1982

Multi-dimensional neutron diffusion

Jerard Michael Barry
University of Wollongong

Recommended Citation
NOTE

This online version of the thesis may have different page formatting and pagination from the paper copy held in the University of Wollongong Library.

UNIVERSITY OF WOLLONGONG

COPYRIGHT WARNING

You may print or download ONE copy of this document for the purpose of your own research or study. The University does not authorise you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site. You are reminded of the following:

Copyright owners are entitled to take legal action against persons who infringe their copyright. A reproduction of material that is protected by copyright may be a copyright infringement. A court may impose penalties and award damages in relation to offences and infringements relating to copyright material. Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.
Multi-Dimensional Neutron Diffusion

by

Jerard Michael Barry

B.Sc. (UNSW)

M.Sc. (Sydney)

Submitted for the degree of

Doctor of Philosophy

in the

School of Mathematics

at

University of Wollongong

April 1982
This thesis is submitted to the University of Wollongong and has not been submitted for a higher degree to any other University or Institution.

Jerard Barry

April 1982
Grau, teurer Freund, ist alle Theorie
Und grün des lebens goldner Baum.

Goethe
The numerical solution of time dependent neutron diffusion approximation to the transport equation is of vital interest to those concerned with reactor design and safety. The growth of modern computing power has increased the scale with which computations may be undertaken. To exploit these electronic advances fully, however, it is necessary to review existing numerical approaches and substitute more efficient techniques wherever appropriate.

This work summarises an approach typical of the conventional method of solution. It concentrates subsequently on the heart of the method, namely the solution of very large sparse systems of linear equations. A new numerical approach is formulated. This essentially seeks a splitting for the iteration matrix which makes the iterative process more implicit. The role that 'experimental mathematics' and computer graphics played in its evolution is discussed. The method has three other possible applications. Two of these are pursued here. Of these secondary applications, the acceleration of convergence in energy has tremendous potential for thermal reactor studies.

Various aspects concerning the behaviour of the new iterative approach are observed. The method is tested extensively on a number of reactor configurations which demonstrate several aspects of design and modelling techniques. The implicit approach is contrasted and
compared with relaxation and conjugate gradient methods. Considerable attention is devoted to the efficient implementation of the three iterative schemes on computers with virtual memory.

Traditionally, a secondary acceleration of the iterative technique is achieved with a variational approach. Several modern variations of this, with linear approximating functions instead of constant rescaling factors are considered. The scheme requires the solution of a reduced system of linear equations. Matrix properties of the reduced system are considered, and methods of their solution discussed.

Justification for the secondary acceleration is analysed with a Fourier approach. The applicability of each scheme to the three basic iterative methods is considered by analysing the reduction of frequency components of error in the solution estimate. Predictions from theory are then compared with results on several reactor geometries, with and without the secondary acceleration.
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Background to nuclear reactor modelling</td>
</tr>
<tr>
<td>2</td>
<td>Derivation of the neutron diffusion equation from transport theory</td>
</tr>
<tr>
<td>3</td>
<td>Numerical solution of the time dependent multigroup neutron diffusion approximation equations</td>
</tr>
<tr>
<td>4</td>
<td>Method of implicit non-stationary iteration (MINI) and other iterative approaches</td>
</tr>
<tr>
<td>5</td>
<td>Computational implementation of ICCG and MINI</td>
</tr>
<tr>
<td>6</td>
<td>Acceleration of the iterative process by coarse mesh rebalancing</td>
</tr>
<tr>
<td>7</td>
<td>Fourier analysis of iterative procedures and coarse mesh rebalancing</td>
</tr>
<tr>
<td>8</td>
<td>A comparison of SLOR, MINI and ICCG on real reactor models</td>
</tr>
<tr>
<td>9</td>
<td>Results of coarse mesh rebalancing</td>
</tr>
<tr>
<td>10</td>
<td>Conclusion and summary</td>
</tr>
<tr>
<td>A</td>
<td>Temporal integration of the time dependent neutron diffusion equation</td>
</tr>
<tr>
<td>B</td>
<td>Spatial discretisation of the neutron diffusion equation</td>
</tr>
<tr>
<td>References</td>
<td></td>
</tr>
</tbody>
</table>
'You mustn't expect too much meaning. But to me there seems a hint. That's all. A hint.'

Graham Green, 'The Hint of an Explanation'.

The degree of refinement possible for the formulation and solution of the time-dependent neutron diffusion equation is related to the capacity and performance of available computer facilities. Early computer systems had limited directly addressable memory and low instruction speeds. This restricted reactor studies to the solution of reactor point models, when temporal aspects were considered. The analogue computer, rather than the digital machine, offered a more attractive means of handling models that involve time dependent variation (Gilbert [1964]).

The analogue computer is still faster than the digital machine for the point kinetics studies, despite enormous advances in digital technology. Even with problems arising from their limited accuracy, and the difficulties in setting up program boards, there are many applications even today (particularly real time situations), where the use of an analogue or hybrid system is preferred.
As digital computers advanced, it became feasible to solve steady state reactor models involving one spatial variable. By the late 50's and the development of second generation digital computers, two-dimensional (space) problems were fairly typical of the type of problem that could be investigated (e.g. the PDQ series of codes [Bilodeau et. al. 1957]). The inclusion of time variation didn't occur until the mid 60's [Stacey 1969].

Analogue systems can handle the spatial aspects of reactor physics problems to a certain degree, however, their accuracy is a limiting factor. With the large number of electronic components necessary to discretise the spatial domain in problems of interest to reactor designers, however, they fail to be competitive.

Throughout the history of nuclear code development, reactor physicists always have extended their models to the limit imposed by the computing facilities available. It is no coincidence in the 60's that many of the largest computers were employed in the nuclear industry. This tendency to saturate the available computing facilities has continued for later generations of computers, where solution of three-dimensional models with kinetics are now possible.

The need for more accurate reactor modelling has grown since the time relatively crude approximations were used to design the first atomic piles. More accurate modelling allows for less conservative design criteria (forced by a lack of knowledge), enabling more
efficient operation. In a span of about thirty years, a tremendous extrapolation in operating power of reactor systems has occurred. These advances were enabled by strong research programs in several advanced industrial nations. Significant changes in reactor technology have resulted e.g. advances in gas reactors, the use of highly pressurised cooling systems, fuel enrichment, and the promise of broader reactors to increase the life of fuel reserves. Every advance in technology required a more accurate and detailed knowledge of the many aspects of reactor behaviour, even for the preliminary design stages. The demand for numerical schemes to provide answers led to vast sums of money being expended on computing facilities in this industry. These are justified when offset against the capital investment in nuclear plant itself.

