering flowsheeting applications. With each of these methods, an initial estimate of the state variables must be provided, and the technique is applied iteratively until convergence is reached.

Direct linearisation:
A method which requires the set of equations to be represented by an equivalent set of linear equations, which are then solved by Gauss elimination.

Successive substitution:
A fixed point iteration method which requires the re-arrangement of the non-linear equations $f_n(x_n) = 0$ in the form $x_n = g_n(x_n)$. The current estimate of the solution is substituted into the functions $g_n(x_n)$ to provide updated values.

Wegstein acceleration:
An acceleration technique which is applied to the method of successive substitution in an attempt to improve its convergence properties. This method also uses the equations in the form $x_n = g_n(x_n)$.

Newton's method:
A method based on the idea of constructing a local linear approximation to the functions by using a matrix of partial derivatives (the

Jacobian). The method is an n dimensional analogue of the Newton-Raphson method for solving a single non-linear equation in one unknown.

Broyden's method:
A quasi-Newton method based on the idea of approximating the Jacobian in order to avoid the computational effort required for its repeated evaluation.

3.5 DIRECT LINEARISATION

One method of solving a set of non-linear equations is by direct linearisation. The complete set of non-linear equations is represented by an equivalent set of linear equations, which are then solved using exact methods. The process of representation requires approximation, and this gives rise to an iterative procedure in which the linear equations become an improved approximation to the non-linear equations as the solution is approached.

Linear approximations to non-linear terms in the mass balance equations can be formulated in a number of ways. In selecting the appropriate linearisation, a set of linear equations must be chosen which gives rise to a process of iteration that eventually converges. This is not always possible; some of the possibilities may actually diverge. In the situation where more than one set converges, it is the different rates of convergence from a range of starting values that will determine the selection. It is difficult to generalise about the rate of convergence, or about the region from which convergence will be possible. Generally, however, it is possible to construct some form of linear approximation which, from a starting point sufficiently close to the solution, will eventually converge to that solution.

To illustrate the multiple possibilities of linearisation, consider the non-linear function:

$$f(X,Y) = X * Y \qquad (3.10)$$

In order to create a linear approximation, the first two terms of a Taylor's expansion about the point (X_0, Y_0) can be formulated. The point (X_0, Y_0) should lie in the region of interest. This will yield:

$$f(X,Y) = f(X,Y)_{(X_0,Y_0)} + \left.\frac{\partial f}{\partial X}\right|_{(X_0,Y_0)} \Delta X + \left.\frac{\partial f}{\partial Y}\right|_{(X_0,Y_0)} \Delta Y \qquad (3.11)$$

This simplifies to:

$$f(X,Y) = X_0 Y + X Y_0 - X_0 Y_0 \qquad (3.12)$$

Equation (3.12) represents one possible linear approximation to Eq(3.10). There are, however, other possible linear equations that could used to approximate the non-linear equation. Other options can usually be developed from a further examination of Eq (3.11) as well as utilising additional information as regards the point (X_0, Y_0). If, for example, $\Delta X \approx \Delta Y$, $X_0 \approx 1000$ and $Y_0 \approx 1$, then the differing orders of magnitude of the two terms could be used to make an important simplifying assumption to Eq (3.11). In this case, consider the contribution of the terms

$$\left.\frac{\partial f}{\partial X}\right|_{(X_0,Y_0)} \Delta X = Y_0 \Delta X \approx 1.\Delta X$$

and

$$\left.\frac{\partial f}{\partial Y}\right|_{(X_0,Y_0)} \Delta Y = X_0 \Delta Y \approx 1000.\Delta Y$$

When $\Delta X \approx \Delta Y$, the first term will be negligible in comparison to the second. Therefore, Eq (3.11) could be reduced to:

$$f(X,Y) = f(X,Y)_{(X_0,Y_0)} + \left.\frac{\partial f}{\partial Y}\right|_{(X_0,Y_0)} \Delta Y \qquad (3.13)$$

which yields the linear approximation

$$\begin{aligned} f(X,Y) &= X_0 Y_0 + X_0 (Y - Y_0) \\ &= X_0 Y \end{aligned} \qquad (3.14)$$

The approach leading to Eq (3.14) is one that can often be used successfully in the direct linearisation method for biological systems. This simplification is possible because these systems often incorporate particulate compounds at high concentrations and soluble compounds at low concentrations. The reason for interest in this approach is that it often leads to simpler equations. (compare Eqs (3.12) and (3.14)).

3.5.1 A numerical example

Westerberg et al (1979) presents a useful example which clarifies the approach presented above. He considers the dissociation, in dilute solution and at constant temperature, of a species A to two molecules of species B:

$A \rightleftharpoons 2B$

This leads to the following equations, where the first equation represents conservation of mass, and the second, thermodynamic equilibrium:

$$C_A + \tfrac{1}{2}C_B = C_A^*$$
$$K\,C_A - C_B^2 = 0 \qquad (3.15)$$

where C_A = concentration of species A
C_B = concentration of species B
C_A^* = initial concentration of C_A
K = equilibrium constant for the reaction

The example presents two possible forms of linearisation of these equations. One method would be to use the scheme of Eq (3.14), which would result in the equations being written in the form:

$$\begin{bmatrix} 1 & \tfrac{1}{2} \\ K & -C_{B(p)} \end{bmatrix} \cdot \begin{bmatrix} C_A \\ C_B \end{bmatrix} = \begin{bmatrix} C_A^* \\ 0 \end{bmatrix} \qquad (3.15)$$

This is equivalent to an iterative scheme:

$$C_{B(p+1)} = \frac{K\,C_A^*}{\frac{1}{2}K - C_{B(p)}} \qquad (3.16)$$

where $C_{B(p)}$ = the value for C_B after the p^{th} iteration

Alternatively, the equations can be linearised by incorporating all the first order terms in the Taylor's expansion as in Eq (3.12) and writing the equations in the form:

$$\begin{bmatrix} 1 & \frac{1}{2} \\ K & -2C_{B(p)} \end{bmatrix} \cdot \begin{bmatrix} C_A \\ C_B \end{bmatrix} = \begin{bmatrix} C_A^* \\ -C_{B(p)}^2 \end{bmatrix} \qquad (3.17)$$

This gives rise to the iterative scheme:

$$C_{B(p+1)} = \frac{K\,C_A^* + C_{B(p)}^2}{\frac{1}{2}K + 2C_{B(p)}} \qquad (3.18)$$

Table 3.1 shows the first few steps of these two iterative procedures for a particular case. While both converge to a solution, it is clear that the second method does so much more rapidly than the first. In fact it can be shown that the process described by Eq (3.18) is a second-order process, equivalent to a Newton-Raphson type iteration i.e. one where the number of significant digits of accuracy tends to double with each iteration as the solution is approached.

There is also another solution to the set of non-linear equations, at $C_B = -2.0$, but it is not a physically meaningful solution. However, the two processes have quite different convergence properties in the region of the negative solution. The second order process of Eq (3.18) converges to the genuine (but not physically meaningful) solution, while the first order process of Eq (3.16) diverges from the negative solution and ends up with the (physically meaningful) solution, $C_B = 1.0$. The divergence from the negative solution is, however, very slow in the neighbourhood of that solution, so that special tests might have to be devised to test for it. With this proviso, it may be shown that the process of Eq (3.16) will converge to the physical solution from any starting value $-\infty < C_{B(0)} < +\infty$, while $C_{B(0)} \neq -2.0$; the second order process of Eq (3.18) will converge to the physical solution from any

$C_{B(p+1)} = \dfrac{K\,C_A^*}{(\frac{1}{2}K - C_{B(p)})}$	$C_{B(p+1)} = \dfrac{K\,C_A^* + C_{B(p)}^2}{(\frac{1}{2}K + 2C_{B(p)})}$
For $K = 2$, $C_A^* = 1$, and $C_{B(0)} = 1.5$	
$C_{B(0)} = 1.5$	$C_{B(0)} = 1.5$
$C_{B(1)} = 0.8$	$C_{B(1)} = 1.0675$
$C_{B(2)} = 1.111111$	$C_{B(2)} = 1.001250$
$C_{B(3)} = 0.947368$	$C_{B(3)} = 1.000001$
$C_{B(4)} = 1.027027$	$C_{B(4)} = 1.000000$
$C_{B(5)} = 0.986667$	
$C_{B(6)} = 1.006711$	
$C_{B(7)} = 0.996656$	
$C_{B(8)} = 1.001675$	
$C_{B(9)} = 0.999163$	
$C_{B(10)} = 1.000417$	

Table 3.1 Comparison of two iterative solutions to linearised equations (Westerberg et al, 1979)

starting point $-0.5 < C_{B(0)} < +\infty$, and to the alternative solution from any starting point $-\infty < C_{B(0)} < -0.5$. The convergence from $C_{B(0)} = \pm n$ for the second order method, where n is any large number is, however, slow and of first order. In physical reality, $C_{B(0)}$ is bounded, $0 < C_{B(0)} < 2$, and from any point within that range, convergence is assured for either method, but more rapidly for the method of Eq (3.18).

3.5.2 Returning to the case study

Let us now return to the case study of the single reactor plus settler. Consider the non-linear term in Eq (3.1):

$$\frac{\hat{\mu}\ S_S\ X_B}{(K_S + S_S)}\ V \qquad (3.19)$$

A linear approximation to this term can be created in a number of ways. These include, amongst others, the following two possibilities:

(i) A complete Taylor's expansion about the point (X_{B0}, S_{S0}).

(ii) The same Taylor's expansion could be employed, but the resulting linearisation could be further simplified by using the fact that we have additional information as regards the nature of certain of the terms in the equation.

For the second option, we note that, for every unit of soluble substrate (S_S) utilised, Y units of biomass (X_B) are created. Because $Y \approx 0.66$, we can assume that ΔX_B and ΔS_S are of similar magnitude. i.e.

$$\Delta\ X_B \approx \Delta\ S_S \qquad (3.20)$$

In addition, in the situations encountered in practice, the concentration of S_S is generally low (≈ 1) and the concentration of X_B is generally high (≈ 1000). Thus, the non linear term of Eq (3.1) could be linearised using the simplifying assumptions outlined for Eq (3.14) in the region (X_{B0}, S_{S0}) as follows:

$$f(X_B, S_B) = \frac{\hat{\mu}\, X_{B0}\, V}{(K_B + S_{B0})} S_B \qquad (3.21)$$

This is the approach that has been used in the method of direct linearisation employed in the simulation program here. Similar simplifying assumptions may be applied to all the non-linear terms in the mass balance equations. The resulting matrix for the case study is illustrated in Fig (3.3). This matrix represents a linear approximation to the set of non-linear equations, and may be solved by simple Gaussian elimination at each iteration.

3.5.3 The algorithm for direct linearisation

Step 1: Set up linear approximations for all the non-linear terms in the mass balance equations.

Step 2: Create the A matrix and the B vector.

Step 3: Initialise the A matrix with seed values of the state variables.

Step 4: Find new values of the state variables, X, by establishing the solution to the matrix problem
A X = B
using Gaussian elimination.

Step 5: Test for convergence.
If the convergence criterion is satisfied, then terminate the iteration. Otherwise, insert the new values of the state variables into the A matrix and B vector and return to Step 4.

3.5.4 Considerations in application of the method

Although the method of direct linearisation as described above has been successfully applied to a variety of biological system configurations, particular drawbacks to its application should be noted:

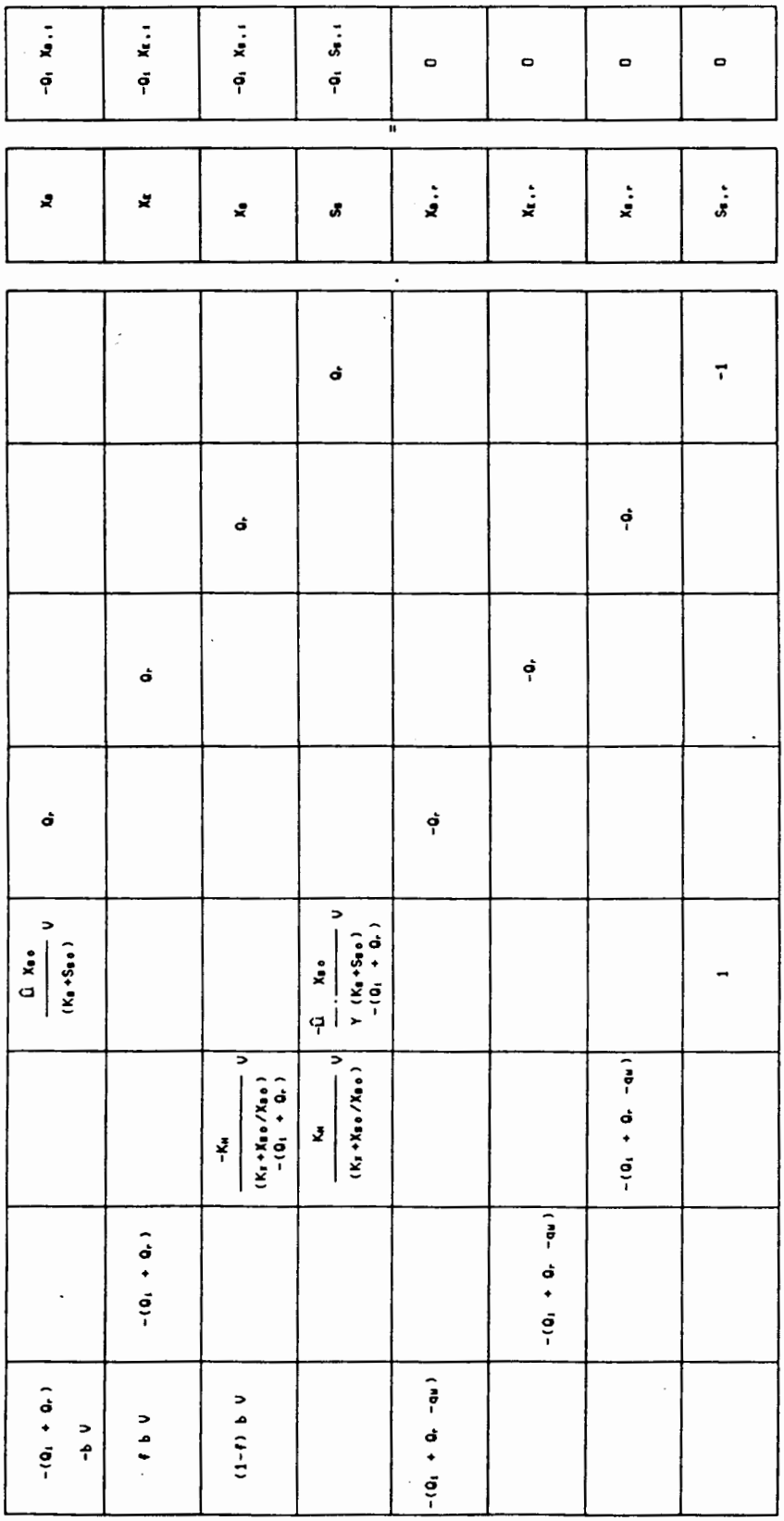


Figure 3.3 A matrix representation illustrating the effect of linearising the non-linear terms in the mass balance equations.

(i) Non-linear terms in the equations must be linearised. This can require extensive mathematical manipulation before the method can be implemented.

(ii) Some linear approximations lead to systems of equations which do not converge. Therefore, a certain amount of skill, and perhaps trial and error, is necessary in selecting suitable linearisations.

(iii) A set of linear equaitons must be set up for each system configuration and each biological model. Any changes to the model or the configuration will necessitate a reworking of the equations.

3.6 SUCCESSIVE SUBSTITUTION

The method of successive substitution is one of a class of indirect methods which can be used to find the solution to a set of non-linear equations. The major advantage of this method is that it is simple to apply, although it can exhibit erratic behaviour and often does not converge. The general approach of this technique is to rearrange the equations into a form in which they can be used to generate, given initial estimates, new estimates for the solution that is sought. An iterative procedure is then followed until convergence is achieved. Figure 3.4 indicates the general nature of this iteration scheme. (Westerberg et al, 1979)

For the one-dimensional case, the rearrangement of the equation $f(x) = 0$ takes the form

$$x = g(x) \qquad (3.22)$$

The current estimate of x can be substituted into the function $g(x)$ to provide the next estimate of x. The iteration procedure is initiated with a guessed value of x and is eventually terminated when successive estimates of the root do not change significantly. The nature of the

function g(x) directly influences the convergence properties of the method. If the equation can be rearranged so that g(x)' is relatively insensitive to x, then the method of successive substitution can be a robust one, which quickly converges to a good approximate answer.

Reklaitis (1983) demonstrates that, for the one-dimensional case, if

$$\left|\frac{dg}{dx}\right| < 1$$

in the region between the initial estimate, $x_{(0)}$, and the root, x^*, then the iterations will converge.

An n dimensional extension of the method of successive substitution involves rewriting the n non-linear equations in the form:

$$\begin{aligned} x_1 &= g_1(x_1, x_2, \ldots . x_n) \\ x_2 &= g_2(x_1, x_2, \ldots . x_n) \\ &\vdots \\ x_n &= g_n(x_1, x_2, \ldots . x_n) \end{aligned} \qquad (3.23)$$

In general form

$$x = g(x)$$

$$\text{where } x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Each equation has one of the variables isolated on the left hand side and each variable appears on the left hand side of only one equation.

The iteration scheme that is subsequently followed exactly parallels that for the one dimensional calculations. Starting with assumed trial

values for x, each equation $g_i(x)$ is evaluated at this point to yield new values of x. This vector of x values then serves to initiate the next round of evaluations of the functions, until successive estimates of the x vector do not change significantly.

Multidimensional successive substitution exhibits similar convergence characteristics to that of the single dimensional method. It is always possible that the method may fail to converge, or may only converge very slowly. In addition, the form of the functions g(x) as well as the choice of starting values of x exert a strong influence on the ultimate convergence properties.

Reklaitis (1983) provides a convergence test for the multivariable case of successive substitution. It can be shown that if the sum of the partial derivatives of each function $g_i(x)$ is less than one at each successive iterate between the initial point $x_{(0)}$ and the solution x^*, then the iteration will converge. Unfortunately, this convergence test depends on knowing the solution in advance, and is therefore not particularly useful. In practice, if a problem should fail to converge whilst using the method of successive substitution, then one of two possible remedies can be employed:

(i) the iterations can be restarted with a new set of initial guesses of x; or

(ii) the functions can be rearranged to yield a different set of equations g(x) and a new attempt initiated.

3.6.1 A numerical example

Reklaitis (1983) provides a two dimensional example of the method of successive substitution which demonstrates some of the characteristic behaviour of the technique as outlined above. He considers two equations, rearranged in the form:

$$x_1 = 0.4\, x_1^2 + 0.1\, x_2 + 0.5$$

$$x_2 = 4(x_1 + 3x_2)^{-1} \qquad (3.24)$$

The solution to these equations is $x^* = (1,1)$. By selecting different sets of starting values, Reklaitis demonstrates how the initial guess can affect the convergence behaviour of the method. Specifically,

(i) If $x_{(0)} = (½,2)$, then the method converges, although slowly.

(ii) If $x_{(0)} = (2, ½)$, then the method diverges.

The results for the first six iterations for each set of starting values are presented in Table 3.2.

This example can be extended to demonstrate how the nature of the rearrangement of the functions exerts an influence on the speed of convergence, and whether or not the method converges at all. Consider the following alternative form of rearrangement of the functions:

$$x_1 = 0.2\, x_1^2 + 0.1\, x_2 + 0.7$$
$$x_2 = 2(x_1 + 3x_2)^{-1} + 0.5 \qquad (3.25)$$

Using the same set of starting values as above, the following behaviour is noted:

(i) If $x_{(0)} = (½,2)$, then the method converges more rapidly than with the earlier rearrangement.

(ii) If $x_{(0)} = (2, ½)$, then the method converges, where previously it diverged.

The results for the first six iterations for each set of starting values are presented in Table 3.3.