This work is concerned with only one aspect of the computational requirements in reactor design, namely the determination of neutron flux distribution. It should be remembered, however, that the end product of this work, a nuclear code, is required to interface with many other modules of equal complexity concerned with other aspects of reactor integrity. Together these modules constitute the nuclear engineer's basic tools.

Reliance on raw computing power alone to extend the existing nuclear codes, so they may handle the new requirements and complexities of reactor models is insufficient. Mere extrapolation of the older numerical techniques to tackle the larger modern problems is
not necessarily appropriate. Some possible numerical advances are considered in this work.

The structure of this thesis is summarised next. The traditional introduction and survey of publications pertinent to the topics discussed are given within the remaining chapters as new aspects arise.

In Chapter 2 the scene is set with an outline of the neutron diffusion approximation to the transport equation, and the interface between this work and the overall determination of neutron distribution is identified.

An outline of the numerical techniques used to solve the fine mesh, few energy group stage of the overall process is given in Chapter 3. These procedures form the nuclear code POW3D which evolved in this work. Practical nuclear codes are by necessity large and POW3D is no exception. It comprises approximately twenty thousand source statements.

The purpose of this work was to develop a three-dimensional working nuclear code with some limited kinetics capabilities. This document primarily concentrates on the heart of the nuclear code, i.e. the solution of a large sparse system of linear equations. A new technique for solving these equations, the Method of Implicit Non-stationary Iteration (MINI) is introduced. The technique is compared
with the recently modified conjugate gradient approach, while the traditional method of successive over-relaxation is included to complete the evaluation.

In Chapter 4 MINI is described from a somewhat historical perspective. MINI didn't eventuate overnight, it grew out of earlier attempts to make the Gauss-Seidel method of solution more implicit. Considerable use of computer graphics made its refinement possible. It is doubtful that a pleasing finished procedure would have emerged without this experimental phase.

Unfortunately no general proof for its convergence has been discovered. The method has never failed for the class of matrix envisaged in reactor physics, for which MINI specifically was designed. Convergence frequently was observed for a wider class of matrix. Various interesting aspects of MINI's behaviour are presented, and an attempt is made by restricting MINI's flexibility to associate it with the class of relaxation methods. Historically, however, MINI didn't evolve from such techniques. It is conjectured finally that MINI, in some ways is a practical machine implementation for the class of relaxation methods based on human intuition.

The work covers the implementation of such techniques on modern generation IBM computer systems. In Chapter 5 considerable attention is given to the data structures and I/O transfers necessary to make the code perform efficiently in a virtual machine environment, while
allowing it to be transported to machines with older memory systems, without great difficulties. With the continuing evolution in computer architecture, numerical techniques which once were accepted as optimal will require closer scrutiny.

The transition to a machine with virtual memory occurred during this work. The nuclear code that resulted performs successfully in the old and new worlds. The particular IBM machine used is a sequential machine for instruction processing; there is little instruction look-ahead or decoding. As pipeline designs become more commonplace these developments should be taken into account in devising numerical techniques. Ultimately, however, the integration of processors capable of working in parallel, may result in a fundamental re-appraisal of numerical techniques currently employed. It will be necessary for a coordinated approach by hardware designers, language developers and mathematicians to maximise the benefit technological changes can bring. At present there is a tendency for innovators to be too narrow. This results in computer scientists often experimenting with new parallel hardware on mathematical techniques which mathematicians have discarded for many years. At the same time an excessive pre-occupation with mathematical rigour, at the expense of everything else, may cause some worthwhile numerical techniques to be passed over, while difficulties associated with implementation of the most respectable mathematical techniques may not surface in the abstract environment. An additional danger exists, whereby the value of mathematical analysis on sequential algorithms may be undercut as a
new generations of parallel hardware emerge.

Secondary techniques exist for accelerating the iterative solution of large systems of linear equations. These have been used in reactor physics for a long time. There are many variations of these and some are considered in Chapter 6. Recent developments, properties of the reduced matrices and methods of solution are discussed.

Speculations as to why these secondary methods assist convergence are considered. In Chapter 7 a Fourier analysis of the removal of error is performed to determine whether these techniques are appropriate for use in conjunction with the newer primary iterative methods.

In the long run, however, it is necessary to test the procedures on actual reactor models. Chapter 8 is devoted to a comparative testing of the traditional relaxation approaches against MINI and the recent conjugate gradient variation. Chapter 9 is concerned with the performance of the coarse mesh rebalancing techniques relative to each iterative scheme.

Throughout this work theorems quoted without proof are attributable to others and the source is identified. Theorems stated and proved indicate either a new result, an attempt to supply a more rigorous proof for a known result, or a formal proof not noted previously.
CHAPTER 2

DERIVATION OF THE NEUTRON DIFFUSION EQUATION FROM TRANSPORT THEORY

'Beware the past;
Within it lie
Dark haunted pools
That lure the eye
To drown in grief and madness.'

James Mc Auley, 'Warning'.

2.1 INTRODUCTION

Derivation of multigroup neutron diffusion approximations for the Neutron Transport Equation are discussed frequently in the literature [Glasstone and Edlund 1952; Duderstadt and Hamilton 1976; Henry 1975]. Consequently only a brief introduction to the key terms involved in modelling reactor behaviour is given in this section, but much of the tedious algebra is avoided. Instead Chapter 2 is devoted to demonstrating some of the difficulties and shortcomings involved in approximating transport theory with a diffusion model.

It is possible to write down an appropriate diffusion equation that represents idealised reactor behaviour by invoking Fick's law directly. The simplicity of this approach, however, fails to consider many of the assumptions and approximations necessary for describing what is really a transport phenomenon.
After the diffusion theory approximation is established in this chapter, the basis for the contribution of the remainder of this work to the overall study of neutron behaviour is demonstrated. The interface between the practical nuclear code developed here and the existing suite of programs at the AAEC's Research Establishment is discussed.

2.2 NEUTRON TRANSPORT EQUATION

The concept of neutron density in phase space $N(r,\Omega,E,t)$ is fundamental, for it allows the number of neutrons in a small element of volume $dV$ surrounding a point $r$, and travelling in a small solid angle $d\Omega$ surrounding the direction $\Omega$ and contained in a small energy band $dE$ about energy $E$ at time $t$ to be written as

$$N(r,\Omega,E,t)dVd\Omega dE$$

From the neutron density in phase space, a second quantity the directional flux density (sometimes known as the angular neutron flux)

$$\gamma(r,\Omega,E,t) \equiv v(E) N(r,\Omega,E,t)$$

immediately follows, where $v(E)$ denotes the velocity of neutrons (with energy $E$). The directional flux density is a more useful quantity than neutron density in measuring the interactions between neutrons travelling in the beam and the nuclei of the media through which the
Two further quantities, the **scalar flux density** $\phi(r,E,t)$ and **net current density** $J(r,E,t)$ are required. For the scalar flux density,

$$\phi(r,E,t) = -f(r,n,E,t)\frac{d\Omega}{d\Omega} - v(E) N(r,\Omega,E,t)\frac{d\Omega}{d\Omega},$$

the angular dependence of $\nu(r,\Omega,E,t)$ is 'averaged out' through integration in $\Omega$. $\phi(r,E,t)dVdE$ measures the total number of neutrons in $dEdV$ (without regard to the direction in which they travel) multiplied by their speed. This is a key quantity to be computed in the diffusion theory formulation of the problem, because it is relevant to the probability of an interaction between a neutron and an atomic nucleus. The net current density is defined as

$$J(r,E,t) = \nabla' r(r,n,E)dn.$$

$J(r,E,t)$ signifies the maximum over all orientations of a unit surface located at $r$ of the net number of neutrons with energies in the range $dE$ about $E$ crossing that unit surface per second, the direction of the vector quantity being the direction of maximum net flow.

The neutron continuity (or balance) equation is derived for the steady state condition (so time dependence will be omitted from $\nu(r,\Omega,E,t)$ etc. for simplicity). To formulate this equation the concept of a nuclear cross section is required. The microscopic cross
section \( \sigma^j_a(E) \) expresses the relative probability that a nuclear interaction \( \alpha \) will occur between a neutron of energy \( E \) and the isotope \( j \). Consequently when there are \( n^j(r) \) nuclei of isotope \( j \) per unit volume at the point \( r \), the expected number of interactions of type \( \alpha \) with nuclei \( j \) in an element of phase space \( dV \) is

\[
N(r,\Omega,E)v(E)n^j(r)\sigma^j_a(E)dV = \gamma(r,\Omega,E)n^j(r)\sigma^j_a(E)dVd\Omega dE.
\]

For a large neutron population a deterministic description of neutron behaviour is possible, the number of interactions of type \( \alpha \) in \( dVd\Omega dE \) being

\[
\sum_j \gamma(r,\Omega,E)n^j(r)\sigma^j_a(E)dVd\Omega dE.
\]

With the introduction of a macroscopic cross section \( \Sigma^j_\alpha(r,E) = n^j(r)\sigma^j_a(E) \) for a single isotope \( j \) and a summation over all isotopes, the number of interactions of type \( \alpha \) in \( dVd\Omega dE \) is expressed as

\[
\sum_j \Sigma^j_\alpha(r,E)\gamma(r,\Omega,E)dVd\Omega dE = \Sigma_\alpha(r,E)\gamma(r,\Omega,E)dVd\Omega dE.
\]

Consider a portion of phase space \( dVd\Omega dE \) in the form of a cylinder (Figure 2.1), where \( dVd\Omega dE = ds|d\Omega|d\Omega dE \), then the rate at which neutrons are removed from the cylinder by scattering and absorption for a beam of neutrons with energies in the range \( E \) to \( E+dE \) and travelling within a solid angle surrounding \( \Omega \) is
A cylinder used for the derivation of a fundamental neutron balance condition

Figure 2.1
\[ \Sigma_t(r,E) \gamma(r,\Omega,E)dVd\Omega dE \tag{2.2.1} \]

where \( \Sigma_t(r,E) \) is the total macroscopic cross section.

The rate at which neutron leakage occurs at the extremities of the cylinder is

\[ \left[ \gamma(r+dr,\Omega,E) - \gamma(r,\Omega,E) \right] dEd\Omega dS \]

which after a Taylor expansion and neglecting the higher terms in \( dr \) gives a leakage of

\[ \Omega \cdot \nabla \gamma(r,\Omega,E)dVd\Omega dE \tag{2.2.2} \]

Neutrons can be added to \( dVd\Omega dE \) by the fission process. Fission neutrons are assumed to be isotropic. The macroscopic fission cross section is denoted \( \Sigma_f^j(r,E') \), while \( \gamma^j(E') \) represents the number of neutrons emitted when a neutron of energy \( E' \) causes fission in isotope \( j \), and \( f^j(E) \) is the fission spectrum for isotope \( j \) (i.e. \( f^j(E)d\Omega dE \) represents the fraction of neutrons emitted from the isotope in \( d\Omega dE \)). An expression for the addition of neutrons to \( dVd\Omega dE \) then is given by

\[ \sum_j f^j(E)dE d\Omega \int d\Omega' \int dE' \gamma^j(E') \Sigma_f^j(r,E') \gamma(r,\Omega',E')dV \tag{2.2.3} \]
Neutrons may enter $d\Phi dE$ in the cylinder by scattering, the number entering is

$$
\int dE' \int d\Omega' [\Sigma_s (r, \Omega' \rightarrow \Omega, E' \rightarrow E) d\Omega dE] \Psi (r, \Omega', E) dV ,
$$

(2.2.4)

where $\Sigma_s (r, \Omega' \rightarrow \Omega, E' \rightarrow E)$ is the differential scattering cross section for neutrons with energy $E'$ and direction $\Omega'$. Neutrons also may be added from an (n,2n) reaction, where a nucleus excited by a neutron capture emits two neutrons as it relaxes. This reaction may be viewed as similar to fission, Henry [ibid.], but other treatments involve adjustments to the appropriate cross sections [Robinson 1975]. The probability of such reactions occurring are orders of magnitude lower than for the fission process itself.

Finally neutrons may enter from non-fission sources; the number entering from a source of density $Q(r, \Omega, E)$ is

$$
Q(r, \Omega, E)V d\Omega dE .
$$

(2.2.5)

The steady-state condition of balance between neutron creation and neutron loss is obtained by combining equations (2.2.1, 2.2.2, 2.2.3, 2.2.4 and 2.2.5). This gives the steady-state Boltzman transport equation in terms of the directional flux density $\Psi (r, \Omega, E)$ viz.
\[ \nabla \cdot \mathbf{\gamma}(r, \Omega, E) + \Sigma_t(r, E) \mathbf{\gamma}(r, \Omega, E) = \int dE' \int d\Omega' \left[ \Sigma_f^{\text{HE}}(E) \mathbf{\gamma}_f^{\text{HE}}(r, E') \right] + \Sigma_s(r, \Omega' \rightarrow \Omega, E' \rightarrow E) \mathbf{\gamma}(r, \Omega', E') + Q(r, \Omega, E). \] (2.2.6)

Finally, consideration of boundary and other conditions imposed upon the solution to (2.2.6) is required. On physical grounds \( \mathbf{\gamma}(r, \Omega, E) \geq 0 \) for all \( r, \Omega \) and \( E \); \( \mathbf{\gamma}(r, \Omega, E) \) must be continuous in \( r \) for direction \( \Omega \). The assembly on which the calculation is performed is assumed isolated so that the boundary condition involves no incoming angular flux. Mathematically, for \( \mathbf{n} \) an outward pointing normal to the surface,

\[ \mathbf{\gamma}(r, \Omega, E) = 0 \] for all \( r \) on the surface with \( \mathbf{n} \cdot \Omega < 0 \).