3.6.2 Returning to the case study

Consider the eight steady state mass balance equations describing the single reactor plus settler problem. [Eqs (3.1) to (3.8)]. The presence of the non-linear terms enables a number of rearrangements of the equations in the form $x = g(x)$. Two of the possible options are:

Starting values	$x_{(0)}$ = (½, 2) Converging		$x_{(0)}$ = (2, ½) Diverging	
Iteration number	x_1	x_2	x_1	x_2
0	0.5	2	2	0.5
1	0.8	0.6154	2.15	1.1429
2	0.8175	1.5116	2.4633	0.7170
3	0.9185	0.7473	2.9988	0.8669
4	0.9122	1.2656	4.1380	0.7144
5	0.9594	0.8494	7.5731	0.6322
6	0.9531	1.1403		
Solution	1.0000	1.0000	1.0000	1.000

Table 3.2 Comparison of the effect of different starting values on the convergence of a successive substitution scheme (Reklaitis, 1983)

Starting values	$x_{(0)}$ = (½, 2) Rapid Convergence		$x_{(0)}$ = (½, 2) Convergence	
Iteration number	x_1	x_2	x_1	x_2
0	0.5	2	2	0.5
1	0.95	0.8077	1.550	1.071
2	0.9613	1.0929	1.288	0.920
3	0.9941	0.9717	1.124	0.994
4	0.9948	1.0116	1.052	0.987
5	0.9991	0.9963	1.020	0.998
6	0.9993	1.0015		
Solution	1.0000	1.0000	1.0000	1.000

Table 3.3 Comparison of the effect of the form of rearrangement of the equations on the convergence of a successive substituion scheme (Reklaitis, 1983)

Option 1:

$$X_B = \{ Q_r X_{B,r} + Q_i X_{B,i} \} / \{ (Q_i + Q_r) + bV - \frac{\hat{\mu} S_S V}{(K_S + S_S)} \} \quad (3.26)$$

$$X_E = (Q_r X_{E,r} + Q_i X_{E,i} + f b X_B V) / (Q_i + Q_r) \quad (3.27)$$

$$X_S = \{ Q_r X_{S,r} + Q_i X_{S,i} + (1-f) b X_B V \} / \{ (Q_i + Q_r) - \frac{K_H V}{(K_X + X_S / X_B)} \} \quad (3.28)$$

$$S_S = \{ Q_r S_{S,r} + Q_i X_{S,i} + \frac{X_S K_H}{(K_X + X_S / X_B)} V \} / \{ (Q_i + Q_r) + \frac{\hat{\mu}}{Y} \cdot \frac{X_B}{(K_S + S_S)} V \} \quad (3.29)$$

$$X_{B,r} = (Q_i + Q_r - q_w) X_B / Q_r \quad (3.30)$$

$$X_{E,r} = (Q_i + Q_r - q_w) X_E / Q_r \quad (3.31)$$

$$X_{S,r} = (Q_i + Q_r - q_w) X_S / Q_r \quad (3.32)$$

$$S_S = S_{S,r} \quad (3.33)$$

Option 2:

$$S_S = \{ (Q_i + Q_r) X_B + b X_B V - Q_r X_{B,r} - Q_i X_{B,i} \} / \{ \frac{\hat{\mu} X_B V}{(K_S + S_S)} \} \quad (3.34)$$

$$X_E = (Q_r X_{E,r} + Q_i X_{E,i} + f b X_B V) / (Q_i + Q_r) \quad (3.35)$$

$$X_B = \{ (Q_i + Q_r) X_S - Q_r X_{S,r} - Q_i X_{S,i} + \frac{X_S K_H}{(K_X + X_S / X_B)} V \} / \{ (1-f) b V \} \quad (3.36)$$

$$X_S = \{ (Q_i + Q_r) S_S + \frac{\hat{\mu}}{Y} \cdot \frac{S_S X_B}{(K_S + S_S)} V - Q_r S_{S,r} - Q_i X_{S,i} \} / \{ \frac{K_H V}{(K_X + X_B / X_B)} \} \quad (3.37)$$

$$X_{B,r} = (Q_i + Q_r - q_W)\, X_B / Q_r \qquad (3.38)$$

$$X_{E,r} = (Q_i + Q_r - q_W)\, X_E / Q_r \qquad (3.39)$$

$$X_{S,r} = (Q_i + Q_r - q_W)\, X_S / Q_r \qquad (3.40)$$

$$S_S = S_{S,r} \qquad (3.41)$$

In this case, Option 1 converged for a range of initial values, whereas Option 2 became unstable and eventually diverged. The reason for this behaviour lies in the interaction between the soluble and particulate compounds in the biological model and the resultant effect that this has on the sensitivity of the functions. The divergence is largely due to the extreme sensitivity of Eq (3.36) to X_B even in the region of the solution.

3.6.3 The algorithm for successive substitution

Step 1: Select an initial estimate for the state variables, $x_{(0)}$, and a suitable convergence criterion.

Step 2: Calculate

$$x_{(p+1)} = g(x_{(p)})$$

Step 3: Test for convergence

If $\Sigma \left| g(x_{(p)}) - x_{(p+1)} \right|^2 <$ convergence tolerance

then terminate the iteration.

Otherwise, replace $x_{(p)}$ by $x_{(p+1)}$ and return to Step 2

3.6.4 Considerations in the method

Although the method of successive substitution has the advantage of being simple and straightforward in its application, certain drawbacks are apparent:

(i) A certain amount of mathematical manipulation is necessary before the method can be applied, as the equations need to be rearranged in a form suitable for the fixed point iteration.

(ii) The convergence behaviour of the method depends on the form of rearrangement. Functions that display sensitivity to any of the state variables could become unstable and prevent the system from converging.

(iii) Careful consideration needs to be given to the selection of starting values. The initial estimates of the state variables often need to be very close to the solution in order to ensure convergence.

(iv) The set of equations $x = g(x)$ is specific to both the biological model and the system configuration. Any changes to the model or configuration would necessitate a complete reworking of the equations.

3.7 THE SECANT METHOD OF WEGSTEIN

A drawback of the successive substitution method is that its rate of convergence is only linear. A number of "acceleration procedures" have been proposed in order to improve this rate. The most widely used is Aitken's (1925) "δ^2 acceleration" method, which uses linear extrapolation through two points generated initially by a successive substitution formula. The same idea was later "rediscovered" by Steffensen (1933) and even later by Wegstein (1958), and hence it is known under these various names (Sargent, 1981). The method is a one dimensional acceleration method in which each variable is treated separately by driving it with a uniquely associated function. Interactions with other variables are consequently ignored at each iteration.

The Wegstein method can be presented most simply be considering the one-dimensional case. The method attempts to find the root of the equation $f(x) = 0$ by first rearranging it in the form:

$$x = g(x) \tag{3.42}$$

To initiate the method, one successive substitution step is taken. That is, if $x_{(0)}$ is the initial estimate of the solution, then the second point is:

$$x_{(1)} = g(x)_{(0)} \tag{3.43}$$

If at this stage the convergence criterion is not yet satisfied, a third (and subsequent) points are calculated as follows (see Fig 3.5): The function $g(x)$ is approximated by a line joining the two points $[x_{(0)}, g(x)_{(0)}]$ and $[x_{(1)}, g(x)_{(1)}]$. The new approximation to the root, $x_{(2)}$, is given by the intersection of this secant line with the line $y = x$. i.e.

$$x_{(2)} = t \cdot g(x)_{(1)} + (1-t) \cdot x_{(1)} \tag{3.44}$$

where $t = \dfrac{1}{(1-m)}$

and m = slope

$$= \frac{g(x)_{(1)} - g(x)_{(0)}}{x_{(1)} - x_{(0)}}$$

Equation (3.44) is the iteration formula of Wegstein. Figure 3.5 illustrates the method for a function with derivative greater than unity. As shown in the figure, when $x_{(p+1)}$ lies between $x_{(p)}$ and $x_{(p-1)}$, Wegstein's formula will interpolate for the value $x_{(p+1)}$. This will be the case when $-\infty < m < 0$ and $0 < t < 1$. At the limit $m = -\infty$, $t = 0$, and successive estimates of x will be identical. At $m = 0$, $t = 1$, and Wegstein's formula is reduced to

$$x_{(p+1)} = g(x)_{(p)}$$

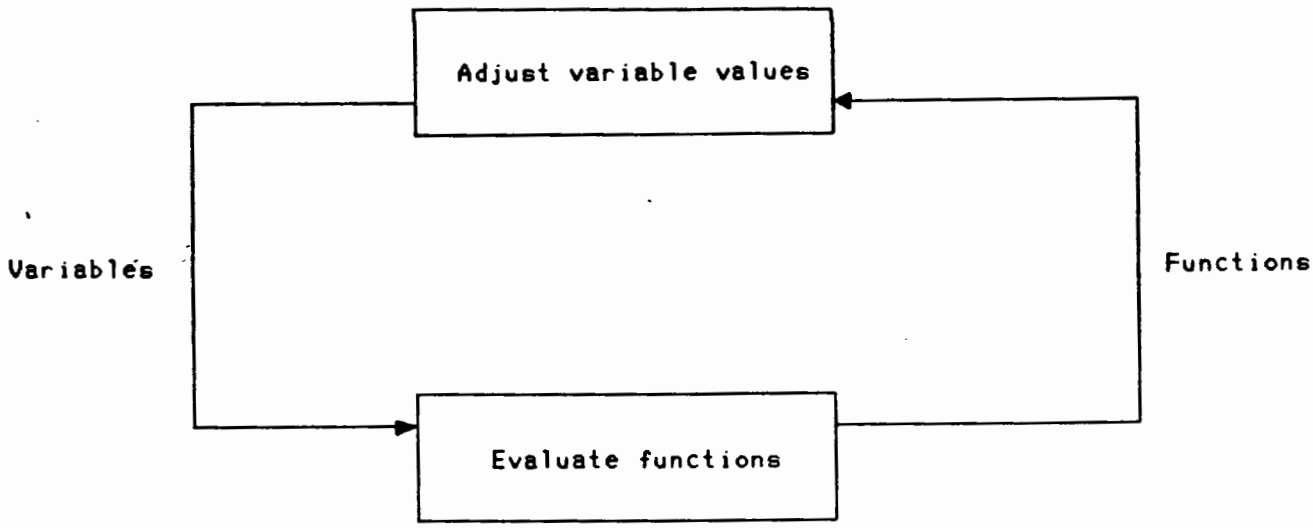


Figure 3.4 Indirect methods: a general scheme

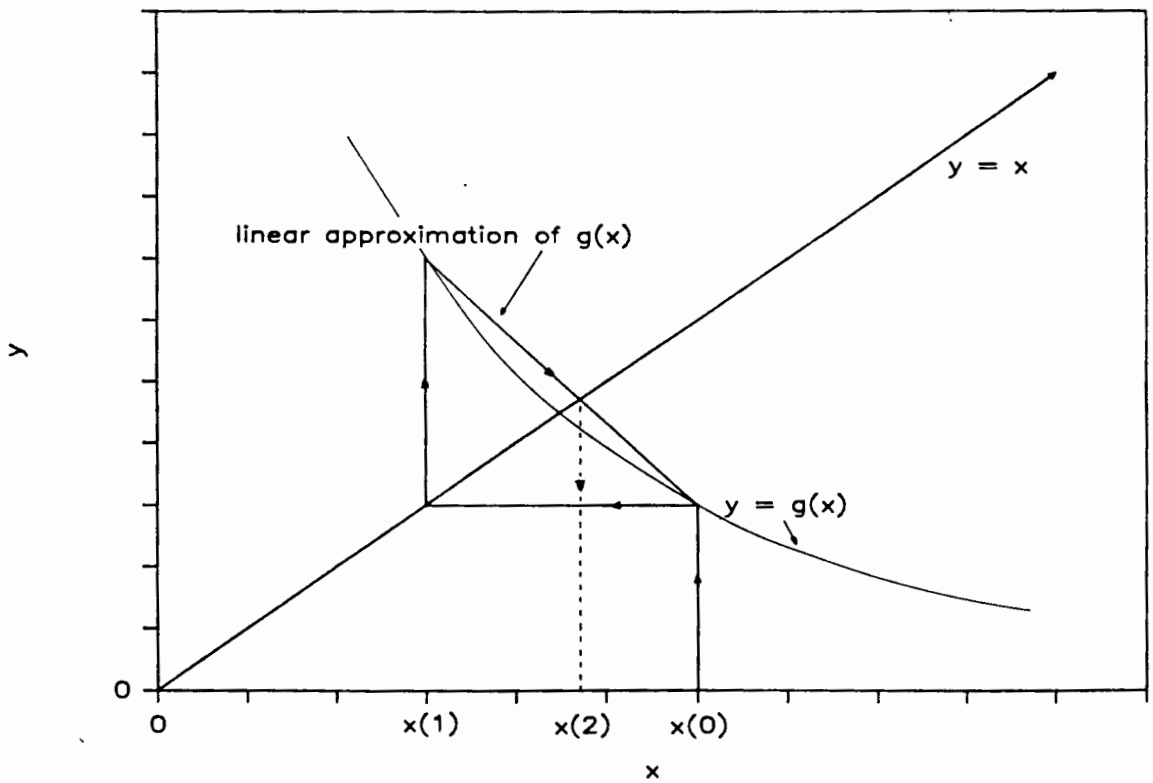


Figure 3.5 A graphical illustration of Wegstein's method in one dimension

which is the method of successive substitution.

When $x_{(p+1)}$ does not lie between the previous two estimates, Wegstein's formula will extrapolate to get the next value. This is the case when $0 < m < \infty$ and $-\infty < t < 0$ or $1 < t < \infty$. At $m = 1$, t is undefined, and Wegstein's extrapolation is not valid. To circumvent this problem, the most common procedure is to limit the degree of extrapolation by setting upper and lower limits for t. The version of Wegstein's method which uses this type of constraint is sometimes referred to as the bounded Wegstein method.

An n dimensional extension of Wegstein's method uses a similar construction to that of the one dimensional method to create linear approximations to the functions g(x). Since in this case n variables are involved, calculation of the slopes of the linear approximations should in principle involve a matrix of n partial derivatives. However, this would involve evaluating the functions g(x) at n different points for each of the n variables before all the slopes, **m**, could be calculated. To avoid this computational effort, a simplification is made. It is assumed that, for each function $g_i(x)$, the only significant slope is that with respect to x_i; all others can be ignored. In other words, the g(x)'s are approximated by:

$$g_1(x_1, x_2, \ldots, x_n)_{(1)} - g_1(x_1, x_2, \ldots, x_n)_{(0)} = m_1 \cdot (x_{1(1)} - x_{1(0)})$$
$$g_2(x_1, x_2, \ldots, x_n)_{(1)} - g_2(x_1, x_2, \ldots, x_n)_{(0)} = m_2 \cdot (x_{2(1)} - x_{2(0)})$$
$$\vdots$$
$$g_n(x_1, x_2, \ldots, x_n)_{(1)} - g_n(x_1, x_2, \ldots, x_n)_{(0)} = m_n \cdot (x_{n(1)} - x_{n(0)})$$

In general form

$$g(x)_{(1)} - g(x)_{(0)} = \mathbf{m}(x_{(1)} - x_{(0)}) \qquad (3.47)$$

where $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ and $\mathbf{m} = [\, m_1, m_2, \ldots, m_n \,]$

With this simplification, the evaluation of the slope vector, **m**, for each variable can be carried out using only one additional point:

$$m = \frac{g(x)_{(1)} - g(x)_{(0)}}{x_{(1)} - x_{(0)}}$$

Roughly the same convergence conditions that are applicable to successive substitution will apply to Wegstein's method. In general, the method has a lesser tendency to diverge than does successive substitution. The major advantage of the method, however, is that, when it converges, the rate can be faster. In the worst case, in which the accelerating functions must be reset to the bounds, Wegstein's method is reduced to a scaled form of successive substitution. Hence, the worst rate of convergence will be that of successive substitution.

Reklaitis (1983) notes that Wegstein's method may encounter difficulties if the slope for any variable does not dominate the slopes associated with the other variables which have been neglected in deriving the method. In practice, testing the validity of this assumption would require the evaluation of all of the partial derivatives of the functions $g_i(x)$.

Westerberg et al (1979) comments that this method could suffer from instability in a multidimensional environment, since large acceleration factors are encountered in most problems. He suggests using the bounded Wegstein method with delay, which would involve applying the acceleration function only every few iterations.

In spite of the shortcomings of Wegstein's method, it remains a commonly used algorithm, and has been accepted as the "best" one dimensional method available (Westerberg et al, 1979).

3.7.1 A numerical example

Reklaitis (1983) uses the same set of equations as in Section 3.6.1 [Eq (3.25)] to illustrate the properties of Wegstein's method:

$$x_1 = 0.2\, x_1^2 + 0.1\, x_2 + 0.7$$
$$x_2 = 2(x_1 + 3x_2)^{-1} + 0.5 \qquad (3.48)$$

The example uses the starting value x = (½,2) and sets the upper and lower bounds of t at 10 and -10 repectively. The results for the first six iterations are presented in Table 3.4. It can be noted that, in this example, convergence is not significantly better than that observed with successive substitution (cf. Table 3.4).

3.7.2 The Wegstein algorithm

Step 1: Select an initial estimate for the state variables, $x_{(0)}$, a suitable convergence criterion, and upper and lower bounds for t. ($|t_{upper}| = |t_{lower}| = t_{max}$.)

Step 2: Calculate

$$x_{(1)} = g(x)_{(0)}$$

Step 3: Calculate the slopes

$$m = \frac{g(x)_{(p)} - g(x)_{(p-1)}}{x_{(p)} - x_{(p-1)}}$$

Step 4: Calculate

$$t = \frac{1}{(1-m)}$$

If $|t_i| > t_{max}$ then $t_i = t_{max}$

Step 5: Calculate

$$x_{(p+1)} = (1-t) \cdot x_{(p)} + t \cdot g(x)_{(p)}$$

Step 6: Test for convergence

If $\Sigma|g_i(x)_{(p)} - x_{i(p+1)}|^2 <$ convergence tolerance

then terminate the iteration.

Otherwise, replace $x_{(p)}$ by $x_{(p+1)}$ and return to Step 3.

3.7.3 Considerations in the method

Wegstein's method retains the advantages and drawbacks of the simple successive substitution approach, whilst usually exhibiting improved convergence characteristics. These features have already been outlined in Section 3.6.4. A additional drawback is the possibility that the method may introduce instability where successive substitution converges in a stable manner.

3.8 NEWTON'S METHOD

Newton's method is a more sophisticated root-finding technique which overcomes the problems of the relatively slow and often unpredictable convergence properties of the successive substitution and Wegstein methods. It has a much improved rate of convergence, although this is at the computational expense of requiring values of the partial derivatives of the functions.

The method is based on the idea of approximating a set of non-linear functions of the form $f(x) = 0$ by local linear approximations with slopes given by the derivatives of the functions. These functions are then used in an iterative procedure that generates new, and hopefully better, approximations to the solution.

Newton's method is best illustrated by considering the one dimensional case again (see Fig 3.6). The non-linear equation is first expressed in the form:

$$f(x) = 0 \qquad (3.49)$$

Given an initial estimate of the root, $x_{(0)}$, f(x) can now be approximated by expanding f(x) linearly in a Taylor series about the current estimate $x_{(0)}$:

$$f(x) \approx f(x_{(0)}) + \left.\frac{df}{dx}\right|_{x(0)} \cdot (x-x_{(0)}) \qquad (3.50)$$

Since it is the root of f(x) that is being sought, f(x) is set equal to zero in the approximating function, and the equation is solved for x as follows:

$$f(x_{(0)}) + \left.\frac{df}{dx}\right|_{x(0)} \cdot (x-x_{(0)}) = 0$$

which yields:

$$x = x_{(0)} - \left[\left.\frac{df}{dx}\right|_{x(0)}\right]^{-1} \cdot f(x_{(0)}) \qquad (3.51)$$

Figure 3.6 is a graphical representation of Newton's method for a single equation. Since the derivative is the best local approximation to the slope of f(x), the resulting iteration formula can be expected to exhibit a better rate of convergence than the root-finding methods considered so far. This is in fact usually the case.