The steady-state transport equation (2.2.6) may be generalised to include time dependence

\[ \frac{1}{V} \frac{\partial}{\partial t} \mathbf{\gamma}(r, \Omega, E, t) + \mathbf{\gamma}_S \mathbf{\gamma}(r, \Omega, E, t) + \Sigma_t(r, E, t) \mathbf{\gamma}(r, \Omega, E, t) = \int dE' \int d\Omega' \left[ \Sigma_f^{\text{HE}}(E) \mathbf{\gamma}_f^{\text{HE}}(r, E') + \Sigma_s(r, \Omega' \rightarrow \Omega, E' \rightarrow E) \right] \mathbf{\gamma}(r, \Omega', E', t) + Q(r, \Omega, E, t). \] (2.2.7)

Should an efficient solution be possible for either form of the transport equation, then the accuracy of the solution would be limited only by the errors involved in the determination of the physical cross sections. Unfortunately such a solution is unavailable. Solutions
may be attempted by various methods including spherical harmonic expansions, Fourier transformation, discrete ordinate and Monte Carlo techniques [Clancy 1971]. The last two form the basis of most nuclear codes that seek an explicit consideration of transport theory effects, while the first two are too limited for large scale practical geometries. They do, however, act as the basis for establishing the more computationally efficient and economical multigroup diffusion theory with which most full scale reactor calculations are undertaken.

2.3 OPERATOR REPRESENTATION OF THE TRANSPORT EQUATIONS

The transport equations may be expressed in operator form. The steady-state equation (2.2.6) becomes

\[ L \hat{\Psi} = M \hat{\Psi} + Q, \]

where

\[ L = \mathbf{\Omega} \cdot \nabla + \Sigma_t(r,E) - \int \int dE' \Sigma_s(r,\mathbf{\Omega}', \mathbf{\Omega}, E' \rightarrow E), \]

and

\[ M = \int \int dE' \Sigma_f(r, E') \mathbf{\nabla} \mathbf{j} \mathbf{x} \rightarrow \mathbf{j} \mathbf{x} \rightarrow \mathbf{j} \Sigma_f(r, E'), \]

represent net loss and fission production operators respectively. In the absence of an external source (Q=0) the transport equation is
homogeneous and non-zero solutions only exist under special conditions. The existence of a non-zero solution corresponds to a critical physical assembly. A non-zero solution can be imposed by artificially adjusting the number of neutrons produced in the fission process through the introduction of a positive real number $\lambda$. Consequently a non-trivial mathematical solution to

$$L \gamma = \frac{1}{\lambda} M \gamma$$  \hfill (2.3.1)$$

is anticipated, where the largest $\lambda$ corresponds to the multiplication factor ($k_{\text{eff}}$) of the assembly.

There are an infinite number of eigenvalues $\lambda_n$ and corresponding eigenfunctions $\gamma_n(\lambda) (r, \theta, \phi, E)$ satisfying (2.3.1) and the boundary conditions; but not the positivity condition. Knowledge about the nature of $\lambda_n$ is very limited, and the completeness of the eigenfunctions has not been established [Henry ibid.]. This unfortunately precludes the rigorous use of eigenfunction expansions. In practice, however, completeness is assumed frequently.

2.4 DERIVATION OF THE CONTINUOUS ENERGY DIFFUSION EQUATION

The continuous energy diffusion equation can be derived from the transport equation. An outline of the derivation is presented only for the steady state case (this can be extended, however, to the time dependent form). The use of spherical harmonics gives one method for
solving the transport equation. The spherical harmonics form a complete set of functions in the spherical polar angular variables $\mu(=\cos \theta)$ and $\varphi$. These variables are sufficient to define the unit vector $\Omega$. The functions are defined as

$$\gamma_n^m(\Omega) = \gamma_n^m(\mu, \varphi) = \frac{(2n+1)(n-m)!}{(n+m)!} \sum_{j=0}^{(2n+1)/2} \frac{1}{Z} P_n^m(\mu) e^{im\varphi},$$

where $P_n^m(\mu)$ are the associated Legendre polynomials. The directional flux density then can be expanded as

$$\psi(r, \Omega, E) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \gamma_n^m(r, E) \gamma_n^m(\Omega),$$

and orthonormality allows the coefficients to be expressed formally as

$$\gamma_n^m(r, E) = \int d\Omega \gamma_n^m(\Omega) \psi(r, \Omega, E),$$

where $\gamma_n^m(\Omega)$ is the complex conjugate of $\gamma_n^m(\Omega)$. The cross sections $\Sigma_s(r, \Omega \rightarrow \Omega', E' \rightarrow E)$ involving angular dependence are expanded similarly

$$\Sigma_s(r, \Omega \rightarrow \Omega', E' \rightarrow E) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \Sigma_s^m(r, E' \rightarrow E) \gamma_n^m(\Omega),$$

and the coefficients again are
Substitution of these expansions into the transport equation leads to an infinite set of coupled equations. (The form of these is rather complicated for general geometries; details are given by Weinberg and Wigner [1958]; Bell and Glasstone [1970].) The so-called $P_N$ approximation results from truncating the expansion after the first $N$ terms. In practical situations it is difficult to apply such a procedure to three-dimensional models for $N > 1$. The simpler $P_1$ approximation forms the basis for diffusion theory, and the justification for the restricted approximation hinges on the assertion that directional flux density is only weakly dependent on angle (i.e. is linearly anisotropic).