Newton's method can be erratic in regions where f'(x) is small. Johnston (1982) notes that direct convergence to a root x^* can only be guaranteed if the condition

$$\left|\frac{f(x) \cdot f''(x)}{(f'(x))^2}\right| < 1$$

is satisfied at each step in the iterative procedure.

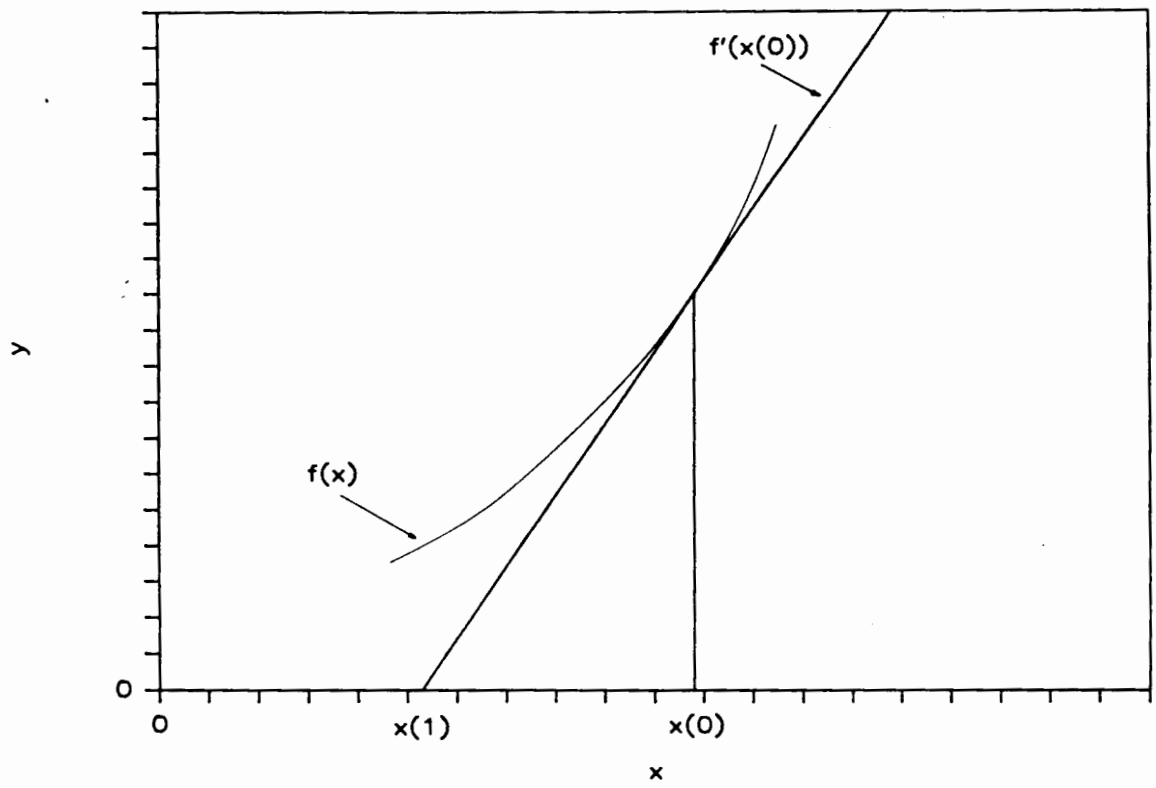


Figure 3.6 A graphical illustration of Newton's method for a single non-linear equation

The main advantage of Newton's method is that its rate of convergence is quadratic. This is, however, countered by the fact that each iteration requires two function evaluations: f(x) and f'(x). In addition, it has the disadvantages of being sensitive to the initial estimate of the root and, even more importantly, of requiring an explicit representation of the derivative of the function. In many applications, the latter can be a serious drawback.

In order to extend Newton's method to more than one dimension, an analogue of the derivative f'(x) is needed. In n dimensions, this is an n by n matrix termed the Jacobian, J, with entries that are the partial derivatives:

$$\begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_1} & \cdots\cdots\cdots & \frac{\partial f_n(x)}{\partial x_1} \\ \frac{\partial f_1(x)}{\partial x_2} & \frac{\partial f_2(x)}{\partial x_2} & \cdots\cdots\cdots & \frac{\partial f_n(x)}{\partial x_2} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_1(x)}{\partial x_n} & \frac{\partial f_2(x)}{\partial x_n} & \cdots\cdots\cdots & \frac{\partial f_n(x)}{\partial x_n} \end{bmatrix}$$

Thus, an n dimensional analogue of Newton's one dimensional method is:

$$x_{1(p+1)} = x_{1(p)} - [J(x_1, x_2, \ldots, x_n)_{(p)}]^{-1} \cdot f_1(x_1, x_2, \ldots, x_n)_{(p)}$$

$$x_{2(p+1)} = x_{2(p)} - [J(x_1, x_2, \ldots, x_n)_{(p)}]^{-1} \cdot f_2(x_1, x_2, \ldots, x_n)_{(p)}$$

$$\vdots$$

$$x_{n(p+1)} = x_{n(p)} - [J(x_1, x_2, \ldots, x_n)_{(p)}]^{-1} \cdot f_n(x_1, x_2, \ldots, x_n)_{(p)}$$

That is:

$$x_{(p+1)} = x_{(p)} - [J(x)_{(p)}]^{-1} \cdot f(x)_{(p)} \qquad (3.52)$$

Equation (3.52) is Newton's method for a multidimensional environment. The properties of Newton's method for n dimensions are similar to those for one dimension.

(i) It has been noted that, in regions where $|f'(x)|$ is small, the one dimensional method behaves erratically. The corresponding property in n dimensions is near-singularity of the Jacobian matrix, J(x). Therefore, Newton's method will behave erratically if x lies in a region where J(x) is nearly singular. This is a far more significant disadvantage than it was for single equations because the cost of each Jacobian evaluation is quite high. Consequently, to ensure direct convergence, a very good initial estimate of $x_{(0)}$ may be necessary.

(ii) Like Newton's method in a single dimension, the n dimensional analogue exhibits quadratic convergence in the region close to the root.

Setting up the Jacobian for Newton's method requires explicit expressions for each of the n^2 partial derivatives. This may appear tedious to implement. Also, repeated evaluation of the many terms in the Jacobian may appear costly in terms of computer time. However, the quadratic rate of convergence is a strong aspect in its favour.

3.8.1 A numerical example

Reklaitis (1983) uses a set of two non-linear equations to demonstrate the superior convergence properties of Newton's method over Wegstein's method. The functions are as follows:

$$f_1 = 0.25\, x_2^2 + 0.75$$
$$f_2 = 3\,(2x_1 + x_2)^{-1} \qquad (3.54)$$

The partial derivatives of the functions are:

$$\frac{\partial f_1}{\partial x_1} = 0 \qquad \frac{\partial f_1}{\partial x_2} = 0.5\, x_2$$

$$\frac{\partial f_2}{\partial x_1} = -6\,(2x_1 + x_2)^{-2} \qquad \frac{\partial f_2}{\partial x_2} = -3\,(2x_1 + x_2)^{-2} \tag{3.55}$$

Table 3.5 presents the results for the first few iterations from a starting value $x_{(0)} = (2,2)$, as generated by the Wegstein and Newton's methods respectively. After 3 iterations, Newton's method has already satisfied the convergence criterion, and the iterations are terminated. The Wegstein method, however, requires 13 iterations to converge to the same tolerance. The reason for the slow convergence of the Wegstein iterations is that the assumption concerning the dominance of the slope vector is not satisfied. For example, at $x_{(0)} = (2,2)$:

$$\left|\frac{\partial f_1}{\partial x_1}\right| = 0 \text{ which is less than } \left|\frac{\partial f_1}{\partial x_2}\right| = 1$$

and

$$\left|\frac{\partial f_2}{\partial x_2}\right| = \frac{1}{12} \text{ which is less than } \left|\frac{\partial f_2}{\partial x_1}\right| = \frac{1}{6}$$

and similarly at $x^* = (1,1)$

3.8.2 An extension to Newton's method

Newton's method requires the evaluation of the partial derivatives of each of the n functions with respect to each of the n variables in order to evaluate the Jacobian matrix. This means that, if Newton's method is used to solve the biological system equations, any changes to the model or to the process configuration would require the re-evaluation of all of the partial derivatives. To avoid this problem, a finite difference approximation of the Jacobian may be used. Each term is evaluated as follows:

$$\frac{\partial f_i(x)}{\partial x_j} = \frac{f_i(x_1, x_2, \ldots, x_j + \Delta x_j, \ldots, x_n) - f_i(x_1, x_2, \ldots, x_j, \ldots, x_n)}{\Delta x_j} \tag{3.56}$$

where Δx_j = a small perturbation to x_j

$\approx 10^{-6} \cdot x_j$

Starting value	$x_{(0)} = (½, 2)$	
Iteration number	x_1	x_2
0	0.5	2
1	0.95	0.8077
2	0.9616	1.0378
3	0.9417	1.0052
4	1.0209	1.0053
5	1.0013	1.0052
6	1.0009	1.0049
Solution	1.0000	1.0000

Table 3.4 Wegstein's method applied to two equations (Reklaitis, 1983)

	Wegstein's Method		Newton's Method	
Iteration number	x_1	x_2	x_1	x_2
0	2	2	2	2
1	1.75	0.5	0.5833	0.8333
2	0.8125	0.7143	0.9510	0.8991
3	0.8734	1.2824	0.9967	0.9982
5	0.8592	1.0884		
8	0.9764	0.9817		
11	1.0058	0.9902		
13	0.9974	1.0046		
Solution	1.0000	1.0000	1.0000	1.000

Table 3.5 Comparison of the convergence properties of Wegstein's and Newton's methods for two non-linear equations (Reklaitis, 1983)

Dennis and Schnabel (1983) show that, when the analytical Jacobian in Newton's method is replaced by a finite difference approximation, the quadratic convergence properties of Newton's method can be retained provided the functions are not too non-linear. In fact, for most problems, Newton's method using analytical derivatives and Newton's method using properly chosen finite differences are virtually indistinguishable.

A finite difference approach would not save on the major expense involved in evaluating the n * n partial derivative matrix - in fact this can be a more costly process than when analytical derivatives are used. It does, however, render a simulation program more generally applicable because of not requiring further analysis when changes occur in the functions as a result of adjustments to the biological model or the system configuration.

3.8.3 Returning to the case study

To test the validity of using a finite difference approximation to the Jacobian, the single reactor plus settling tank test case was analysed. For the same set of starting values, the non-linear equations for the test case were solved using both an analytical and a finite difference Jacobian matrix. For the analytical Jacobian, the derivatives were written into the program code and were thus specific to this particular example.

For both cases, four iterations were required before the convergence criterion was satisfied. That is, in this case, the finite difference Newton's method can be used interchangably with the one using the analytical Jacobian. This would imply that the functions in this test case are not too non-linear. Therefore, it is likely that the discrete Newton method can be used successfully for other biological systems, thus obviating the need to re-program analytical derivatives every time the system configuration or biological model is altered.

3.8.4 The Newton algorithm

Step 1: Express the non-linear functions in the form $f(x) = 0$. Select initial estimates for the roots $x_{(0)}$ and a suitable convergence criterion.

Step 2: Evaluate $J(x)_{(p)}$

Step 3: Calculate $x_{(p+1)} = x_{(p)} - [J(x)_{(p)}]^{-1} \cdot f(x)_{(p)}$ as follows:

(i) Solve the set of linear equations:

$$J(x)_{(p)} \cdot h_{(p)} = - f(x)_{(p)}$$

for $h_{(p)}$

(ii) $x_{(p+1)} = x_{(p)} + h_{(p)}$

Step 4: Test for convergence

If $\Sigma |f_i(x)_{(p)}|^2 <$ convergence tolerance

then terminate the iteration.

Otherwise, replace $x_{(p)}$ by $x_{(p+1)}$ and return to Step 2.

3.8.5 Considerations in the method

Newton's method is generally superior to the successive substitution method and the secant method of Wegstein. The major advantages and disadvantages of the method may be summarised as follows:

Advantages:

(i) It exhibits quadratic convergence properties.

(ii) It has been found to be extremely efficient for problems that are near linear (Johnston, 1982).

(iii) When the finite difference approximation to the Jacobian is used, the method has a very general applicability. Any changes to the biological model or system configuration can thus be easily

incorporated into a computer program without having to re-evaluate the partial derivatives.

Disadvantages:

(i) In regions where the Jacobian is nearly singular, the method can behave erratically.

(ii) Implementation of the method is a costly exercise, as both the functions and the Jacobian matrix need to be recalculated at every iteration.

3.9 BROYDEN'S METHOD

Broyden's algorithm (Broyden, 1965, 1969) is a modification of Newton's method that was designed specifically to reduce the number of function evaluations necessary in finding a solution to a set of non-linear simultaneous algebraic equations. It is one of a whole class of methods which may be termed "quasi-Newton methods". These are techniques based on the idea of approximating the Jacobian in order to avoid the computational effort required to evaluate it fully. For a one dimensional environment, the secant method fills this role since it is based on approximating the derivative, $f'(x)$ of a single function $f(x) = 0$. Hence, any quasi-Newton method may be regarded as an n dimensional extension of the secant method. For any of these techniques, it is only the method for approximating the Jacobian matrix that will be different, the rest of the Newton algorithm remains unchanged.

The secant method in one dimension uses a finite difference approximation, $b_{(p)}$ to the derivative $f'(x)_{(p)}$, which is defined as follows:

$$f'(x)_{(p)} \approx b_{(p)} = \frac{f(x)_{(p)} - f(x)_{(p-1)}}{(x)_{(p)} - (x)_{(p-1)}} \qquad (3.57)$$

For a system of n non-linear equations, an analogue of $b_{(p)}$, which in this case approximates the n by n Jacobian matrix, can be defined as follows:

$$B_{(p)} = \frac{f(x)_{(p)} - f(x)_{(p-1)}}{(x)_{(p)} - (x)_{(p-1)}} \qquad (3.58)$$

It can be seen that Eq (3.57) for the one dimensional case allows only one solution for $b_{(p)}$. In contrast, examination of Eq (3.58) shows that it does not define $B_{(p)}$ uniquely when $n > 1$. For a system of n non-linear equations, $B_{(p)}$ will have n^2 components, whereas Eq (3.58) only specifies n conditions for them. As a result, there are $n^2 - n = n(n - 1)$ degrees of freedom in defining $B_{(p)}$. That is, a number of possiblities exist for defining $B_{(p)}$ fully. The construction of a successful secant approximation consists of selecting a good way to choose from among these possibilities. Hence, a whole class of methods has been developed, all of which can be considered as extensions of the secant method in one dimension. Broyden's is one such method.

To complete the definition of $B_{(p)}$, Broyden's method uses the concept of direction and least change:

(i) Eq (3.58) may only be used to approximate the Jacobian matrix in one of n directions; say in the direction of the vector

$$s_{(p)} = (x)_{(p)} - (x)_{(p-1)} \qquad (3.59)$$

To completely define the matrix, the method must also prescribe how $B_{(p)}$ is to approximate the Jacobian in the remaining (n-1) directions. It is the definition of $B_{(p)}$ in these remaining directions which characterises Broyden's method.

(ii) The essence of the method is that, in updating the Jacobian, it leaves projections along directions orthogonal (perpendicular) to $s_{(p)}$ unchanged i.e. it preserves as much as possible of the previous matrix. Johnston (1982) demonstrates how this approach

satisfies the remaining n(n - 1) conditions for $B_{(p)}$. The formula that is used for updating the matrix is as follows:

$$B_{(p)} = B_{(p-1)} + \frac{1}{s^T_{(p)} s_{(p)}} (y_{(p)} - B_{(p-1)} \cdot s_{(p)}) s^T_{(p)} \qquad (3.60)$$

where $y_{(p)} = f(x)_{(p)} - f(x)_{(p-1)}$

$s_{(p)} = (x)_{(p)} - (x)_{(p-1)}$

The second term on the right hand side of Eq (3.60) is termed the update matrix. Johnston (1982) observes that this matrix has rank one; that is, all the columns are merely multiples of one another. For this reason, Broyden's method is sometimes referred to as a rank-one update method.

Like any method for solving non-linear equations, Broyden's method involves an iterative procedure. To initiate an iteration, initial guesses of both the solution and the B matrix are required. Johnson (1982) maintains that the choice of $B_{(0)}$ is not critical to the rate of convergence and that an identity matrix can be used to seed the method. In practice, however, for the biological reaction systems analysed it was found that the choice of $B_{(0)}$ has a significant effect on whether or not the method converges. This is confirmed by Sargent (1981) who emphasises the importance of supplying a good initial approximation to the Jacobian matrix. For the biological systems simulation program, a successful approach was found to be the use of a finite differences approximation to the Jacobian as the starting B matrix.

Broyden's method is considered to be the most successful secant extension to solve systems of non-linear equations (Dennis and Schnabel, 1983). The advantage of Broyden's method over a fully generalised secant method is its saving on storage - no previous x vectors need be stored. The disadvantage is that Broyden's method has only superlinear convergence properties (Dennis and More, 1977), whereas a generalised secant method is quadratically convergent. However, although Broyden's method may not converge as rapidly as Newton's method, far fewer

function evaluations are required at each iteration. This is a major factor in favour of the method.

3.9.1 A refinement to the method

One aspect of the biological model that exerts a significant influence on the ultimate success of any solution method is the relative scales of the state variables. In the types of system under consideration, the independent variables often have vastly differing magnitudes. For example, organism concentrations are usually in the order of thousands while soluble substrate concentrations may be less than unity. This can lead to problems in convergence of a numerical method, where the contribution of the smaller variables may be outweighed. Dennis and Schnabel (1983) suggest a scaling procedure to eliminate large differences in magnitude. This procedure involves changing the units of one or more of the variables in order to bring them into the same range. Practically, this is achieved through changing the independent variables to $x^* = x \cdot D_x$ where D_x is a a diagonal scaling matrix. D_x is composed of terms representing the reciprocals of "typical magnitudes" of the variables. For example, in a two dimensional situation, if typical magnitudes of x_1 and x_2 are 10^3 and 10^6 respectively, D_x will be as follows:

$$D_x = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 10^{-6} \end{bmatrix}$$

Dennis and Schnabel (1983) note that this transformation of the units of a problem has no effect on the Newton direction for systems of non-linear equations.

3.9.2 A numerical example

Dennis and Schnabel (1983) illustrate the behaviour of Broyden's method in comparison to Newton's method with a set of two non-linear equations as follows:

$$f_1 = x_1 + x_2 - 3$$
$$f_2 = x_1^2 + x_2^2 - 9 \qquad (3.61)$$

The roots of the functions are $x^* = (3,0)$ and $x^* = (0,3)$.

For the first iteration, starting from the initial guess, $x_{(0)} = (1,5)$, the following is required:

$$B_{(0)} = \begin{bmatrix} 1 & 1 \\ 2 & 10 \end{bmatrix}$$ (which happens to be the Jacobian at $x_{(0)}$)

$$f(x)_{(0)} = \begin{bmatrix} 3 \\ 17 \end{bmatrix} \qquad s_{(0)} = -(B_{(0)})^{-1} \cdot f(x)_{(0)} = \begin{bmatrix} -1.625 \\ -1.375 \end{bmatrix}$$

$$(x)_{(1)} = (x)_{(0)} + s_{(0)} = \begin{bmatrix} -0.625 \\ 3.625 \end{bmatrix} \qquad f(x)_{(1)} = \begin{bmatrix} 0 \\ 4.531 \end{bmatrix}$$

Then, from Eq (3.60):

$$B_{(1)} = B_{(0)} + \begin{bmatrix} 0 & 0 \\ -1.625 & -1.375 \end{bmatrix} \qquad (3.62)$$

The complete sequence of iterates produced by both Broyden's and Newton's methods for this example are given in Table 3.6. For $p \geq 1$, $x_{1(p)} + x_{2(p)} = 3$ for both methods, so only $x_{2(p)}$ is listed in the table.

3.9.3 The Broyden algorithm

Step 1: Express the non-linear functions in the form $f(x) = 0$. Select initial estimates for the roots $x_{(0)}$, an initial approximation to the Jacobian matrix, $B_{(0)}$ and a suitable convergence criterion.

	Broyden's Method	Newton's Method
Iteration number	x_2	x_2
0	1	1
1	3.625	3.625
2	3.0757575757575	3.0919117647059
3	3.0127942681679	3.0026533419372
4	3.0003138243387	3.0000023459739
5	3.0000013325618	3.0000000000018
6	3.0000000001394	3.0
7	3.0	

Table 3.6 Comparison of Broyden's and Newton's methods for two non-linear equations in two unknowns (after Dennis and Schnabel, 1983)

Step 2: Solve the set of linear equations

$$B_{(p)} \cdot s_{(p)} = -f(x)_{(p)} \quad \text{for } s_{(p)}$$

Step 3: Calculate

$$x_{(p+1)} = x_{(p)} + s_{(p)}$$

$$y_{(p)} = f(x)_{(p)} - f(x)_{(p-1)}$$

Step 4: Calculate

$$B_{(p)} = B_{(p-1)} + \frac{1}{s^T_{(p)} s_{(p)}} \left(y_{(p)} - B_{(p-1)} \cdot s^T_{(p)}\right) s_{(p)} \qquad (3.60)$$

Step 5: Test for convergence

If $\Sigma \left| f_i(x)_{(p)} \right|^2 <$ convergence tolerance

then terminate the iteration.

Otherwise, replace $x_{(p)}$ by $x_{(p+1)}$ and return to Step 2

3.9.4 Considerations in the method

Broyden's method seems to be particularly suited to flowsheeting type problems and has been analysed widely in the chemical engineering literature. The major advantage of the method, and indeed, the reason for its development, is that it preserves many of the positive characteristics of Newton's method whilst only requiring roughly half the computational effort with respect to the Jacobian evaluation.

Certain potential drawbacks to the method should be noted:

(i) The convergence rate of Broyden's method is superlinear but not of the same order as Newton's method. Therefore, more iterations will be required than for Newton's method.

(ii) A good approximation to the Jacobian matrix is necessary to seed the method, otherwise it may fail to converge.

(iii) The method can behave erratically in regions where the partial derivative matrix is nearly singular.

(iv) In many flowsheeting applications, (for example, the biological system) the Jacobian matrix is sparse. In updating the approximation to the Jacobian using Broyden's method, non-zero terms (of very small magnitude) may be introduced into the approximating matrix, $B_{(p)}$, at points where the true Jacobian, $J_{(p)}$, would contain zeroes. This has certain implications because part of Broyden's method involves solving a set of linear equations incorporating $B_{(p)}$ (see Step 2 of the algorithm). Solution methods such as Gausssian elimination with pivotal rearrangement will now require more computation at this step because the matrix has become less sparse. This will partially negate the benefit of fewer function evaluations required to set up the Jacobian.

CHAPTER FOUR

STEADY STATE ANALYSIS: CASE STUDIES

4.1 INTRODUCTION

In Chapter 3, five approaches were presented for solving sets of simultaneous non-linear algebraic equations typically encountered in describing the state of a reaction system under steady state conditions. The presentation was oriented towards biological reaction systems in particular. However, details concerning the different numerical methods were of a more general nature. In this Chapter, the objective is to evaluate and compare the different methods in application to solution of a range of specific steady state biological system problems. Before evaluating the numerical techniques, two aspects should be specified. The first is that a biological model must be selected. The second is the selection of a range of reactor configurations to be considered in case studies. These configurations should incorporate the characteristics of the various types of flowsheet encountered in practice.

4.1.1 Selection of a biological model

Considerations that are involved in the selection of a biological model have been referred to in Chapters 1 and 2. The model selected for the purpose of evaluating the various numerical techniques in this study is a restricted version of the IAWPRC Task Group model for the activated sludge process. Only aerobic heterotrophic growth phenomena have been included, as shown in the model matrix of Table 2.2. The values for the kinetic and stoichiometric parameters that have been used in the simulations are in line with those selected by the IAWPRC Task Group and described by Dold and Marais (1985); Table 4.1 summarises these parameters.

Symbol	Value	Units
Kinetic parameters:		
$\hat{\mu}$	4.0	day^{-1}
K_S	5.0	g COD m^{-3}
b	0.62	day^{-1}
K_H	2.2	$\frac{\text{g COD}}{\text{g cell COD} \cdot \text{day}}$
K_X	0.15	g COD (g COD)$^{-1}$
Stoichiometric parameters:		
Y_H	0.666	$\frac{\text{g COD cell yield}}{\text{g COD utilised}}$
f	0.08	

Table 4.1 Kinetic and stoichiometric parameters used in the case studies. The biological model is presented in Table 2.2.

4.1.2 Selection of the case studies

An evaluation of the suitability of the different numerical methods needs to be grounded in the types of situation that the methods will encounter in practice. For example, in a wastewater treatment application, a numerical technique would need to handle problems stemming from a wide variety of system configurations and operating conditions. These may range from a simple single reactor process operated at short sludge age to a more complex system incorporating numerous reactors in series linked by both forward and recycle flows and operated at a long sludge age.

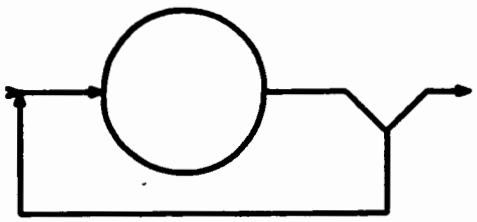
Five configurations were selected as case studies for evaluating the numerical methods. These specific configurations with associated recycles and operating conditions were chosen as they incorporate facets of a spectrum of systems encountered in biological wastewater treatment. Although specific to activated sludge systems, the configurations include certain features general to most biological reaction systems. Table 4.2 summarises the details of the system configurations and operating conditions for the five test cases. The configurations are shown diagramatically in Figure 4.1. Because a limited biological model was used for the study, no provision is made for the usual phenomena encountered with unaerated reactors e.g. denitrification. Hence, all the reactors in the test case configurations are aerated even though unaerated reactors would usually be incorporated in certain of the configurations; for example, the UCT process (Case 5). Aspects particular to the five selected configurations are as follows:

Case Study 1 : This case study is the simplest possible configuration that could be encountered in an activated sludge process. It consists of an aerated reactor and a settling tank. The underflow from the settling tank is recycled to the reactor. The configuration is the same as that introduced in Section 2.5.

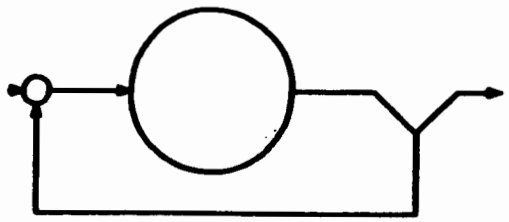
Case Study 2 .: This case study corresponds to a "selector reactor" configuration utilised in the control of sludge bulking. It consists of

		Case 1	Case 2	Case 3	Case 4	Case 5
Configuration		Single Reactor	Selector Reactor	Contact Stabili-sation	Series Reactors	UCT Process
REACTOR VOLUMES (litres)	Reactor 1	8	0.25	12	1.5	2
	Reactor 2		8	2	1.5	3
	Reactor 3				1.5	6
	Reactor 4				1.5	
	Reactor 5				1.5	
SLUDGE AGE (days)		3	3	6	5	20
FEED RATE (l/day)		20	20	36	20	10
RAS RECYCLE RATE (l/day)		20	20	72	20	10
A RECYCLE	From Reactor To Reactor Rate (l/day)					3 2 40
B RECYCLE	From Reactor To Reactor Rate (l/day)					2 1 10
INFLUENT COD	500 $g.m^{-3}$ [S_S = 100 $gCOD.m^{-3}$; X_S = 400 $gCOD.m^{-3}$]					

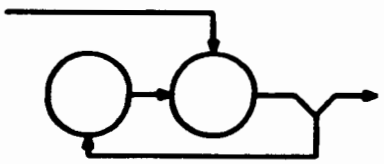
Table 4.2 Summary of system configurations and operating conditions for the Case Studies.



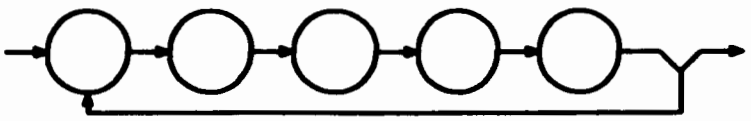
CASE STUDY 1



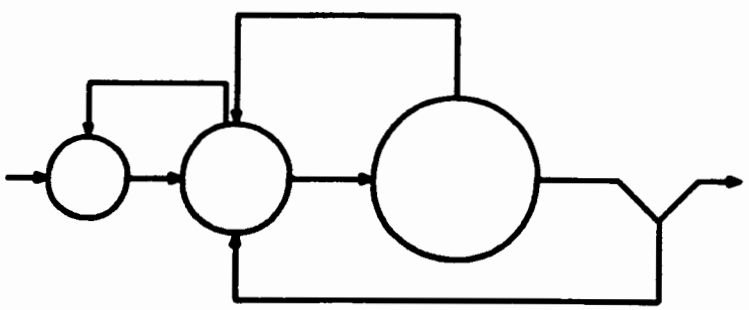
CASE STUDY 2



CASE STUDY 3



CASE STUDY 4



CASE STUDY 5

Figure 4.1 The Case Studies.

two aerobic reactors in series, the first reactor being very much smaller than the second (volume ratio 1:32). All the feed enters the first reactor, as does the underflow from the settling tank.

Case Study 3 : Case Study 3 corresponds to a "contact stabilisation" process, in which all the feed enters the "contact" reactor, which is the second of two aerobic reactors in series. The underflow from the settling tank is recycled to the first reactor.

Case Study 4 : This case study consists of five aerated reactors in series, with all the feed entering the first reactor. Underflow from the settling tank is recycled to the first reactor in the series.

Case Study 5 : This configuration corresponds to a "UCT process" with three reactors in series. The distinguishing feature of the configuration is the arrangement of recycles between reactors. Mixed liquor recycles are taken from the third to the second and from the second to the first reactors. Underflow from the settling tank is recycled to the second reactor in the series. All the feed enters the first reactor.

4.2 CRITERIA FOR EVALUATING NUMERICAL METHODS

The general characteristics, advantages and disadvantages of five selected numerical methods have been outlined in Chapter 3. In attempting to select a numerical method appropriate to a particular application, the main criteria that need to be satisfied are:

(i) The method must offer a reasonable guarantee of convergence to a solution from the specified initial values.

(ii) It should converge as "efficiently" as possible.

The "efficiency" of a method is a measure of how much computational effort is required to calculate a reasonable approximation to a solution. Two aspects need to be considered here:

(i) the number of iterations required before a method converges;

(ii) the amount of computation required to perform each iteration.

In general, when comparing numerical methods, it has been found that those that are superior with respect to guarantee of convergence will usually be slow. Conversely, the faster numerical methods are more likely to diverge (Johnston, 1982). Consequently, in choosing a numerical method, a decision has to be made as to which qualities are more important at any one time. For example, in situations where the location of the solution is completely unknown, a slow method, but one which is unlikely to diverge despite crude initial estimates, will be preferred.

4.3 IMPLEMENTATION OF THE NUMERICAL METHODS

A computer program was written to test the different numerical techniques. The simulation program was written in Turbo Pascal (Borland, 1983), a language which was found to be suitable for use with an IBM PC or compatible machine. The program was specific to the selected biological model but allowed flexibility in the choice of system configuration and operating conditions. Each numerical technique was written as a module, which was then inserted in its entirety into the simulation program. This was done in an effort to eliminate any bias that might be introduced by different programming codes affecting the relative efficiencies of any of the methods. That is, computer code for setting up and evaluating reaction rates, reactor input and output terms, etc., was common to all the methods.

Three features incorporated in the computer program, and thus common to evaluation of each numerical technique, should be noted. These are:

(i) Calculation of sludge wastage rate in accordance with a specified sludge age;
(ii) the initial estimates of the solution; and
(iii) the convergence criterion.

4.3.1 Calculation of the wastage rate, q_w

In setting up the simulation problem, sludge age (solids retention time, SRT) is specified as an operating parameter. This is defined as:

$$\text{Sludge age} = \frac{\text{Mass of sludge in the system}}{\text{Mass of sludge wasted per day}}$$

$$= \frac{\text{Mass of sludge in the system}}{q_w \cdot C_n} \qquad (4.1)$$

where C_n = concentration of solids in the n^{th} reactor

In this study, it is assumed that sludge wastage always comes from the last reactor (n^{th}) in the series i.e. hydraulic control of sludge age. If all the feed enters the first reactor and the settling tank underflow is recycled to the first reactor, then the concentration of sludge from reactor to reactor is more or less constant. In this case, the required sludge wastage rate, q_w, to maintain a specified sludge age is given by:

$$q_w = \frac{\text{Total volume of system}}{\text{Sludge age}} \qquad (4.2)$$

When specifying sludge age as an operating parameter, a problem in specifying the wastage rate occurs where the concentration of sludge varies from reactor to reactor. This will be encountered when the feed enters, for example, the second reactor in the contact stabilisation process (Case 3) or where the settler underflow is not recycled to the first reactor as in the UCT process (Case 5). The problem arises because the wastage rate can only be determined once the distribution of sludge between the reactors and particularly the concentration in the last reactor is known. However, this concentration is influenced by the wastage rate itself. To overcome this problem, the following iterative procedure was employed once the reactor configuration and feed and recycle rates had been specified:

Step 1 : Assume that a particulate inert tracer is introduced into the influent at some constant concentration. This fixes the mass of inert tracer in the system for a given sludge age.

Mass of tracer = Daily inflow * concentration of tracer in the influent * sludge age

Step 2 : Provide an initial estimate of the wastage rate from Eq (4.2).

Step 3 : For the selected q_w, solve the set of mass balance equations describing the concentration of tracer in each reactor and in the underflow recycle

Step 4 : Recalculate the wastage rate from Eq (4.1).

Step 5 : Test for convergence.

If convergence is achieved, then terminate the iteration.

Otherwise, return to Step 3.

4.3.2 Initial estimates of the solution

To initiate any of the iterative numerical procedures, an estimate of the solution is required. If these estimates are not accurate, it is possible that the numerical method will not converge to the correct solution. Also, the less accurate the initial estimate, the greater the number of iterations that will be required to attain convergence.

In the computer program, initial estimates of the state variables are based on steady state wastewater treatment theory (WRC, 1984) and on empirical estimates. The simulation program estimates the masses of the active organism (X_B) and endogenous residue (X_E) fractions from this theory, based on the effective steady state endogenous respiration rate. The masses of these particulate materials, biomass, X_B, and endogenous residue, X_E, are distributed amongst the reactors in accordance with the distribution of the inert particulate tracer as discussed in Section 4.3.1 above. The initial concentration of particulate substrate, X_S, in each reactor is assumed to always be ten percent of X_B, and the initial estimate of the soluble substrate, S_S, is always taken as 1.5 g COD m^{-3}.

4.3.3 Convergence criteria

The solution to the steady state problem is reached when the set of mass balance equations, $f(x) = 0$ is satisfied. In converging to the solution, a measure of the accuracy of the current values at each iteration is given by the magnitude of the functions. To have some global measure which will embrace all the state variables, the convergence criterion was formulated in terms of

$$\Sigma[f_i(x)]^2 \qquad (4.3)$$

It was assumed that a solution had been reached when this summation was less than a certain error tolerance. In choosing the magnitude of this tolerance, a balance between efficiency and reliability should be maintained. The convergence tolerance must be reasonably small in order to prevent early termination. Choosing too small a value, however, can delay termination unnecessarily. The selection of 10^{-3} as the convergence tolerance was found through practice to result in accurate solutions. At the same time, it is not so stringent that the numerical methods take unacceptably long to satisfy it.

4.4 CASE STUDY RESULTS AND DISCUSSION

The five numerical methods discussed in Chapter 3 were applied to each test case. Each method was allowed to run until convergence was achieved, and subsequently assessed in terms of:

(i) how long it took to reach an acceptable solution from a standard set of starting values; and

(ii) how many iterations were required for the given convergence criterion. [1]

---

[1] All the results were obtained using Turbo Pascal Version 3.0 running on a standard IBM PC operating at 4.77 MHz. The configuration did not include an 8087 maths co-processor.

The results for each method and test case are presented in Table 4.3. Certain overall aspects are apparent from the results. These are discussed in Section 4.4.1. In addition, a more detailed comparison of some of the numerical methods was carried out to assess the actual manner in which different techniques approached the solution. For certain of the techniques, this evaluation involved examination of potential instability problems. For others, an assessment was made regarding exactly how much computational energy was expended at each point in an iteration loop. This was in order to develop more of an understanding of the behaviour of each method in its practical implementation, and to establish a qualitative feel for more than just the convergence properties of a particular technique. The more detailed comparison of methods is discussed in Section 4.4.2 and Section 4.4.3.

4.4.1 General comments

(i) All the methods converged to the same solution for all the test cases. However, it should be remembered that for the direct linearisation approach, successive substitution and Wegstein's method, the set of equations had to be arranged in particular ways in order for convergence to be attained. Some forms of re-arrangement of the equations did not converge from the specified initial conditions.

(ii) The test case results bear out a generally expected trend of convergence characteristics. The successive substitution and Wegstein methods, exhibiting only linear convergence rates, needed significantly more iterations in order to converge to a solution. Newton's method, with a quadratic rate of convergence, requires very few iterations to attain convergence. Broyden's method, which has a convergence rate that is superlinear, although not quadratic, required approximately twice as many iterations to converge as did Newton's method.

(iii) For all the case studies, Newton's method was always the fastest to converge. Despite the fact that each iteration in this method

	METHOD									
	Direct Linearisation		Successive Substitution		Wegstein's Method		Newton's Method		Broyden's Method	
	Its.	Time	Its.	Time	Its.	Time	Its.	Time	Its.	Time
CASE 1 Single Reactor	16	12.9	108	41.8	133	54.1	4	5.5	5	7.0
CASE 2 Selector Reactor	16	24.8	256	143.5	258	175.5	4	12.2	10	32.0
CASE 3 Contact Stabilisation	12	18.3	619	347.0	576	391.4	4	12.2	8	25.5
CASE 4 Five - in - Series	14	62.6	1663	2945.2	1605	3004.9	3	36.4	7	49.6
CASE 5 UCT Process	10	25.1	606	501.5	615	605.4	4	24.1	8	49.2
where Its. = number of iterations Time = time in seconds										

Table 4.3 Test case results.

involves a complete re-computation of the Jacobian matrix, the computational time expended per iteration is not excessive. In addition, the case studies verify the advantage of the quadratic convergence rate, as Newton's method requires significantly fewer iterations than any of the other methods to reach a solution. This seems to be irrespective of the complexity of the configurations, as the method consistently required only three or four iterations to converge.

(iv) Broyden's method generally required approximately twice the number of iterations as Newton's method. This is in agreement with the general convergence characteristics of quasi-Newton methods i.e. those using an approximation to the Jacobian matrix. However, the time taken to reach convergence by the two methods should then be approximately equal, given that Broyden's method requires only half the number of function evaluations to estimate the Jacobian. Examination of Table 4.3 shows that, in practice, this does not occur. In fact, Broyden's method consistently required longer than Newton's method to converge. This aspect is discussed in more detail in Section 4.4.3.

(v) Both the methods of Wegstein and successive substitution were found to perform consistently poorly for all the test cases. This was not entirely unexpected. The fact that both are simple to implement and require very little computational effort per iteration is counterbalanced by inferior rates of convergence.