Henry [ibid.] gives an alternate derivation of the $P_1$ three-dimensional approximation to the transport equation. Here spherical harmonics are used to handle scattering angular dependence, and a weighted residual technique is used with a trial function

$$\gamma_t(r, \Omega, E) = \phi(r, E) + 3\Omega \cdot J(r, E)$$

chosen as an analogue of a one-dimensional analysis with $P_1$ approximations. The weighting functions $\frac{1}{\sqrt{3}}$ and $\Omega$ lead to the same coupled equations as those based entirely on spherical harmonics (other weighting functions of course produce a different approximation). The $P_1$ approximation to the transport equation is
\[ \nabla \cdot J(r,E) + \Sigma_t(r,E)\phi(r,E) = \int_0^\infty dE' [\Sigma_f J'(E) + \Sigma_f J'(r,E')] + \Sigma_{s0}(r,E' \rightarrow E)\phi(r,E') \] (2.4.1)

and

\[ \frac{1}{3} \nabla \phi(r,E) + \Sigma_t(r,E)J(r,E) = \int_0^\infty dE' \Sigma_{s1}(r,E' \rightarrow E)J(r,E') . \] (2.4.2)

It is conventional to reduce the two \( P_1 \) approximation equations (2.4.1, 2.4.2) to a single continuous energy diffusion equation by transforming the second equation into a relationship of the form

\[ J(r,E) = -D(r,E) \nabla \phi(r,E) , \]

where \( D(r,E) \) is a diffusion coefficient. Such a relationship is analogous to Fick's law (which is fundamental to ordinary diffusion theory). With this relationship, the coupled \( P_1 \) approximations are reduced to a single equation. When anisotropic scattering and the form of equation (2.4.2) are considered, selection of a simple diffusion constant \( D(r,E) \) is impossible, except under some rather restrictive assumptions. Instead the diffusion coefficient must be replaced by an operator or a functional depending upon \( J(r,E) \) itself.

Equation (2.4.2) can be written as three separate scalar equations
\[ \frac{\partial}{\partial \mu} \phi(r, E) = -3 \left[ \Sigma_t(r, E) J_u(r, E) - \int_0^\infty dE' \Sigma_{s1}(r, E' \rightarrow E) J_u(r, E') \right] \]

\[ \equiv -3 A_1 J_u(r, E) , \quad (u=x,y,z) \]

and a diffusion constant operator \( \tilde{D} \equiv -\frac{1}{3} A_1^{-1} \), defined such that

\[ J_u(r, E) = -\tilde{D} \frac{\partial}{\partial \mu} \phi(r, E) \]

or

\[ J(r, E) = -\tilde{D} \nabla \phi(r, E) \cdot \]

Consequently (2.4.1) becomes

\[ -\nabla \cdot (\tilde{D} \nabla \phi(r, E)) - \Sigma_t(r, E) \phi(r, E) = \int_0^\infty dE' \left[ \Sigma^{j}_{f}(E) \nu^{j}_{f} \Sigma^{j}_{s}(r, E') \right] + \Sigma_{s0}(r, E' \rightarrow E) \phi(r, E') \cdot \quad (2.4.3) \]

As an alternative to the use of an operator D, a functional form may be employed. Three functions

\[ \Sigma_{tr,u}(r, E) \equiv \Sigma_t(r, E) - \frac{\int dE' \Sigma_{s1}(r, E' \rightarrow E) J_u(r, E')}{J_u(r, E)} \]

are defined, as are three components of a directional diffusion constant.
for $u=x, y$ or $z$. In this form the three components of $D_u(r,E)$ are generally different and (2.4.1) becomes

$$-\nabla \cdot D(r,E) \nabla \phi(r,E) - \Sigma_t(r,E) \phi(r,E)$$

$$= \int dE' \left[ \Sigma_f^j(E) v^j \Sigma_f^j(r,E') + \Sigma_s^0 (r,E' \rightarrow E) \right] \phi(r,E') .$$

Equations (2.4.3) and (2.4.4) are formally equivalent to the $P_1$ formulation (2.4.1) and (2.4.2) but are more complicated than the usual continuous energy diffusion equation derived directly from diffusion theory. The forms (2.4.3) and (2.4.4) can be simplified by means of multigroup and few-group approximations.

2.5 REDUCTION OF CONTINUOUS ENERGY TO GROUP DIFFUSION THEORY MODELS

The energy range is divided into subintervals

$$\Delta E_g = E_{g-1} - E_g , \quad (g=1,2,\ldots,G)$$

where $E_0$ represents the highest energy considered (approximately 15 MeV). Group fluxes $\phi_g(r)$ and group currents $J_{gu}(r)$ are defined over these subintervals by
\[
\phi_g(r) = \int_{E_g}^{E_g-1} \phi(r,E) \, dE ,
\]

(2.5.1)

and

\[
J_{gu}(r) = \int_{E_g}^{E_g-1} J_u(r,E) \, dE .
\]

(2.5.2)

Group constants are defined analogously:

\[
E_{ag}(r) \equiv \frac{\int_{E_g}^{E_g-1} \Sigma_\alpha(r,E) \phi(r,E) \, dE}{\phi_g(r)} \quad (\alpha = t, f) ,
\]

(2.5.3)

and

\[
A_{gg',u} \equiv \int_{E_g}^{E_g-1} \int_{E_g}^{E_g-1} \int dE' \Sigma_0(r,E') \phi(r,E') \, dE' \, [\delta_{E'}(r,E) - \delta(E' - E)]
\]
The $P_1$ equations (2.4.1) and (2.4.2) are integrated over $\Delta E_g$ to give the $P_1$ energy group equations

$$\nabla \cdot J_g(r) + \Sigma_t g(r) \phi_g(r) = \frac{G}{g'_{=1}} \sum_{j=1}^{G} \sum_{g'=1}^{G} \Sigma_{gg'}(r) \phi_g(r)$$

and

$$\frac{1}{3} \frac{\partial}{\partial t} \phi_g(r) = - \sum_{g'_{=1}}^{G} A_{1gg',u(r)} J_{g',u(r)} (g=1,2,\ldots,G).$$

The energy group form of the operator $\tilde{D}$ is obtained by writing (2.5.5) in matrix form

$$\frac{1}{3} \frac{\partial}{\partial t} [\phi(r)] = - [A_{1,u(r)}][J_{u(r)}] ,$$

where the notation $[\ ]$ denotes a matrix; $[\phi(r)]$ is a diagonal matrix whose components are column vectors

$$\{ \phi_1(r), \phi_2(r), \ldots, \phi_G(r) \} ;$$

$[J_{u(r)}]$ is a diagonal matrix whose components are column vectors.
\{J_{1u}(r), J_{2u}(r), \ldots, J_{Gu}(r)\};

\[[A_{1,u}(r)]\] is a GxG full matrix with elements \{A_{1gg',u}(r)\}. The GxG diffusion-constant matrices \[[\tilde{D}_u(r)]\] are immediately defined by

\[[\tilde{D}_u(r)] = \frac{1}{3} [A_{1,u}(r)]^{-1},\]

and the elements of \[[\tilde{D}_u(r)]\] denoted by \(\tilde{D}_{gg',u}\) \((g'=1,2,\ldots,G; g=1,2,\ldots,G)\). Consequently (2.5.6) provides a vector equation for each element of \[[J_u(r)]\]