(vi) Successive substitution and Wegstein's method may appear to perform disproportionately poorly for the five-in-series reactor configuration of Case 4. On consideration, however, this result is to be expected. Both these methods involve fixed point iteration in which each state variable is modified without regard for the simultaneous changes in other variables at each iteration. In contrast, Newton's method, for example, accounts for this "simultaneity" via the partial derivatives in the Jacobian. Therefore, when successive substitution or Wegstein's

method is applied to a long train of reactors with no internal recycles such as Case 4, inaccuracies in the initial estimates "work through" the system slowly. The performance of these methods is improved relatively when internal recycles are included in the configuration as in Case 5 for example. These recycle links in effect partially account for the interaction between the state variables which is not directly considered with successive substitution or Wegstein's method.

To explain this, consider a certain compound in a two reactor system where the respective concentrations are denoted by x_1 and x_2. If there is no recycle from reactor 2 to reactor 1, then the mass balance equation for x_1 does not contain the variable x_2. As a result, in the fixed point iteration step for x_1 the influence of the variable x_2 is disregarded. In contrast, if there is a recycle from reactor 2 to 1, then the variable x_2 is incorporated in the mass balance equation for x_1. In this case, cognisance is given to x_2 when iterating for x_1.

(vii) The performance of Wegstein's acceleration method compared to that of successive substitution was surprisingly poor. The lack of improvement over successive substitution indicates that Wegstein's method is not an appropriate acceleration technique for these types of functions. A more detailed examination of the relative merits of successive substituion and Wegstein's method is presented in Section 4.4.2.

(viii) The direct linearisation method produced very favourable results for all the test cases. Accurate solutions were achieved, and convergence was both rapid and efficient. In fact, for Case 4, its performance is almost comparable to that of Newton's method. The efficiency of the method also seems to be relatively independent of the complexity of the system configuration and operating conditions. In fact, fewer iterations and computational effort were required to reach a solution in Case 5 - the most complex configuration - than in Case 1 - the simplest case study. The reason for the success of the direct linearisation

method is that the functions for the biological system under consideration are not particularly non-linear in the regions of interest, and thus the linearised functions give a good approximation to the non-linear equations. However, as noted earlier, a severe drawback of the method is the prior skill and mathematical manipulation that are necessary before the method can be implemented.

4.4.2 Comparison of the Wegstein and successive substitution methods

These two numerical methods both reached convergence for all situations, although in Case 4 many iterations were required before the tolerance was eventually satisfied. The amount of computational time expended per iteration for both techniques is near equal, although Wegstein's method generally takes slightly longer than successive substitution for each loop. This is to be expected, as the methods are identical except for the relatively inexpensive additional calculation of acceleration factors and checks on these that are introduced with Wegstein's method.

The number of iterations required by each method in order to attain convergence was found to be near equal, although Wegstein's method consistently required a few more iterations than did successive substitution. This is contrary to what was expected, as Wegstein's method was originally implemented to accelerate the rate of convergence of successive substitution. This result demanded further investigation.

To examine the phenomenon more fully, various modifications of Case Study 1 were considered. In one of these modifications, the sludge age was changed to 30 days instead of 3 days (all other parameters were maintained as before) and both methods were re-tested. Figure 4.2 shows the trend observed in the sum of the squares of the function values, which was used as the stopping criterion for both methods. As expected, Wegstein's acceleration method moves to the region of solution more rapidly than does successive substitution. What is surprising, however, is that ultimately Wegstein's method requires more iterations to reduce

the error to within the specified tolerance. On closer examinaiton it was apparent that the reason for this behaviour was a slight instability introduced by Wegstein's method. This is not readily noticable in the plot of Fig 4.2.

Figure 4.3 shows the path followed by the concentration of particulate biomass, X_B, in approaching the solution for Case Study 1. Again, with Wegstein's method, the value of X_B initially converges more rapidly to the solution, with less overshoot, as would be expected. However, although the general trend introduced by the acceleration is towards a more "damped" path, the individual points on the curve have more of a tendency to oscillate than those generated by the successive substitution technique. When the solution is approached, this instability prevents the convergence criterion from being satisfied.

It appears from the results that there would perhaps be some merit in using the approach suggested by Westerberg et al (1979); that is, applying Wegstein's method at intervals. This would presumably accelerate the successive substitution whilst avoiding the instabilities associated with Wegstein's method.

4.4.3 Comparison of Broyden's and Newton's methods

The relative convergence rates of these two methods bear out the expected trends: Broyden's method does not converge as rapidly or as efficiently as Newton's method. However, the fact that Broyden's method is so computationally expensive merits further investigation.

Case Study 2 was used to examine the details of how the computational energy for each iteration in the methods was distributed. Table 4.4 shows this "division of effort" for the second, third and fourth iterations. Both techniques took approximately three seconds to complete each iteration. The major components are: (i) the time required to set up the Jacobian (or its approximation) and (ii) the time required to solve the resulting set of linear equations.

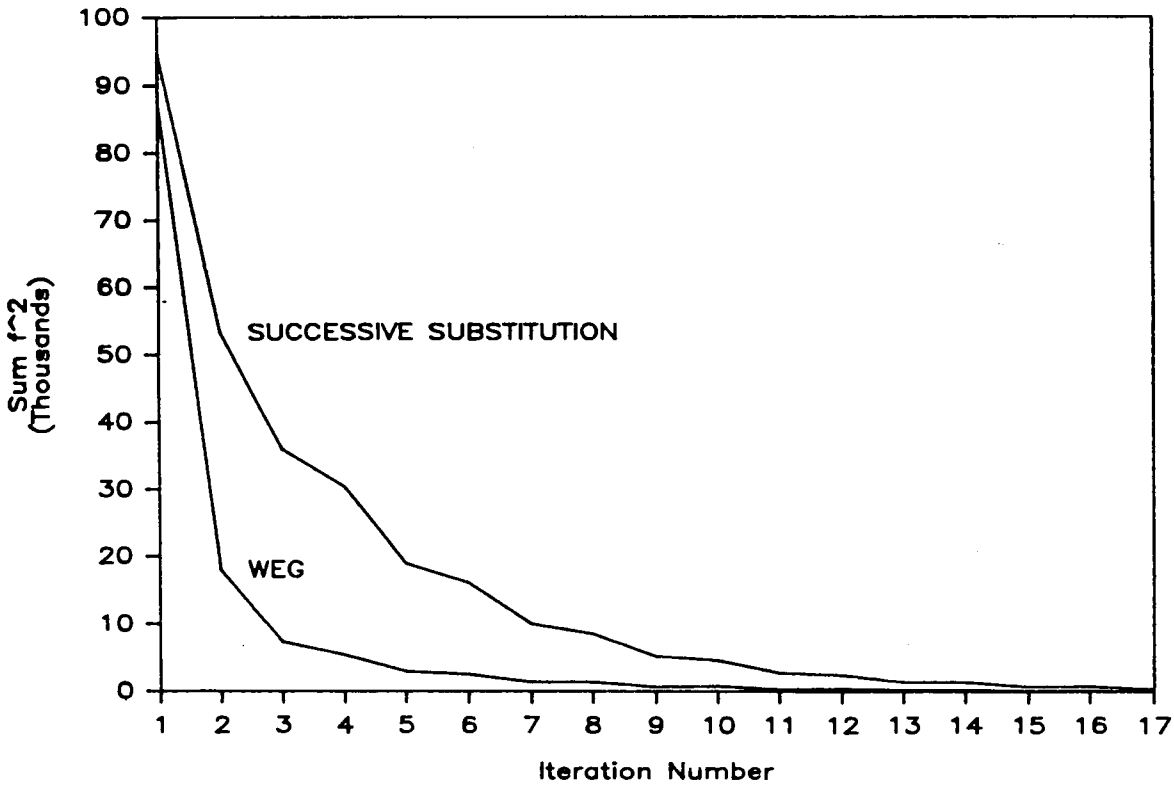


Figure 4.2 Comparison of Wegstein and successive substitution methods for Case Study 1 (Sludge age = 30 days)

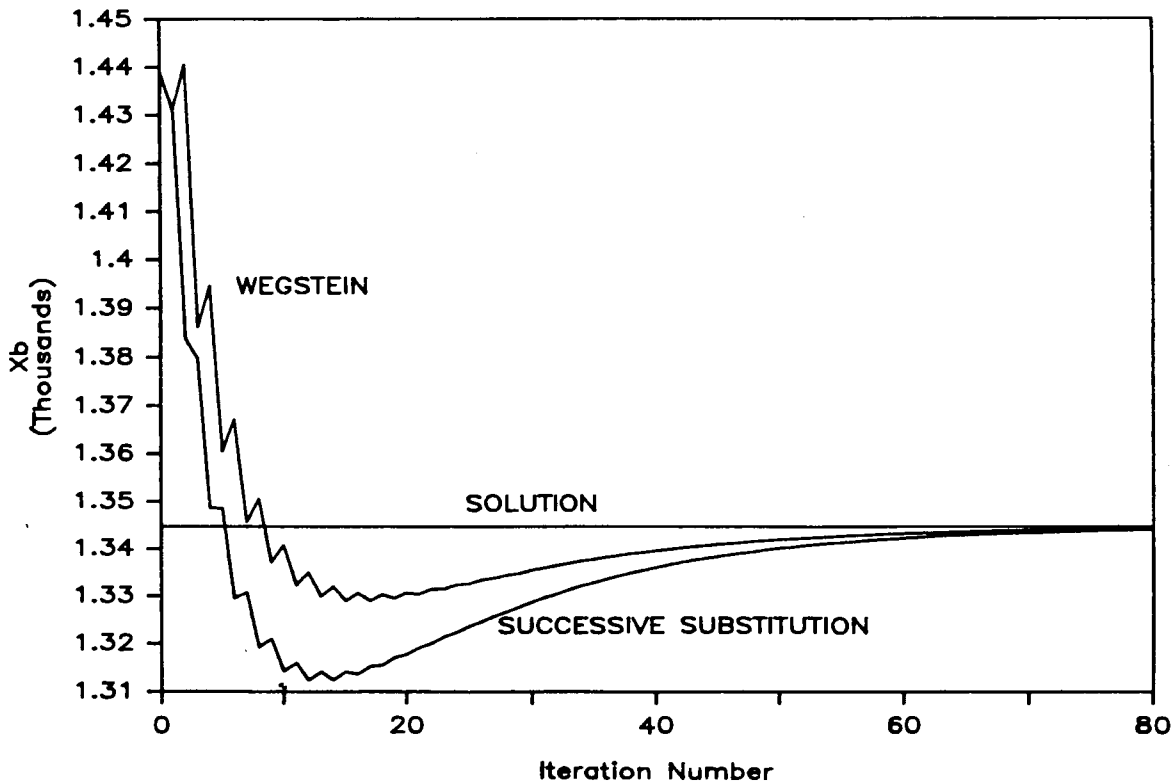


Figure 4.3 Comparison of X_B values for Wegstein and successive substitution methods for Case Study 1.

CASE STUDY 2		Time (sec)	
Iteration Number		Broyden's Method	Newton's Method
2	Gauss Matrix	1.71 0.98	0.72 2.04
3	Gauss Matrix	1.76 0.99	0.71 1.98
4	Gauss Matrix	1.70 0.98	0.72 2.03
where Gauss = time spent solving linear equations Matrix = time spent updating the matrix			

Table 4.4 Comparison of time per iteration as expended by Broyden's and Newton's methods for iterations 2 to 4 in Case Study 2.

(i) Previous discussion has indicated that the major advantage of Broyden's method is that, to set up the Jacobian approximation, it requires fewer function evaluations at each iteration than Newton's method, and thus should require less time per iteration. An examination of the results in Table 4.4 shows that the time spent by Broyden's method in updating the approximation to the Jacobian is roughly half that spent by Newton's method in re-evaluating the complete matrix of partial derivatives (0.98 secs versus 2.02 secs). This is to be expected as half the number of function evaluations are required when using Broyden's method.

(ii) The major expense in Broyden's method is the disproportionate time spent in solving the resulting system of linear equations by the Gauss elimination procedure used here. In Broyden's method, the Gauss elimination takes more than twice as long to implement as it does in Newton's method (1.72 secs versus 0.72 secs). The reason for this is that small non-zero terms are introduced into the matrix by Broyden's updating formula in locations that would usually contain zeroes in the Jacobian. This effectively reduces the sparsity of the Broyden matrix and severely hampers the operation of the Gaussian technique, which relies on pivotal rearrangement for its efficiency.

From the results above, it is apparent that, if the saving in the number of function evaluations in Broyden's method is to be exploited, then attention should be paid to the method used to solve the linear equations. Perhaps this could be improved by using some specialised matrix technique, rather than the Gaussian elimination used here.

4.5 GENERAL CONCLUSIONS

(i) The method of direct linearisation, although performing relatively efficiently for these test cases, is not a suitable technique for general use in a simulation program. Thus, although the preliminary analysis seems to have paid off in the satisfactory performance of the method, the requirements of the

program that it be as generally applicable as possible eliminates direct linearisation from the possibilities that can seriously be considered.

(ii) The methods of Wegstein and successive substitution, although simple and robust, are inappropriate due to their slow rates of convergence. Instability problems may be encountered in their implementation and, as a result, convergence cannot always be guaranteed.

(iii) The poor performance of Wegstein's method in comparison to that of successive substitution could perhaps be eliminated by applying Wegstein only at selected intervals, as suggested by Westerberg et al (1979).

(iv) Broyden's method converges to a solution in comparatively few iterations. The computational effort required to set up the Jacobian approximation is roughly half that required by Newton's method in setting up the true Jacobian. However, the method as implemented here requires an excessive amount of computational effort per iteration. The major portion of this effort is concentrated in the solution of the system of linear equations. Perhaps this bottleneck could be removed by employing a specialised sparse matrix technique.

(v) Of all the methods evaluated, Newton's appears to be the most favourable. In addition, the use of a finite difference approximation to the Jacobian matrix renders it a generally suitable technique for the biological flowsheeting systems under consideration.

CHAPTER FIVE

MODELLING OF THE DYNAMIC CASE

5.1 INTRODUCTION

In practice, the inputs to a biological system are unlikely to remain constant. Because the influent to the system varies with time, the mass balance equations describing the response of the system will take the form of a set of differential equations incorporating time-dependent terms [see Eqs (2.14) to (2.21), for example]. This set of equations will define how the values of the concentrations of each compound in each reactor (the state variables) vary with time.

Solving the set of simultaneous differential equations is an initial value problem. The magnitudes of the concentrations of each compound in each reactor are specified as the initial condition, and thereafter the equations are solved by integrating forward in time. In this way, the changes in concentration in each reactor can be tracked, subject to the variations in the influent flow rate and concentrations. In certain circumstances, such as an activated sludge system, the influent pattern of flow rate and concentration is repeated closely from day to day i.e. a daily cyclic basis. A useful facility, therefore, is to predict the steady state cyclic response when it is assumed that the influent pattern is repeated identically from day to day. To find this solution will require integrating forward through perhaps many cycles until convergence to the solution is attained. Convergence in this case requires that the cyclic concentration response of each compound in each reactor is identical from cycle to cycle, and the values at the start and end of each cycle are the same.

The set of differential equations describing the response of a biological system under dynamic conditions will contain non-linear terms, as did the mass balances for the steady state case. The task of finding the solution to a such a set of non-linear ordinary differential equations is certainly not unique to biological systems. Many systems of

interest to engineers and scientists are described by non-linear differential equations. A multitude of numerical integration techniques exist for the solution of these sets of equations. Consequently, when faced with such a set of equations, the problem in finding a numerical method is the selection of an appropriate one from the many diverse methods available.

In the selection of an integration technique for the dynamic problem, the approach taken was to initially establish a rudimentary integration module which could be gradually refined and improved. In the process of refining the module, a greater understanding of the actual dynamics of the system was generated. This in turn indicated further adjustments that could be made to the routine to improve it still further. Thus, through an interactive process, the integration routine was gradually tailored to better meet the demands of the biological system under consideration. In the interests of clarity, this presentation will follow the manner in which the actual integration module was developed and incorporated in a dynamic simulation computer program. Each of the series of refinements will be presented and discussed in the same sequence as incorporated into the program.

5.2 USING NUMERICAL INTEGRATION TECHNIQUES

Because the exact solution to the set of differential equations is not, in general, known and cannot be calculated analytically a numerical integration technique will be required to provide an approximation to the solution. One common approach, which will be the focus of this presentation, is to use a time-stepping or difference method which approximates the solution by its value at a sequence of discrete points called the mesh points. Given a differential equation $y'(x) = 0$, a difference method provides some rule for approximating y at a point x_n ($y(x_n)$) in terms of the value of y at x_{n-1} and possibly at preceding points. Ideally, the solution should be represented by its actual value at each mesh point so that it can be approximated to high accuracy by interpolation between the mesh points. However, the exact solution to the differential equation is not known, so it is always an approximation

that is sought. Many techniques assume that the mesh points are equally spaced. However, since the step size seems to have an effect on the error introduced, it is usually possible to vary the mesh spacing to account for this. For the moment, it will be assumed that the mesh spacing remains constant during the stepping procedure. However, variable step size integration procedures will be examined in more detail later.

5.2.1 A simple Euler method

The simplest stepping technique available is Euler's rule. This was used as the first attempt to solve the set of differential equations generated by the biological model. In the Euler method, the value of the dependent variable at one point is calculated by straight line extrapolation from the previous point. Generally referred to as a one-step method, Euler's rule is an algorithm which prescribes the numerical technique for calculating the approximation to the solution at x_{n+1} in terms of the value at one previous step, x_n. Consider the function y with

$$y'(x) = \frac{dy}{dx} = f(x,y) \qquad (5.1)$$

The value of y at $x_{n+1} = (x_n + h)$ may be approximated by a Taylor's expansion. Truncating after the first two terms in the series yields:

$$y(x_n+h) \approx y(x_n) + h \cdot f(x_n, y(x_n)) \qquad (5.2)$$

where h = steplength

The error in this approximation is described by the remaining terms in the Taylor's expansion:

$$\frac{h^2}{2!} \cdot y''(x_n) + \frac{h^3}{3!} \cdot y'''(x_n) + \ldots. \qquad (5.3)$$

and is called the local truncation error. A more detailed discussion about the errors introduced by a stepping method, and the resultant implications will be covered in Section 5.3.

Euler's rule is usually formulated as:

$$y_{n+1} = y_n + h \cdot f_n \qquad (5.4)$$

and in this form can be described as an explicit linear one step method of first order.

5.2.1.1 An illustrative example

Dahlquist and Bjorck (1974) provide an example to illustrate the use of Euler's formula for a single differential equation:

$$\frac{dy}{dx} = y \quad \text{with } y(0) = 1$$

Euler's rule gives the following:

$$y_{n+1} = y_n + h \cdot y_n \quad y_0 = 1 \qquad (5.5)$$

Table 5.1 presents the results obtained by first computing the solution with h = 0.2 and then with h = 0.1, and compares these with the exact solution. An examination of the Table reveals that the error is approximately proportional to the stepsize. In other words, if the error in the integration is to be halved, then the stepsize will also need to be halved. This implies that, to attain reasonable accuracy with Euler's method, the stepsize chosen needs to be small. This is an inherent weakness in using a first order method.