\[J_{gu}(r) = - \sum_{g'=1}^{G} \tilde{D}_{gg',u}(r) \frac{\partial}{\partial u} \phi_{g'}(r) \quad (g=1,2,\ldots,G)\]

and (2.5.4) leads to a set of energy group diffusion equations

\[- \sum_{g'=1}^{G} \left[ \nabla \cdot \tilde{D}_{gg',u}(r) \nabla \right] \phi_{g'}(r) + \Sigma_{g'}(r) \phi_{g'}(r)\]

\[= \sum_{g'=1}^{G} \left[ \sum_{j} \chi_{g^j} \gamma_{j} \Sigma_{g^j}(r) + \Sigma_{gg'}(r) \right] \phi_{g'}(r) \quad (g=1,2,\ldots,G), \quad (2.5.7)\]

where the \(\{D_{gg'}\}\) components display directional dependence.

The energy group analogue of (2.4.4) follows in much the same way:
\[-\nabla \cdot D_g(r) \nabla \phi_g(r) + \Sigma_{rg}(r) \phi_g(r) = \]

\[= \sum_{g'=1}^{G} \left[ \sum_{j} \chi_{g,j} \gamma_{j} \Sigma_{fg',g}(r) + \Sigma_{gg',g}(r) \right] \phi_{g'}(r) \quad (g=1,2,\ldots,G), \quad (2.5.8)\]

where

\[\Sigma_{tr,g,u}(r) \equiv \frac{\sum_{g'=1}^{G} A_{g,g',u}(r) J_{g'} u(r)}{J_{g} u(r)}, \quad (2.5.9)\]

and

\[D_{gu}(r) \equiv \frac{1}{3 \Sigma_{tr,g,u}(r)}\]

and the removal cross section \(\Sigma_{r}(r)\) is introduced as

\[\Sigma_{rg}(r) = \Sigma_{tg}(r) - \Sigma_{gg}(r),\]

where now \(\Sigma_{gg}\) is set to zero (on the RHS of (2.5.8) and in the later equations). Both forms (2.5.7) and (2.5.8) are considerably more complicated than the conventional group diffusion theory equation. This is because both \(\tilde{D}_{gg'}(r)\) and \(D_g(r)\) exhibit directional dependence, and because of the additional leakage terms in equation (2.5.7). Simplification is possible through multigroup and few-group approximations.
In multigroup theory, approximations for $\phi(r,E)$ and $J_u(r,E)$ are expressed as separable functions of the two independent variables:

$$\phi(r,E) = \phi_g(r) F_{0g}(E)$$

and

$$J_u(r,E) = J_{gu}(r) F_{1g}(r)$$

for energy $E$ in the interval $\Delta E$ and for all $r$. $F_{0g}(E)$ and $F_{1g}(E)$ are some normalised preselected appropriate shapes, independent of direction. Because $F_{1g}(E)$ is independent of direction, the elements of $A_{1gg',u}$ are the same for all $u$ and consequently $[\tilde{D}_u(r)]$ is also independent of $u$. This provides some simplification in the leakage terms of (2.5.7), although the coupling of leakage terms still remains. No such simplification for (2.5.8) exists, so one either needs to determine $\Sigma_{t,rg,u}(r)$ iteratively, or to make some appropriate assumption about $J_{g',u}(r)$. Either choice is equally unattractive.

To reach a more satisfactory situation, a form of asymptotic transport theory is used to find a spectrum function that is appropriate deep inside a region composed entirely of the material present at $(r)$. In a Fourier transform approach to solving the transport equation, $\gamma(r,\omega,E)$ is expressed.
\[ (r,\Omega,E) = F(\Omega,E)e^{i\beta \cdot r} \quad (2.5.10) \]

With spherical harmonics it is possible to determine the \( P_0 \) and \( P_1 \) components of the asymptotic spectrum (\( F_{0g}(E) \) and \( F_{1g}(E) \) respectively). The last expression in (2.5.3) for \( A_{1gg'},u \) with material \( k \) becomes

\[ A_{1gg'}^k = \frac{\int_{E_g}^{E_g'-1} dE \int_{E_g}^{E_g'-1} dE' \left[ \Sigma_t^k(E) \delta(E'-E) - \Sigma_s^k(E'\rightarrow E) \right] F_{1g}(E')}{\int_{E_g}^{E_g'-1} dE' F_{1g}(E')} \quad (2.5.11) \]

where \( F_{1g}^k(E') \) is the \( P_1 \) component of the asymptotic spectrum associated with material \( k \). Directional dependence is now lost because \( F_{1g}(E) \) is independent of \( u \). Although (2.5.11) is expressed as though it is continuous in energy, it is the result of a multigroup approximation involving a set of multigroup fluxes \( F_{1n}^k(n=1,2,\ldots,N; N \gg 0) \).

Expressed in terms of a few groups (\( G \)), (2.5.11) becomes

\[ A_{1gg'}^k = \sum_{n \in g} \Delta E_n \sum_{n' \in g'} \left[ \Sigma_t^{k,n'} \delta_{n'n'} - \Sigma_s^{k,n';n} \right] F_{1n'}^k \quad (2.5.12) \]

where the sums are over the fine groups \( n,n' \) contained in the coarse groups \( g,g' \). Few-group expressions similar to (2.5.3) for
\( \sum_{g}^{k} \gamma_{g} \), and \( \chi_{gg}^{k} \) may be obtained. A direction independent diffusion constant matrix with components \( \tilde{D}_{gg}^{k} \) arises from (2.5.12). The result is a few-group approximation of identical form to the multigroup equation (2.5.7), except that directional dependence of the components \( \{D_{gg'}^{k}\} \) is eliminated. The form still involves leakage coupling between groups \( g \) and \( g' \).

The more conventional \( G \) group diffusion equations arise from (2.5.8), after the assumption is made that \( J_{g'u}(r) \) and \( J_{gu}(r) \) may be expressed as portions of the \( P_{1} \) spectrum \( F_{1}^{k}(E) \). Because \( F_{1}^{k}(E) \) is independent of direction, \( D_{g}(r) \) is also directionally independent and in few-group representation

\[
\Sigma_{tr,g}^{k} = \sum_{n}^{N} \Delta E_{n} \sum_{n'=1}^{N} \left[ \sigma_{t}^{k} \delta_{nn'} - \sigma_{s}^{k} \delta_{nn'} \right] F_{1}^{k} \sum_{n}^{N} F_{1}^{k} \tag{2.5.13}
\]

Consequently with the new \( D_{g}(r) \), equation (2.5.8) appears as the more conventional form of the diffusion approximation.