5.2.2 Multistep methods and predictor-corrector pairs

Multistep methods present a distinct advantage over one-step methods such as the first order Euler's rule. These methods exhibit improved

Exact		STEPSIZE					
Solution		h = 0.2			h = 0.1		
x_n	$y(x_n)$	y_n	$h \cdot f_n$	error	y_n	$h \cdot f_n$	error
0.0	1.000	1.000	0.200	0.000	1.000	0.100	0.000
0.1	1.105				1.100	1.110	-0.005
0.2	1.221	1.200	0.240	-0.021	1.210	0.121	-0.011
0.3	1.350				1.331	0.133	-0.019
0.4	1.492	1.440	0.288	-0.052	1.464	0.146	-0.028
0.5	1.649				1.610	0.161	-0.039
0.6	1.822	1.728		-0.094	1.771		-0.051

Table 5.1 Euler's method executed with two different stepsizes (Dahlquist and Bjorck, 1974)

accuracy and convergence characteristics, although at the expense of requiring additional computation. Recall that Euler's method only required the value at one mesh point to compute the value at the next. Multi-step methods use more than one value of the dependent variable to calculate the equivalent information at the next time interval. Recall also that Euler's method was referred to as explicit; that is, y_{n+1} occurs only on the left hand side of the equation and can be calculated directly from the right hand side values. Linear multistep methods are generally implicit; that is, the unknown value occurs on both sides of the equation and cannot be calculated directly. These implicit methods in general entail a substantially greater computational effort than do explicit methods. On the other hand, implicit methods can be made more accurate than explicit methods and enjoy more favourable stability properties (Lambert, 1974). In fact, these considerations so favour implicit methods that explicit linear multistep methods are seldom used on their own.

The following formula, the second order trapezoidal method,

$$y_{n+1} - y_n = \frac{h}{2}\cdot[f(x_n, y_n) + f(x_{n+1}, y_{n+1})] \qquad (5.6)$$

is an example of an implicit method, since y_{n+1}, which is to be computed, appears implicitly on the right hand side. If f is a non-linear function, a non-linear system will need to be solved at each step. This must be done by some iterative method, for example, by the procedure:

$$y_{n+1} = \frac{h}{2}\cdot[f(x_n, y_n) + f(x_{n+1}, y_{n+1})] + y_n \qquad (5.7)$$

To solve Eq (5.7), a reasonable initial estimate of y_{n+1} can be obtained using past values of y with, for example, Euler's explicit formula. In this context, the explicit formula is usually referred to as the predictor, whilst the implicit formula of Eq (5.6) is referred to as the corrector. Used in combination, these two equations make up a procedure called a predictor-corrector method. Lapidus (1971) refers to a predictor-corrector method that is used in this way as a PECE method, indicating that a predicted value of y_{n+1} is followed by a derivative evaluation , y'_{n+1}, and then y_{n+1} is corrected and y'_{n+1} evaluated.

Termination of the integration step may be controlled in one of two ways. The first consists of continuing the iterative scheme suggested by Eq (5.7) until the iterates have converged. In practice, this would usually involve comparing the difference between two successive estimates of the solution to some preset tolerance. If the difference is smaller than this tolerance, then the latest value of y obtained may be regarded as an acceptable approximation to the exact solution of the equation. Since each iteration corresponds to one application of the corrector, this mode of operation of the predictor-corrector method is referred to as correcting to convergence (Lambert, 1974). In this mode, there is no way of telling in advance how many iterations will be necessary and consequently how many function evaluations will be required at each step.

The second approach for terminating the integration step is motivated by the desire to restrict the number of function evaluations per step. The number of times, m, that the corrector is applied at each step is stipulated in advance. This approach is more common than the method of correcting to convergence. This mode of operation can be described as a PE(CE)^m method, where the predicted y_{n+1} and evaluated f_{n+1} is followed by m corrections and derivative evaluations.

Of the two approaches, Lapidus (1971) recommends the PE(CE)^m method with $m = 1$ as being one of the most successful means to apply these predictor-corrector formulae. Using only a single application of the corrector formula saves on the number of function evaluations required - only two function evaluations are required per iteration step.

5.3 ERROR CONTROL

Once a time-stepping method has been selected to carry out the numerical integration procedure, the next stage is to evaluate the accuracy of the solutions that it generates. For each step of the difference procedure, some form of approximation is used to obtain the next estimate of the solution. Thus, each step taken will generate an associated error term. This is a natural consequence of any approximation technique. Given that this error can never be entirely eliminated, the best approach is to ensure that it is continuously evaluated and maintained at acceptable levels. In addition, the stepping procedure should be able to incorporate adjustments to the relevant parameters as soon as the error begins to accumulate. It is the nature of this cumulative error that will be decisive in the eventual success or failure of each step of the integration method.

5.3.1 Sources of error

In using a stepping or difference method to find the solution to a differential equation, the solution that is eventually found will never be exact. The difference between this solution and the exact solution is

the local error. Sources of error will include, amongst others, (Dahlquist and Bjorck, 1974):

(i) The round-off error introduced by using finite precision numbers.

(ii) The truncation error associated with the linear multistep method used. This is the error occurring when a limiting process is truncated or broken off before the limiting value has been reached.

Once the primary sources of error in a stepping method have been identified, it is necessary to be able to use this information in such a way as to ensure that all errors are minimised as far as possible. This will involve an assessment of how each of the error terms affect the reliability of the method, which sources of error dominate and how a knowledge of the error can be used in maintaining the accuracy of the stepping algorithm.

5.3.2 Estimating the local error

Dahlquist and Bjorck (1974) demonstrate that, for an integration method, of order p the local error is approximately bounded by:

$$l_n \approx \left| c_n \cdot \left[\frac{h}{2} \right]^{(p+1)} \right| \qquad (5.8)$$

where l_n = local error
p = order of the integration method
h = stepsize
c_n = a constant specific to the integration method

Dahlquist and Bjorck (1974) provide an alternative formulation for a predictor-corrector method. They propose that the error of the predicted value can be expressed by a difference function using a constant, c', which is specific to the order of the predictor. The difference between

the predicted and the corrected values, multiplied by c/(c'-c) is then an estimate of the local error of the corrected value:

$$l_n = \frac{c}{(c' - c)}.(y^p - y^c) \qquad (5.9)$$

where c = a constant specific to the order of the predictor
c' = a constant specific to the order of the corrector
y^p = predicted value of y
y^c = corrected value of y

5.3.3 Percentage accuracy

An error tolerance must be selected to satisfy the dual requirements of reliability and efficiency. If a very strict tolerance is chosen, unnecessary computational effort will be expended in order to meet its requirements. If the tolerance chosen is not sufficiently stringent, it is possible that the effect of a cumulative error will eventually lead to instability of the method and jeopardise its chances of successful convergence. In practise, it has been found convenient to express this tolerance in terms of a "percentage accuracy" where this percentage accuracy is defined in the same way as a relative error measurement. That is:

$$\% \text{ acc} = \frac{y_{(p-1)} - y_{(p)}}{y_{(p-1)}} \cdot 100 \qquad (5.10)$$

where $y_{(p)}$ = current estimate of the value of the variable
$y_{(p-1)}$ = previous estimate of the value of the variable
% acc = percentage accuracy

Defining the accuracy requirements in this way enables calculation of error tolerances that are independent of the absolute magnitude of the variables involved. Consequently, the accuracy specifications can be transformed into numerical language that is equally significant for variables with very small or very large magnitudes. This is an important consideration, especially for badly scaled problems, such as those encountered in biological systems. Practically, this is achieved with

the use of an error tolerance, ϵ, which is defined in terms of the percentage accuracy required and is the limiting value that the local error may reach without jeopardising the success of the step. Once a percentage accuracy is specified, this must be transformed into an ϵ value which applies to each variable. This can be achieved by firstly reformulating Eq 5.10 to give:

$$\% \text{ acc} = \frac{y^P - y^C}{y^P} \cdot 100 \qquad (5.11)$$

Given that, in the limiting situation,

$$\epsilon = l_n \qquad (5.12)$$

a limiting value for ϵ for each variable in a set of simultaneous differential equations may be derived by combining Eqs. (5.9), (5.11) and (5.12) as follows:

$$\begin{aligned} \epsilon &= \frac{c}{(c' - c)} \cdot (y^P - y^C) \\ &= \frac{c}{(c' - c)} \cdot \frac{\% \text{ acc} \cdot y^P}{100} \end{aligned} \qquad (5.13)$$

Equation (5.13) now provides a means of selecting an error tolerance for each variable which is based on a percentage accuracy requirement but which also incorporates a measure of the scale of the variable involved, y^P. Equation (5.13) is useful because it allows the user to specify a completely general percentage accuracy requirement for the integration module. This specification is then used to calculate a separate ϵ value for each component in the sytem. This error tolerance can now be used as the basis for estimating the next stepsize for each component.

5.4 STEPSIZE SELECTION

For any efficient difference method for integration, an objective is to use integration steps that are as large as possible, whilst preserving the required accuracy. Once an estimate has been made of the magnitude

of the error generated at an integration step, this estimate can be used to decide whether or not the most recently computed value of the variable is acceptable. If the error is found to be larger than a predetermined tolerance, the value will be rejected and then recomputed using a smaller stepsize. If the error is within the bounds prescribed, then the value will be accepted and a larger step can be taken in order to generate a new estimate.

Ideally, the size of each new steplength should be selected so that it reflects the magnitude of the error in the previous calculation. In other words, if the value falls well within the prescribed error bounds, a large increase in the steplength should be permitted. If the error in the value is close to the tolerance, then the subsequent steplength should be allowed to increase, but not so dramatically. It is suggested by Dahlquist and Bjorck (1974) that, in order to maintain the local error below a given tolerance, the new stepsize should satisfy the following condition:

$$c_n \cdot \left[\frac{h'}{2}\right]^{(p+1)} \leq \theta \cdot \epsilon \qquad (5.14)$$

where h' = new stepsize

θ = a preset safety factor to account for the fact that the error estimates are approximate and based on experience from the preceeding interval ($\theta \leq 1$)

From Eq (5.9):

$$l_n \approx c_n \cdot \left[\frac{h}{2}\right]^{(p+1)} \qquad (5.15)$$

Therefore, eliminating c_n in Eq (5.14) yields:

$$h' = h \cdot \left[\frac{\theta \cdot \epsilon}{l_n}\right]^{1/(p+1)} \qquad (5.16)$$

The usefulness of this formulation to determine the subsequent stepsize rests on the fact that it incorporates the absolute magnitude of the error generated by the previous step. This means that the calculation of the next stepsize is based on a quantitative assessment of exactly how successful the previous step was.

5.5 DYNAMIC BEHAVIOUR OF BIOLOGICAL SYSTEMS

The preceding sections have dealt with the selection of integration technique and step length adjustment in general terms. This information is now implemented for the simulation of biological system behaviour under dynamic conditions. The discussion is best introduced by considering a numerical example.

Consider the behaviour observed in an aerated batch reactor into which heterotrophic organisms (X_B) and a readily biodegradable soluble substrate (S_S) are introduced at time t = 0. Assume that the initial concentrations are X_{B0} = 1000 g.m^{-3} and S_{S0} = 100 g.m^{-3} respectively. Assume also that the behaviour in the batch reactor is governed by the model introduced earlier (Chapter 2, Table 2.2) and that the kinetic and stoichiometric constants are those used in the case studies (Chapter 4, Table 4.1). At the start of the batch test, the changes in concentration of X_B and S_S will be dominated by the growth process. Organism decay will exert only a minor influence on X_B. The rates of change of concentration at t = 0 will be:

For X_B:

$$\left.\frac{dX_B}{dt}\right|_0 = \hat{\mu} \cdot \frac{S_{S0}}{(K_S + S_{S0})} X_{B0} - b \cdot X_{B0}$$

$$= 4 \cdot \frac{100}{(5 + 100)} \cdot 1000 - 0.62 \cdot 1000$$

$$= 3189.5 \ g \cdot m^{-3} \cdot d^{-1}$$

For S_S:

$$\left.\frac{dS_S}{dt}\right|_0 = -\frac{\hat{\mu}}{Y} \cdot \frac{S_{S0}}{(K_S + S_{S0})} X_{S0}$$

$$= \frac{-4}{0.666} \cdot \frac{100}{(5 + 100)} \cdot 1000$$

$$= -5720.0 \ g \cdot m^{-3} \cdot d^{-1}$$

If these rates persisted unchanged, then the S_S concentration would be reduced by 100 percent to zero after a period of 25 minutes; at this time the concentration of X_S would be 1055.7 $g \cdot m^{-3}$ i.e. only 5.5 percent greater than the initial value. In practice, this would not occur, as the growth rate decreases with decreasing S_S concentration, particularly once the S_S concentration falls below 10 $g \cdot m^{-3}$. The actual progression of the batch test over the initial period would be as shown in Figure 5.1.

Let us now consider simulation of the batch test behaviour. If, for example, the Euler rule were employed and a steplength of 30 minutes (0.5 hours) were used, then the predicted concentration for S_S at $t = 0.5$ hours would be:

$$S_S = S_{S0} + \left.\frac{dS_S}{dt}\right|_0 \Delta t$$

$$= 100 - 5720.0 \cdot \frac{0.5}{24}$$

$$= -19.2 \ g \cdot m^{-3}$$

Clearly, this result is meaningless and much shorter step lengths would be required, perhaps of the order of 1 minute. In this case, after the first minute, the predicted concentrations of S_S and X_S would be:

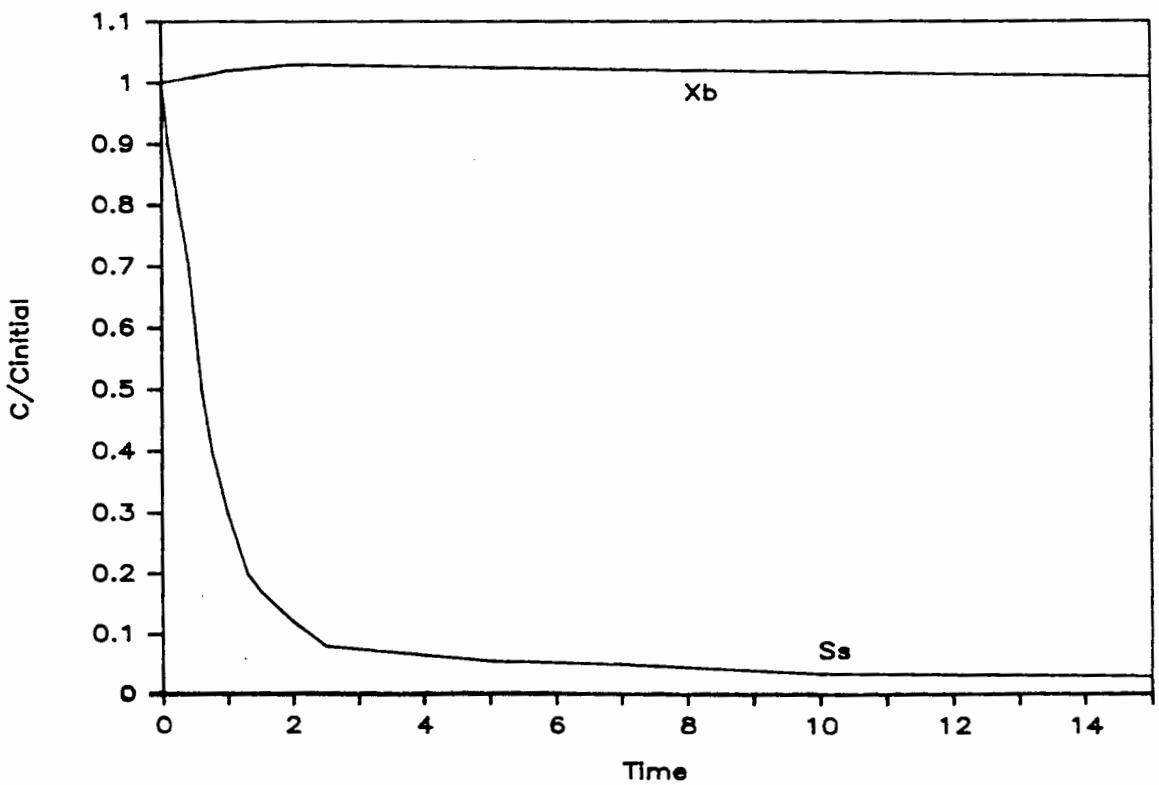


Figure 5.1 The progression of a batch test showing the response of S_s and X_s.

$$S_S = 100 + (-5720.0) \cdot \frac{1/60}{24}$$

$$= 96.0\ g \cdot m^{-3} \quad \text{(i.e. a change of 4 percent)}$$

$$X_B = 1000 + 3189.5 \cdot \frac{1/60}{24}$$

$$= 1002.2 \quad \text{(i.e. a change of 0.22 percent)}$$

Given that the percentage change of X_B is only one twentieth that of S_S over the interval, it appears that the step length of 1 minute is unnecessarily short to track the changes in X_B with acceptable accuracy. However, because the variables X_B and S_S are coupled, the two equations should strictly be integrated simultaneously. This implies that the step length used for the integration procedure will be limited by the maximum allowable size for the rapidly changing S_S, and X_B will be tracked with "unnecessary" accuracy.

Gear (1984) noted that behaviour similar to the reponse in the batch test is encountered in many engineering systems. Although strictly these variables are coupled, he suggested that the degree of coupling between the variables might not be strong. Gear proposed that, if this is so, then the differential equations for each of the variables may be integrated separately. In the batch reactor example, Gear's approach would mean that longer step lengths could be used for integrating X_B than those required for S_S. This offers the advantage of increased computational efficiency without compromising on accuracy requirements.

Implementing Gear's **multirate** approach involves partitioning a system of equations into different groups, each of which is governed by different dynamics. A group governed by "fast" dynamics would require short integration steps, whereas a group exhibiting "slow" dynamics could be integrated using longer integration steps. Gear proposed a number of schemes to account for the coupling between components with "fast" and "slow" dynamics.

5.6 THE USE OF A MULTIRATE TECHNIQUE

5.6.1 Methods for handling the coupled equations

Consider a two component system (one fast, one slow) which is described by two coupled ordinary differential equations. Assume that this system is integrated from t_o to $(t_o + \Delta t)$ using a stepsize h for the "fast" component and H for the "slow" component, where $H > h$ and $H = rh$. This division is shown in Fig 5.2.

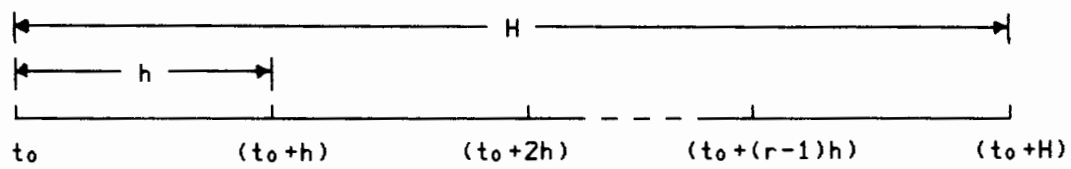


Figure 5.2 Schematic representation of small and large timesteps for a multirate integration technique

At any point in the integration process, values of both the "slow" and the "fast" variables will be required in order to complete the next integration step. At (t_o+h), the next integration step for the "fast" component to (t_o+2h) will require a knowledge of the value of the "slow" compound at least at (t_o+h). If the "slow" compound is integrated first, then its most recently computed value will be that at the end of the long time interval, t_o+H. If the "slow" compound has not yet been integrated, then the latest available value will be that at t_o, the beginning of the large time interval. In either case, an explicit estimate of the value of the "slow" compound at (t_o+h) is not available. The same problem will be encounterd at (t_o+2h), (t_o+3h),...etc. Estimation of the value of the "slow" compound at these intermediate points $(t_o+h, t_o+2h,\ldots,t_o+(r-1)h)$ is the crux of the problem of simultaneous multirate integration.

Two approaches to solving the problem have been suggested. The first approach ("fastest first") integrates the "fast" compounds over (r-1) steps of size h and then simultaneously integrates the "fast" and "slow" components to advance them to the end of the interval using stepsizes h and H respectively. For the small integration steps, values of the "slow" component are obtained by extrapolation from previous values (i.e. at t_0 and before) with a predictor-like formula. Gear (1984) notes that the error incorporated into the method through this extrapolation should be "of tolerable size". The disadvantage of this "fastest first" approach, however, is the amount of storage space consumed by the necessity to back up the "slow" variables in case of an integration step failure. If a step fails and the "slow" variables have not been stored, then they will need to be recalculated. The effort involved in accomplishing this task is an additional drawback to the "fastest first" approach.