Examination of the two few-group approximation forms of (2.5.7) and (2.5.8) reveals that the simpler leakage formulation of (2.5.8) is easier to employ in a numerical study. The more implicit formulation (2.5.7) appears unattractive when it is recalled that the standard practice for iteratively handling the group coupling of these equations is a Gauss-Seidel technique.
The approximations involved in deriving the simpler form (2.5.8) involves estimation of the energy shape of $J_u(r,E)$ over the entire energy spectrum, whereas for (2.5.7) the approximation is over the more limited range $\Delta E_g$. Ordinarily the error is not considered to be as significant as it first appears and for reasons of numerical convenience the standard practice is to solve the simpler form of group diffusion equation (2.5.8) only. It is essential when using diffusion theory in modelling neutron behaviour to ascertain that such theory indeed is applicable for the problem at hand, and that the group constants have been approximated appropriately.

To set the aspect of nuclear reactor calculation which this work primarily is concerned with in perspective, an outline of the procedure for determining the neutron flux $\phi(r,E,t)$ (the quantity of primary interest) is given. The overall process is a three stage procedure with possible iterative refinement (Figure 2.2)

**Stage 1:** The spatial and time dependence is largely ignored and an energy spectrum calculation for $\phi(E)$ is performed. The accuracy required of the estimate is reflected in the group structure chosen. Sometimes analytic approximations across fairly wide energy bands will suffice; at other times (particularly due to resonance phenomena), extremely fine group structures are required. It is generally necessary to give some consideration to the spatial dependence of the flux in generating the group constants. This can be achieved by calculating the neutron spectrum for a typical cell in the reactor.
Three stage determination of $\phi(r, E, t)$

Figure 2.2
core lattice. The spectrum depends on the type of reactor and the conditions under which it is operated. The calculations should be repeated in each region of the reactor where the material composition shows significant variation. A typical code used here is MIRANDA of the AUS scheme [Robinson 1975 and 1977] which is used for stage 2.

Stage 2: The multigroup constants are determined by the formulae given in (2.5.3).

Stage 3: In reality, because the number of groups is restricted, (between 2 and 20 depending upon the reactor type), the cross section data is collapsed by approximations such as (2.5.12). Group constants must be determined for different regions, and redetermination may be necessary where the materials change due to burnup or capture. Hence spectrum calculations, and group constant calculations may be required at frequent intervals in reactor design studies.

The third stage requires solution of the few group equation (2.5.8) (in later chapters referred to as a multigroup equation to distinguish it from one-group (single speed) models. As discussed in Section 2.3 for transport equations, non-zero solutions exist for (2.5.8) under special circumstances only. To ensure a non-trivial solution at all times, a positive real number \( k \) is introduced to adjust artificially the number of neutrons produced in the fission process. This gives the steady state eigenvalue (or criticality) equation
\[-\nabla \cdot d_g(r) \phi_g(r) + \Sigma_{rg}(r) \phi_g(r) = \sum_{g'=1}^{G} \left[ \Sigma_{g'j} \left( \Sigma_{jg'}(r) + \Sigma_{gg'}(r) \right) + \Sigma_{g'}^{fg}(r) \right] \phi_{g'}(r) \quad (g=1, 2, \ldots, G)\]

and when \( k \) is exactly one the configuration is critical. In later treatment of this equation only one fissionable isotope is assumed for simplicity. (An appropriate averaging process is included in some codes which then gives the appearance of one fissionable isotope.)

This thesis is devoted to the third stage of the reactor calculation process (viz. the detailed geometry few-group calculation). A three-dimensional workhorse code POW3D [Barry, Harrington and Pollard 1982] has been developed to this end, but in addition the code serves as a vehicle for testing out some novel ideas to speed up the calculations of Stage 3. In addition, it incorporates some other recent developments reported elsewhere in the numerical literature for the purpose of comparative testing. POW3D is a member of the AUS nuclear package [Robinson 1975], and as such is designed to fit automatically into the global process as described below.

The AUS package, like other nuclear code systems is set up to solve the 'simpler' version of the group diffusion equation (2.5.8), rather than the coupled leakage form (2.5.7). In Chapter 4 a novel scheme known as MINI is described for solving linear systems arising from finite difference discretisations. MINI revels in making the equations more implicit and works well in instances where more
commonly used schemes are not rigorously applicable. MINI would appear
good suited to overcoming some of the numerical disadvantages inherent
in tackling the group coupled leakage terms of (2.5.7). Such testing
is beyond the scope of this work, and it does not lend itself to
inclusion in the present form of the AUS program suite.
They climbed the trees ... As was told before,
The Glugs climbed trees in the days of yore,
When the oldest tree in the land to-day
Was a tender little seedling - Nay,
This climbing habit was old, so old
That even the cheeses could not have told
When the past Glug people first began
To give their lives to the climbing plan.
    And the legend ran
That the art was as old as the mind of man.

C.J.Dennis, 'The Glugs of Gosh'.

3.1 TIME DEPENDENT MULTIGROUP NEUTRON DIFFUSION EQUATIONS

In Chapter 2 the broad outline for derivation of the steady state neutron diffusion equation (2.5.14) from transport theory was given. In this chapter, the numerical techniques for solving the diffusion approximation to the transport equations, as implemented in the nuclear code POW3D are discussed.

In addition to solving the static (or steady state) problem in three spatial dimensions, POW3D was envisaged as a vehicle for undertaking time dependent studies. Consequently we begin from the time dependent (kinetic) multigroup diffusion equations (3.1.1 and 3.1.2) and follow Stacey [1969]. Despite the ability of the code to handle a limited form of kinetic problems, most of the work so far undertaken with POW3D has been directed to steady state studies. For this reason and for simplicity of presentation, the derivation of the time dependent equations was purposefully omitted from Chapter 2.
The time dependent neutron diffusion equations are

\[- \frac{1}{V_g} \frac{\partial}{\partial t} \phi_g(r,t) = - \nabla \cdot D_{n,g}(r,t) \nabla \phi_g(r,t) + \sigma_{rg}(r,t) \phi_g(r,t) - \sum_{g'} \sigma_{g'g}(r,t) \phi_{g'}(r,t) - \sum_{g'} \chi_{g'}(1 - \beta) \frac{\nu}{k} \sigma_{fg}(r,t) \phi_g(r,t) - \sum_d \chi_d \lambda_d C_d(r,t) - S_g(r,t), \quad (g=1,2,\ldots,G) \]

(3.1.1)

and those for the delayed neutron precursor concentrations are

\[ \frac{\partial}{\partial t} C_d(r,t) = \beta_d \sum_g \frac{\nu}{k} \sigma_{fg}(r,t) \phi_g(r,t) - \lambda_d C_d(r,t), \quad d=1,2,\ldots,D, \]

(3.1.2)

where

- \( k \) is the effective steady state multiplication, which has been determined in a criticality calculation (left hand sides of (3.1.1) and (3.1.2) set to zero as also the source term \( S_g \)).