The second approach ("slowest first"), which has been recommended by Gear (1984) and is adopted in this study, involves integrating the "slow" compounds to the end of the large timestep first. Interpolation techniques are then used to obtain intermediate values for the "slow" compounds at the points (t_0+h), (t_0+2h),...,$(t_0+(r-1)h)$. This then allows integration of the "fast" compounds in short steps from t_0 to $(t_0 + H)$. With this approach, the extrapolation of the "fast" variables to integrate the "slow" variables first will lead to large errors in the extrapolated values because the extrapolation is over many timesteps in the "fast" variables. Gear maintains that this is mitigated by the fact that coupling from the "fast" values to the "slow" values is generally small. This, in fact, is the basis for the method. An important advantage of the "slowest first" method is that, if a variable has to be backed up because of an integration failure in another variable, the backup is simply a reduction of the size of the last step taken, and can be done provided that only one additional value is kept for all variables.

5.6.2 Partitioning of a system

Partitioning a system into categories having different dynamics is an important consideration in the use of a multirate technique. Various methods for automatic partitioning of the system have been investigated (Gear, 1984; Oraigloglu, 1983). These methods have been found difficult to implement and expensive in terms of computational time. Therefore, static partitioning, where the division into categories is specified by the user prior to the integration, is generally applied.

A problem that arises in partitioning is what exactly defines dynamics as "fast" or "slow"? One possible answer is that "fast" components exhibit a large differential term i.e. (dC/dt) is large. Another possible solution would be that it is a rapidly changing (dC/dt) term that indicates "fast" dynamics. Alternatively, perhaps the ratio (dC/dt)/C is an appropriate measure for reaction systems where the concentration C cannot decrease to less than zero. The question does not appear to have been resolved in the literature and decisions as to the classification of compounds are usually based on practical experience and knowledge of the physical system.

5.6.3 Integration errors with a multirate technique

In choosing a time-stepping algorithm for the solution of the dynamic problem, the question of error estimation and control is a central one. Decisions as to whether or not concentration variables are acceptable have to be based on some estimation of how close these are to the actual solution. In using a multirate technique to carry out the integration procedure, the contribution to the global error in the method may stem from a number of sources. For the "slowest first" technique, the sources of error include the round off error and the local truncation error as described in Section 5.3.1 as well as:

(i) The error associated with the extrapolation of the "fast" components to allow integration of the "slow" components even though the fast components have yet to be integrated.

(ii) The error associated with the interpolation of the slower components to allow integration of the "fast" components. The interpolation error depends on the method used, and is due to two effects: firstly, the errors in the interpolation formula itself and secondly, errors due to errors in the mesh values. Gear (1984) notes that the errors in the interpolation formula will be significantly less than those in an extrapolation formula over the same interval.

In controlling the error, Gear (1984) suggests ensuring that the contributions from the interpolation and extrapolation are small in comparison to the local truncation error term. The local error term can then be used to select the next stepsize on the basis of a given error tolerance. An error tolerance term, ϵ, can be calculated for each component in each of the "fast" and "slow" groups. Once all the ϵ values for all the components have been computed, the largest ϵ value for each group is chosen as the limiting value. It is this limiting ϵ that will determine the size of the subsequent integration step for all the components in that group.

5.6.4 Stepsize selection with a multirate technique

The concept of using a multirate method is to reduce computation in an integration problem by using different stepsizes for groups of components with differing dynamics. Efficiency can be increased further by using the longest possible stepsize within each group.

Gear (1984) recommends an incremental approach for integration steplength adjustment that was used in the early stages of the development of the integration module here. Very simply, if the error in any component in either the "fast" or "slow" group is greater than the prescribed tolerance, then the value is rejected and the next steplength for that group will be half the size of the previous one. If the error is less than the prescribed tolerance, then doubling of the next

steplength is permitted. These stepsize changes are subject to certain constraints and may only take place at particular points in the integration scheme. This is possibly the most simple approach that can be taken in formulating some kind of dynamic relationship between error magnitude and stepsize control. One of its major limitations is that it makes no distinction between solution estimates that fall well within the prescribed error bounds and those that only just satisfy the error criterion. This is a significant limitation because the stability behaviour of the system is detrimentally affected by the accumulation of errors in the calculated variables.

A possible approach to overcome the limitations of Gear's method for steplength adjustment, would be to use the variable steplength adjustment method of Dahlquist and Bjorck (1974) (See Section 5.4). This is justified because the local error term is dominant.

5.7 IMPLEMENTATION OF GEAR'S MULTIRATE TECHNIQUE

The basis for developing and evaluating the multirate integration technique was a continuation of the case study first introduced in Section 2.5; that is, the single reactor plus settling tank problem based on the limited IAWPRC model (Table 2.2). Equations (2.14) to (2.21) are the set of differential equations and algebraic equations describing the sytem. Numerical values for the problem (reactor volumes, kinetic constants, etc) were the same as those used for Case Study 1 in Chapter 4 (Tables 4.1 and 4.2).

To introduce the dynamic component, a square wave cyclic input pattern was imposed on the system. In this scheme, the full volume of feed for Case Study 1 in Chapter 4 was introduced into the reactor at a constant rate but over a twelve hour period in a twenty four hour cycle. For the remaining twelve hours of the cycle, there was no feed. That is, the system was subjected to a step increase in flow rate and twelve hours later to a step decrease to zero flow. This input pattern was selected because, in the region of the step changes, it would provide a rigorous test of the integration method.

There are two requirements before a multirate integration technique can be initiated:

(i) For an initial value problem such as this, a set of values for each of the state variables at time t = 0 is required to initiate the integration. In this case, the simulation program uses the set of state variables which constitute the solution to the steady state problem as initial values for the dynamic case.

(ii) Implementing a multirate technique involves partitioning the compounds into categories with either "fast" or "slow" dynamics. The model incorporates four compounds: three particulate compounds (X_B, X_E and X_S) and one soluble compound (S_S). After investigating the nature of the dynamics of each of these compounds, it was decided to classify X_B and X_E as "slow" and S_S as "fast".

In the case of X_S, it was found by trial that the dynamics are neither as "fast" as those of the soluble compound nor as "slow" as those of the particulate compounds. In fact, it appears that, in certain circumstances, the behaviour of X_S changes from "fast" to "slow". Classifying the dynamics of X_S as "slow" and using long time steps for its intergration may result in inaccuracies in the solution. On the other hand, categorising it with the "fast" compounds in the system could result in needless extra computational effort as a result of the unnecessarily small timesteps being used at times when X_S exhibits "slow" dynamics. As a result, there seemed to be the potential to incorporate this compound into some kind of intermediate category. Creating an additional category for "intermediate" dynamics would thus have the advantage of enabling the routine to cater specifically for this compound and select an exactly appropriate stepsize for its integration. One drawback of this approach, however, would be the extra programming code required to extend the number of categories from two to three. If the whole purpose of a multirate technique is to improve efficiency, then the added complexity of accounting for

the coupling between three groups of variables would perhaps negate this objective at the outset. On the other hand, the ultimate efficiency of the technique depends on the appropriate partitioning of the system. In viewing these alternatives, it was decided to maintain the simpler approach and restrict the number of divisions to two. The effect of partitioning X_s with either the "fast" or the "slow" group is discussed in Section 5.9.1.

5.7.1 The initial multirate scheme

Gear (1984) has recommended that as many of the parameters as possible in a modern program code should be selected automatically. Achieving this for an integration scheme does present some difficulties particularly when implementing a multirate technique. This is due to the large number of parameter choices involved. As a result, the initial approach to the problem relied on prior specification of a number of the variables in accordance with the suggestions of Gear (1984):

(i) The initial stepsize for the "slow" components, H, was set at an arbitrary value to initiate the integration.

(ii) The ratio of the number of small steps to the number of large steps was also specified and remained fixed throughout the integration.

(iii) Initially, the steplengths could only be increased or decreased by a factor of two.

The "slowest first" technique recommended by Gear (1984) was followed. A simple Euler formula was used for both the "slow" and the "fast" integrations.

5.7.1 The initial algorithm for Gear's method

Step 1 : Select the following parameters:

(i) initial values for the state variables at t=0.

(ii) a stepsize, H, for the "slow" components

(iii) the ratio: $r = \frac{\text{number of "fast" steps}}{\text{number of "slow" steps}}$

where $H = r \cdot h$

(iv) an error tolerance, ϵ

Step 2 : Starting at $t = t_0$ and using the Euler formula, compute values for the "slow" compounds at $t = t_0 + H$:

$y(t_0 + H) = y(t_0) + H \cdot y'(t_0)$

Step 3 : Starting at $t = t_0$ and using the Euler formula, compute values for the "fast" compounds at $t = t_0 + H/r = t_0 + h$:

$y(t_0 + h) = y(t_0) + h \cdot y'(t_0)$

Use linear interpolation between t_0 and $(t_0 + H)$ to provide values for the "slow" compounds at $(t_0 + h)$.

Repeat this step r times, until the end of the large timestep is reached.

Step 4 : Evaluate the error for each of the "slow" and the "fast" compounds at the end of the large timestep.

If the error in any of the "fast" or "slow" compounds is larger than the prescribed tolerance then

(i) reject the most recently computed values of the variables

(ii) halve the large timestep by setting H = H/2

(iii) Return to Step 2.

If all the errors are less than the prescribed tolerance then

(i) accept the most recently computed values of the variables

(ii) double the large timestep by setting H = 2H

(iii) Return to Step 2.

Repeat Steps 2 to 4 until the end of the integration interval (24 hours) has been reached.

Step 5 : Check the values of the compounds ("fast" and "slow") at the beginning and end of the integration interval. If the difference between these is less than the prescribed error tolerance, then terminate the dynamic simulation.
Otherwise, use the most recently computed values as the initial values of the state variables and return to Step 2.

5.7.3 Deficiencies in the initial method

An obvious limitation of the initial approach is the fact that the error is checked only at the end of every large time interval. If the errors in all the compounds are acceptable, only then can the large timestep be doubled. If the error in any one of the compounds is not within the limits prescribed, then the large timestep is halved. Since the ratio of the number of "fast" to the number of "slow" timesteps remains constant, halving the size of the large timesteps also means halving the size of the small timesteps which may not be necessary.

The manner in which steplengths were adjusted is a major inefficiency in the method. In practice, the error in the "fast" compounds was found to be both larger and to accumulate more rapidly than that in the "slow" compounds. Allowing the error in the "fast" compound to accumulate until the end of the large timestep, besides being inefficient, also very often upset the success of the step. In addition, with a fixed ratio of small to large timesteps, the size of the large timestep often had to be unnecessarily reduced in order for the small timestep to be successful.

For this particular system, it was found that if an error in any component was permitted to approach the error bound, it then began to accumulate very rapidly. Eventually this affected the stability of the entire system. Consequently, the integration module needs to be formulated in such a way that errors could be evaluated as soon as any

time step, "fast" or "slow", had been completed. In addition, corrective action should be taken immediately. The cost of such an evaluation process was considered to be well worth it, as it was critical to the stability of the entire system.

5.7.4 An improved version

In an attempt to overcome the limitations outlined above, two refinements were incorporated into the algorithm:

(i) Allowance was made for the small timestep ("fast" dynamics) to vary independently of the large timestep ("slow" dynamics). This replaced the scheme of having a fixed number of "fast" steps per "slow" step. The error in the "fast" step was now evaluated immediately, and the short steplength doubled or halved as appropriate. With both the "slow" (large) and "fast" (small) timesteps being variable, full advantage is taken of the different dynamics of the system. Appropriate action is taken as soon as the error reaches unacceptable proportions. This improved version implies that the groups of "fast" and "slow" compounds are being integrated independently, which is correct, as their different dynamics suggest that they are only weakly coupled.

A restriction on the step adjustment procedure was that step doubling could only take place at a synchronisation point in the mesh. This was in accordance with the suggestion of Gear (1984), who motivated that synchronisation of the "fast" and "slow" meshes is desirable to prevent unnecessary interpolations. In this synchronisation scheme, Gear recommends that halving of a short step may take place at any time, but it may only be doubled when $(t-t_0)/h$ is an even number, where h is the current stepsize. If this doubling procedure is followed, then the end of a "fast" integration step will never fall beyond the end of the "slow" step i.e. the steps will be synchronised at the end of the "slow" step. This scheme requires that $(t-t_0)/h$ is an integer.

(ii) The simple Euler rule was replaced by a predictor-corrector pair. The Euler formula was retained as the predictor, and the second order trapezoidal rule was used as the corrector. With this approach, the number of function evaluations would be doubled at each integration step. However, it was hoped that the more sophisticated integration technique would allow more than a doubling of the step lengths, thus giving an overall increase in efficiency. The single application of the corrector was in line with the recommendation of Lapidus (1971).

5.7.4.1 Deficiencies in the improved method

Two problems were apparent with the improved scheme:

(i) With both timesteps being variable, synchronisation of the meshes is more difficult.

(ii) Computational effort can be wasted if the size of the error in the large timestep is not within acceptable limits and the step fails. The error in the "slow" compounds is only checked at the end of every large interval, which means that the already completed computation for the "fast" compounds is wasted if the large timestep is unsuccessful.

5.7.5 Further improvements

To address the deficiencies outlined above, two additional modifications were proposed:

(i) The first improvement in the integration method involved removing the synchronisation constraint. Thus, doubling and halving of "fast" and "slow" step lengths could take place at any point. In the case of the "fast" steps, if by doubling a "fast" step, it was found that the integration would move to beyond the end of the current "slow" step, then a smaller step would be taken to arrive exactly at the end of the "slow" step i.e. truncating to ensure

synchronisation. At the start of the next "slow" integration step, the new "fast" step length would be based on the step length calculated prior to truncation. This ensured relatively unlimited adjustment of the "fast" step lengths within the "slow" steps. In the case of the "slow" steps, truncation was only required where a "slow" step beyond a data storage point was attempted. In the new scheme, the "slow" step was truncated in a similar manner as for the "fast" step, to end at the data storage point.

(ii) With the improved method, the problem of computational effort "wasted" on the "fast" steps when the "slow" steps failed still existed. In an attempt to overcome the wasted effort, a scheme of multiple corrections was introduced into the integration routine. In this scheme, Euler's formula was used to predict a value for the "slow" compound at the end of the large timestep. The second order trapezoidal rule was implemented as a corrector for the "slow" compounds as before. If the error in the "slow" compounds at the end of the interval was found to be unacceptable, then the corrector was applied again in an attempt to improve the values and reduce the error to within the tolerance. This procedure was motivated by the fact that each correction offers the possibility that the new estimate might be a sufficient improvement to obviate the necessity to halve the steplength and re-perform the calculations. Up to five corrections were applied before the step was abandoned, and the "slow" steplength reduced i.e. from $PE(CE)^1$ to $PE(CE)^5$.

In practice, the multiple correction procedure was not helpful. It was found that the first application of the corrector gave a significant improvement on the value predicted by the Euler rule. However, with repeated applications of the corrector, the improvement was small and the rate of convergence was very slow - no benefit was derived in terms of efficiency.

5.7.6 A modified version of Gear's multirate method

At this stage, a thorough assessment of the integration routine based on Gear's approach illuminated a major defficiency in its operation. This was the fact that it did not account specifically for the range of magnitudes of the error generated at each timestep. The size of each new steplength was only based on whether the value generated at the previous step had satisfied or not satisfied the error criterion. Errors that only just satisfied the error tolerance were accepted and the following steplength was allowed to double, where it would have been more appropriate to increase the steplength by only a small amount. Doubling in this case caused the error to accumulate and a subsequent step would then fail.

The approach suggested by Dahlquist and Bjorck (1974) (See Sections 5.3 and 5.4) was used to develop a more sophisticated algorithm which adjusted the steplengths in a manner based on the absolute magnitude of the error generated at the previous step. For the selected predictor-corrector method (Euler/ trapezoidal), Eq (5.13) and Eq (5.16) were used to calculate the size of each new steplength. The relevant constants for the Euler predictor and the trapezoidal corrector are c = 2 and c' = 12, respectively (Lambert, 1974). Thus, Eq (5.13) becomes

$$\epsilon = \frac{2}{(12 - 2)} \cdot \frac{\% \text{ acc} \cdot y^p}{100}$$

$$= \frac{\% \text{ acc} \cdot y^p}{500} \qquad (5.17)$$

Substituting in Eq (5.16):

$$h' = h \cdot \left[\frac{\theta \cdot \% \text{ acc} \cdot y^p}{500 \cdot l_n} \right]^{1/(p+1)} \qquad (5.18)$$

where p = order of the method = 2

l_n = local error (from Eq (5.9)

After selecting appropriate values for θ and % acc, it is now possible to calculate a new stepsize, h', in such a way that it is appropriate to the magnitude of the error generated at the previous stepsize, h.

5.7.7 The final multirate integration algorithm

STAGE 1 : Select the following parameters:

(i) initial values for the state variables at t = 0
(ii) an initial stepsize, H, for the "slow" components
(iii) an initial stepsize, h, for the "fast" components
(iv) a percentage accuracy requirement, % acc
(v) a value of of the safety factor, θ
(vi) an integration interval for data storage
(vii) a stopping criterion for the 24 hour cycle

STAGE 2 : For each of the "fast" and "slow" components, calculate ϵ from Eq (5.13), using the initial values of the state variables as the predicted values, y^P:

$$\epsilon = \frac{1}{5} \cdot \frac{\% \text{ acc} \cdot y^P}{100}$$

FOR THE "SLOW" COMPOUNDS

STEP 1 : Starting at $T = t_0$ and using the Euler formula, compute values for the "slow" compounds at $T = t_0 + H$:

$$y(t_0 + H) = y(t_0) + H \cdot y'(t_0)$$

FOR THE "FAST" COMPOUNDS

Step 1 : Starting at $t = t_0$ and using the Euler formula, compute values for the "fast" compounds at $t = t_0 + h$:

$$y(t_0 + h) = y(t_0) + h \cdot y'(t_0)$$

Step 2 : Using straight line interpolation, find values for the "slow" compounds at $(t_0 + h)$:

$$y(t_0 + h) = y(t_0) + y'(t_0) \cdot (t - t_0)$$

Step 3 : Starting at $t = t_0$ and using the trapezoidal rule, compute corrected values for the "fast" compounds at $t = t_0 + h$:

$$y(t_0 + h) = y(t_0) + \frac{h}{2} \cdot (y'(t_0) + y'(t_0 + h))$$

Step 4 : Calculate an error term for each of the "fast" components at $(t_0 + h)$ using Eq (5.9):
Set Error = l_n / ϵ

Step 5 : Find the largest value of the error term for the "fast" components and use this to calculate the size of the next step using Eq 5.16:

$$h' = h \cdot \left[\frac{\theta \cdot \epsilon}{l_n}\right]^{1/3} = h \cdot \left[\frac{\theta}{\text{Error}}\right]^{1/3}$$

Step 6 : If, by taking this step, the end of the large timestep will not be reached, then
(i) replace h by h'
(ii) replace t by t + h
(iii) return to Step 1 for the "fast" components.
If, by taking this step, the integration moves to beyong the end of the large timestep, H, then replace h by H - t to arrive exactly at the end of the slow interval, H.
Having integrated to H for the fast compounds, continue to STEP 2 for the "slow" compounds.

STEP 2 : Starting at $T = t_0$ and using the trapezoidal rule, compute corrected values for the "slow" compounds at $T = t_0 + H$:

$$y(t_0 + H) = y(t_0) + \frac{H}{2} \cdot (y'(t_0) + y'(t_0 + H))$$

STEP 3 : Calculate an error term for each of the "slow" components at $(t_0 + H)$ using Eq (5.9):
Set Error = l_n / ϵ

STEP 4 : Find the largest value of the error term for the "slow" components and use this to calculate the size of the next step using Eq 5.16:

$$H' = H \cdot \left[\frac{\theta \cdot \epsilon}{l_n}\right]^{1/3} = H \cdot \left[\frac{\theta}{Error}\right]^{1/3}$$

STEP 5 : If, by taking this step, the end of the large timestep will not be reached, then replace H by H' and return to STEP 1 for the "slow" components.
Repeat STEPS 1 to 5 for the "slow" compounds until the size of the next step to be taken will move the integration of the "slow" compounds to beyond the end of the data storage interval.
Truncate the large timestep and use one that will arrive exactly at the end of the interval.
Continue to STAGE 3 of the general algorithm.

STAGE 3 : Store the values of all the state variables at the end of the data storage interval. Repeat STAGES 1 to 3 of the general algorithm until one 24 hour cycle has been completed.
Check if the differences between the values for all the state variables at the beginning and end of the cycle are less than the stopping criterion.
If this is so, then terminate the integration.
If not, then replace the initial values of the state variables with the most recently values.
Return to STAGE 2 of the general algorithm.
Continue integrating until convergence is achieved.

5.8 THE EFFECT OF CHOICE OF PARAMETERS

Successful operation of the final multirate method was found to be strongly influenced by the values specified for the parameters of percentage accuracy (% acc) and the safety factor (θ). The integration routine thus incorporated a facility for these parameters to be selected by the user according to the specific requirements of the system being analysed.

5.8.1 The effect of percentage accuracy

Once the accuracy requirement has been specified, the integration routine uses this value to calculate the limiting value of the local error (ϵ in Eq 5.13). Since ϵ controls the selection of subsequent stepsizes, it exerts a significant effect on the computational effort required to perform the integration.

Figure 5.3 shows the effect of different accuracy specifications on the behaviour of the "fast" variable S_s over a typical integration period of an hour when the input to the system is held constant and the integration is initiated at the solution i.e. the values of S_s should remain constant. Three different accuracy requirements were tested: 1.0%, 0.1% and 0.01%. The results are presented in Table 5.1.

When the percentage accuracy was specified as 1.0%, the size of the small timestep was, on average, 12 minutes long. Seven integration steps were necessary to reach the end of the interval. However, this includes two steps that were rejected when the accuracy requirements were not met. In addition, the response of S_s was unstable, oscillating more erratically as the timesteps became larger.

When the accuracy requirement was specified as 0.01%, eleven steps were necessary to reach the end of the time interval, the average steplength being 8 minutes. Only one of the steps failed to satisfy the error tolerance and the response of the variable S_s remained stable at all times. The price paid for the stability of the solution response is the necessity to use small step lengths throughout the integration and thus increase the computational effort expended. Examination of Figure 5.3 shows that, for this case, an accuracy requirement of 0.1% appears to meet the demands of stability whilst at the same time not requiring an excessive amount of computational effort, the average steplength being 10.6 minutes, with a very small oscillation in the response of S_s.

5.8.2 The effect of the safety factor, θ

Dahlquist and Bjorck (1974) recommend using a safety factor, $\theta \approx 0.8$ to account for the fact that error estimates are only approximations. To examine how the specification of this factor affected the integration, three different values for θ were selected and tested in the integration problem outlined in Section 5.8.1. Figure 5.4 shows the effect of the choice of θ on the step sizes permitted. When a small magnitude for θ of 0.5 was specified, steplengths greater than 16 minutes were never permitted, and were generally much shorter. The average stepsize for the integration interval was 8.3 minutes and 9 steps were required to reach the end of the interval. Only one of these was unsuccessful. (See Table 5.3).

On the other hand, when a large θ of 0.9 was specified, steplengths were generally longer (average length 10.3 minutes) with a largest steplength of 19.3 minutes. However, of the eleven steps required to reach the end of the interval, four were unsuccessful. In the light of this, it would appear that some intermediate value of θ would offer the most favourable balance between the number of steps required to complete the integration and the possibility of each of these steps being successful. For the purposes of simulation, a θ value of 0.75 was selected as fulfilling these requirements most appropriately.

5.9 FINAL COMMENTS ON PARTITIONING IN THE MULTIRATE METHOD

5.9.1 The effect of X_S as a "fast" or "slow" component

As noted in Section 5.7, the dynamics of X_S, the particulate substrate, were difficult to classify as either "fast" or "slow", and there was a general indication that this compound should occupy some "intermediate" category. However, as the creation of an additional class of compounds was not feasible, the effect of placing this compound in either the "fast" or "slow" categories was examined.

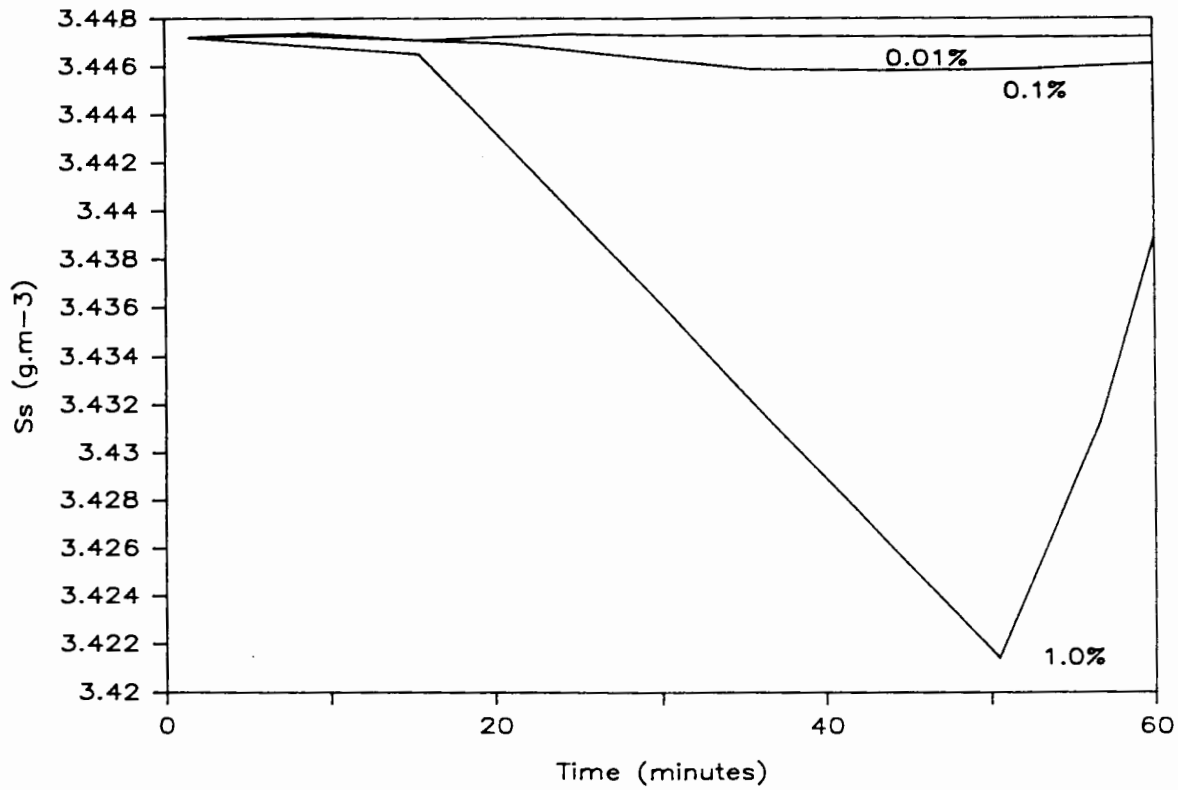


Figure 5.3 The effect of accuracy specifications on the behaviour of the variable S_s

% accuracy	Number of steps	Number of failures	Average stepsize
0.01 %	11	1	8.1 mins
0.10 %	10	3	10.6 mins
1.00 %	7	2	12.0 mins

Table 5.2 The effect of accuracy specifications on the behaviour of the stepping routine

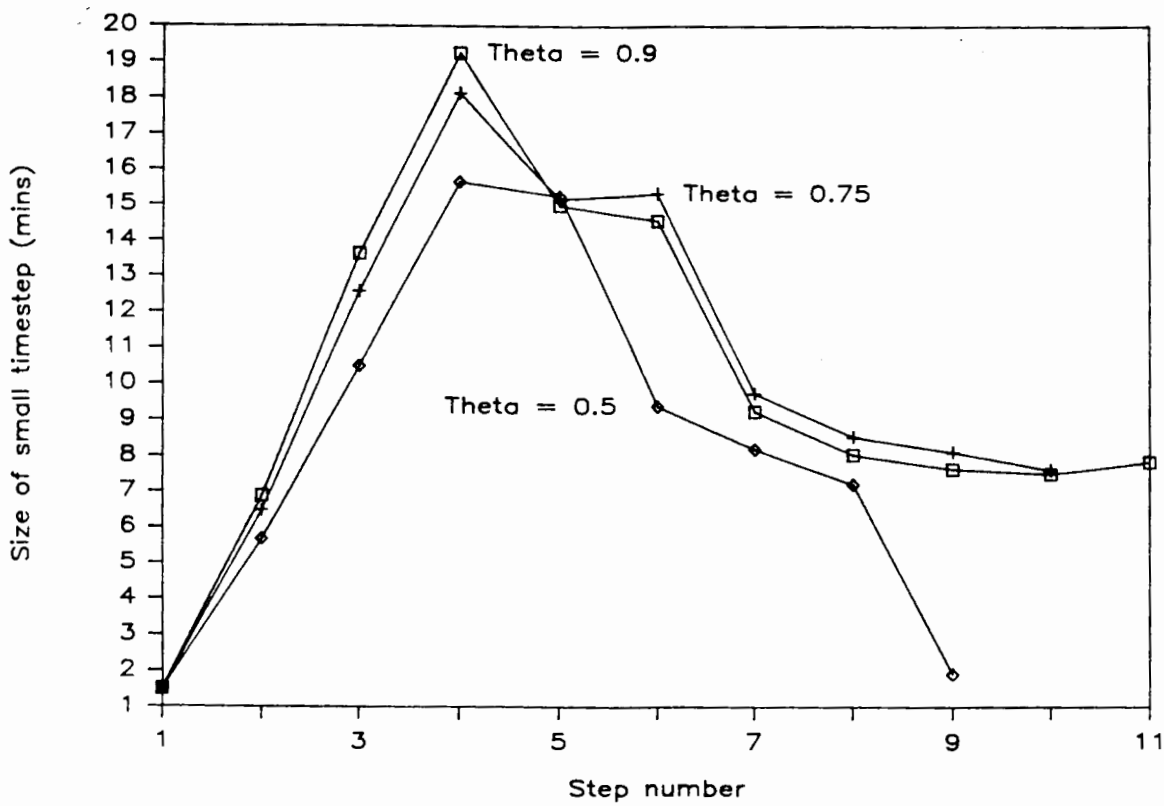


Figure 5.4 The effect of the safety factor on the size of the small timestep

Safety factor (θ)	Number of steps	Number of failures	Average stepsize
0.50	9	1	8.3 mins
0.75	10	3	10.6 mins
0.90	11	4	10.3 mins

Table 5.3 The effect of the safety factor on the behaviour of the stepping routine

When X_S was classified as a "fast" instead of a "slow" component, the sizes of both the "fast" and "slow" steps remained unaffected. In the case of the "fast" steps, this is to be expected, as it is S_S that is the "limiting" compound in the category and which exerts the dominating influence over stepsize selection. Classifying X_S as a "fast" component, however, means that computational requirements are increased for this group, as X_S is now being integrated using many small timesteps. That the size of the "slow" steps did not increase when X_S was removed from the group indicates in fact that X_S is appropriately grouped with the "slow" compounds. This was because, even when X_S was included in the "slow" category, it was observed that the errors in all the "slow" components were consistently small enough to enable the largest possible stepsize to be taken for each integration interval (i.e. H = data storage interval).

From a number of simulations it was found that, even though the dynamics of X_S were "faster" than those of the other compounds in the "slow" category, they were not sufficiently different to cause the error in the integration to increase significantly. As such, when errors at the end of each interval were evaluated, the error in the component X_S was still sufficiently small to allow the largest possible stepsize for subsequent integration steps.

5.9.2 A general comment on partitioning

Partitioning of the biological model has been done by comparing the dynamics of each compound to the other compounds in the model. Generally, this led to a division into soluble as "fast" and particulate as "slow" components. A limitation with this approach, which is general to the multirate method, was identified when simulating behaviour in systems with more than one reactor where there were large differences in reactor size. To illustrate the problem, consider the selector reactor configuration of Case Study 2 in Chapter 4. In this configuration, the first reactor volume was 1/32 that of the second. Obviously, shorter step sizes for the "slow" compounds, are required in the small reactor

with the short retention time than in the larger reactor. Similarly for the "fast" compounds in each reactor. However, because one step length is chosen for each group, the choice of the stepsize in fact is controlled by the variables in the small reactor. This steplength may be unnecessarily small for the compounds in the larger reactor. In fact, the situation could be encountered where, for optimal multirate efficiency, stepsizes for "fast" compounds in large reactors should be larger than stepsizes for "slow" compounds in small reactors. This problem arises because partitioning is on the basis of the model and not on the basis of individual compounds within the configuration of interest.

The limitation above could be overcome if automatic or dynamic partitioning were implemented. However, this possibilty has already been excluded and the limitation had to be accepted. On the other hand, it was felt that generating the division on the basis of the biological model generalised the method and simplified its implementation considerably.

5.10 CLOSURE

A multirate integration procedure has been found to be appropriate for biological systems. In these systems, the dynamics of the different compounds clearly divide into two groups, a "fast" group requiring short integration steps and a "slow" group for which the steps can be larger. Within each group, the range of dynamics is small compared to the difference between the two groups. A general guideline for partitioning a biological system is that it appears that soluble compounds can be grouped as "fast" and particulate compounds as "slow".

Some specific considerations should be noted as regards implementation of the multirate technique:

(i) Partitioning of the system on the basis of the model under analysis can lead to inefficiencies in the implementation of the multirate technique. This is one of the few drawbacks of the

method. In practice, partitioning on the basis of practical experience and by trial examination of the system was found to be the most flexible approach.

(ii) A predictor-corrector method with only one application of the corrector as proposed by Lapidus (1971) was found to be particularly suited to the dynamics of the system.

(iii) The steplength adjustment procedure proposed by Dahlquist and Bjorck (1974) has been found to be appropriate. The success of this procedure rests on the fact that it always uses the largest possible stepsize without allowing the errors to accumulate in the system. The method was found to be superior to Gear's method of steplength halving or doubling.

(iv) Discretion should be excercised in the choice of parameters such as percentage accuracy and the safety factor, θ.

CHAPTER SIX

CONCLUSIONS

The objective of this investigation was to develop and evaluate techniques which can be applied to the modelling and simulation of biological reaction system behaviour.

The model used as the basis for the investigation was a reduced version of the biological model proposed by the IAWPRC Task Group for mathematical modelling in wastewater treatment design. This limited model had the advantage of being easily manageable in terms of analysis and presentation of the simulation techniques. At the same time the model incorporated a range of kinetic formulations encountered with biological growth applications. Once the model had been selected, mass balance equations for each compound in each reactor could be formulated. These constitute a set of simultaneous non-linear ordinary differential and algebraic equations which, when solved, characterised the system behaviour. Two situations were considered for the purposes of simulation:

(i) steady state conditions, where the system operates under conditions of constant influent flow and load;

(ii) dynamic conditions, where the influent to the system varies with time, usually in a cyclic pattern.

The steady state problem:
Under steady state conditions the derivative terms in the differential equations fall away and the problem is reduced to one of solving a set of non-linear algebraic equations. Five approaches to computing the solution were evaluated:

(i) Direct linearisation
(ii) Successive substitution
(iii) The secant method of Wegstein

(iv) Newton's method
(v) Broyden's method

These methods were evaluated through application to five test cases representing the types of system configuration encountered in practice. The most efficient technique for the case studies was found to be Newton's method with a finite difference approximation to the Jacobian.

The dynamic problem:
The dynamic problem involves solving a set of coupled ordinary differential equations. The use of a multirate technique in combination with variable stepsizes for the integration was found to be a most successful approach. Aspects of particular importance concerning the method are:

(i) The system variables are partitioned into two groups: those with "fast" and those with "slow" dynamics. These two groups are integrated separately with different step lengths to reduce computational effort.

(ii) A general guideline for partitioning a biological system is that it appears that soluble compounds can be grouped as "fast" and particulate compounds as "slow".

(iii) The multirate method was based on the approach of Gear (1984). For step length adjustment, however, the approach of Dahlquist and Bjorck (1974) was found to be more efficient than the halving/doubling approach suggested by Gear. With Dahlquist and Bjorck's approach the magnitude of the adjustments to the steplength are based on the magnitude of the integration error.

LIST OF REFERENCES

Aitken A C (1925). On Bernoulli's Numerical Solution of Algebraic Equations. Proc. Roy. Soc. Edinburgh, 46, 289.

Borland International Inc (1985). Turbo Pascal Version 3.0. Borland International Inc. Scotts Valley. CA.

Broyden C G (1965). "A class of methods for solving non-linear simultaneous equations". Math Comp, 19, 577-593.

Broyden C G (1969). "A new double rank minimisation algorithm". AMS notices, 16, 670.

Dahlquist G and A Bjorck (1974). Numerical Methods. Prentice Hall Inc. Englewood Cliffs. New Jersey.

Dennis J R (Jnr) and J J More (1977). Quasi-Newton Methods, motivation and theory. SIAM Rev, 19, 46 - 49.

Dennis J R (Jnr) and R B Schnabel (1983). Numerical Methods for Unconstrained Optimisation and Non Linear Equations. Prentice Hall Inc. Englewood Cliffs. New Jersey.

Dold P L and G v R Marais (1985). Evaluation of the general activated sludge model incorporating specifications proposed by the IAWPRC task Group. Presented at the IAWPRC specialised seminar on Modelling of Biological Wastewater Treatment, Copenhagen, Denmark. Wat.Sci.Tech., 18, 63-89.

Gear C W (1971). Numerical Initial Value problems in ordinary differential equations. Prentice Hall Inc. Englewood Cliffs. New Jersey.

Gear C W (1984). Multirate linear multistep methods. Bit, 24, 484-502.

Grau P, P M Sutton, M Henze, S Elmaleh, C P L Grady, W Gujer and J Koller (1982). Recommended notation for use in the description of biological wastewater treatment processes. Water Research, 16, 1501.

Henze M, CP Leslie Grady, W Gujer, G V R Marais and T Matsuo (1987). Abbreviated Report: A general model for single-sludge wastewater treatment systems. Water Research, 21, 5, 505-515.

Johnston R L (1982). Numerical Methods, A Software Approach. John Wiley and Sons. Canada.

Lambert J D (1979). Computational Methods . John Wiley and Sons. Great Britain.

Lapidus L and J H Seinfeld (1971). Numerical Solution of Ordinary Differential Equations. Academic Press. New York.

Levenspiel O (1972). Chemical Reaction Engineering. 2nd Ed. John Wiley and Sons. New York.

Myers A L and W H Sieder (1976). Introduction to Chemical Engineering and Computer Calculations. Prentice-Hall, Inc. Englewood Cliffs. New Jersey.

Orailoglu A (1983). Software design issues in the implementation of hierachical display editors. Rept No. UIUCDCS-R-83-1139. Dept. Computer Sci. Univ. Illinois.

Petersen E E. (1965). Chemical reaction analysis. Prentice Hall. Englewood Cliffs. New Jersey.

Reklaitis G V (1983). Introduction to Material and Energy Balances. John Wiley and Sons. New York.

Sargent R W H (1981). A Review of Methods for solving non-linear Algebraic Equaitons. In RSH Mah and W D Sieder (eds). Foundations of Computer-Aided Chemical Process Design. Engineering Foundation. New York, Vol 1, 22-76.

Steffensen J F (1933). Remarks on Iteration. Skand. Aktuar. Tidskr, 16, 64 - 72.

Water Research Commission (1984). Theory, design and operation of nutrient removal activated sludge processes. Published by the Water Research Commission. P O Box 824. Pretoria. 0001.

Wegstein J H (1958). Accelerating Convergence of Iterative Processes. Comm. ACM, 1,(6),9.

Westerberg A W, H P Hutchison, R L Motard and P Winter (1979). Process Flowsheeting. Cambridge University Press.