- \( \phi_g(r,t) \) is the group g flux, to be calculated in all studies,

- \( C_d(r,t) \) is the precursor concentration density for the \( d^{th} \) delayed group, also calculated as an intermediate stage in
obtaining $\phi_g(r,t)$,

$v_g$ denotes average velocity for energy group $g$,

$D_{n,g}(r,t)$ denotes possibly (tensor) directional diffusion coefficients (for directions $\eta$ or $-\eta$ parallel to the chosen axes, see Sections 2.4 and 2.5), although the isotropic value $D_g(r,t) = \frac{1}{3\Sigma_{tr,g}(r,t)}$ is usually used. To avoid confusion with summation, the practice of representing all macroscopic cross sections such as $\Sigma_{tr,g}$ with $\Sigma_{tr,g}$ will be used wherever possible throughout the remainder of this work,

$\sigma_g(r,t)$ denotes various (macroscopic) cross sections

$\sigma_{gg'} = \text{scattering cross section (matrix) from groups } g' \text{ to } g$

$\sigma_{rg} = \text{removal cross section from group } g,$

$= \sigma_{ag} + \sum_{g' \neq g} \sigma_{g'g} \text{ (absorption + outscatter from group } g). \text{ The self scatter term } \sigma_{gg} \text{ is set to zero.}$

$\sigma_{fg} = \text{fission cross section for group } g,$

$\nu = \text{the average total number of neutrons emitted per fission}$
\( \chi_{pg} \) is the prompt fission spectrum (normalised to a unit group sum), (note \( \sum_{g} (1-\beta) \frac{1}{k} \sigma_{fg} \) is a matrix however),

\( \chi_{dg} \) is the delayed fission spectrum (normalised to a unit group sum) for the delayed group d,

\( \beta_{d} \) is the fraction of all fission neutrons emitted in delayed group d, note \( \beta (= \sum_{d} \beta_{d}) \),

\( S_{g}(r,t) \) is the source strength in group g of an external source located at a point \( r \) of the reactor,

\( t \) is the time,

\( \lambda_{d} \) is the time constant for the delayed group decay.

The summations are taken over all groups \((g=1,2,...,G; \ d=1,2,...,D)\).

The equations are solved subject to:

(i) the outer boundary conditions -

(a) reflective (zero current), frequently used when reactor symmetry is applicable
\[ n \cdot \nabla \phi_g(r,t) = 0 \quad , \quad (3.1.3) \]

where \( n \) denotes the outward pointing normal, or

(b) zero flux at the extrapolated boundary

\[ d \cdot n \cdot \nabla \phi_g(r,t) + \phi_g(r,t) = 0 \quad , \quad (3.1.4) \]

where \( d \) = extrapolation distance, that is, the distance outside the medium at which the flux would vanish. \( d \) is sometimes taken to be \( \frac{2}{\Sigma_{tr}} \) where \( \Sigma_{tr} \) is the transport cross section, although more rigorous theory leads to \( d = \frac{0.71}{\Sigma_{tr}} \). (See Weinberg and Wigner [ibid.] for further discussion.);

(ii) the internal boundary conditions - for the material interfaces which are assumed to lie parallel to the axes. For left and right hand boundaries these are

(a) the continuity of flux

\[ \phi_g(r,t) \bigg|_L = \phi_g(r,t) \bigg|_R \quad \text{for each group} \ g \quad , \quad \text{and} \quad (3.1.5) \]

(b) continuity of current

\[ (D_n, g(r,t)) \cdot n \cdot \nabla \phi_g(r,t) \bigg|_L - (D_n, g(r,t)) \cdot n \cdot \nabla \phi_g(r,t) \bigg|_R = 0 \quad (3.1.6) \]

for each group \( g \), where \( n \) is normal to the boundary;
(iii) the initial conditions; the reactor may be operating at a steady state power level or be starting from a shutdown position -

(a) steady state conditions

\[ \frac{\partial}{\partial t} \phi_g(r,t) = 0, \quad \frac{\partial C_d}{\partial t} (r,t) = 0, \quad S_g(r,t) = 0, \quad t \leq 0 \text{ for all groups}, \]

\[ \sum_g \int f(r,0) \phi_g(r,0) \, dr = P(0), \]

where \( f(r,t) \) is the energy released from each fission of the material about \( r \), and \( P(0) \) is the required power level at time zero (the start of the study). To satisfy this condition, an eigenvalue problem (i.e. the steady state form of (3.1.1)) is solved to obtain the group fluxes \( \phi_g \). The steady state eigenvector is normalised to produce the appropriate power level \( P_0 \).

(b) shutdown conditions

\[ \phi_g(r,t) = 0, \quad C_d(r,t) = 0, \quad S_g(r,t) = 0, \quad t \leq 0 \text{ for all groups.} \]

In addition to solving the time-dependent equations (3.1.1 and 3.1.2) POW3D is designed to solve other problems, including the steady state eigenvalue (or criticality problem) either in its own right, or as the first stage of a kinetics calculation. The steady state eigenvalue problem is given by
where \( \chi_g = \chi_{pg}(1-\beta) + \sum_d \chi_{dg}\beta_d \). The eigenvalue \( \frac{1}{k} \) is introduced to allow non-trivial solutions, and its significance is as discussed in Section (2.3) for the transport equation.

POW3D also will solve the steady state external source calculation (3.1.9) and its time dependent equivalent.

The adjoint flux is of considerable assistance in undertaking perturbation studies of reactor systems. The adjoint equivalent to the eigenvalue equation (3.1.8) is

Note that the multigroup diffusion equation is not self-adjoint due to the scattering and fission production cross sections. The diffusion and removal terms, however, do constitute a symmetric system.
In addition, the code is capable of performing three types of criticality searches. For these, physical attributes are adjusted by a multiplicative parameter, to achieve a required effective multiplication \( k = k_{\text{required}} \), usually 1. Three types of searches are supplied with POW3D, but the sophisticated user may specify his own requirements, by writing special subroutines which are interfaced easily with the code.

Existence and uniqueness of solutions to equations (3.1.1 and 3.1.2) are assumed throughout this work. For a discussion on the mathematical basis behind this see Habetler and Martino [1961]. Froehlich [1968], gives a more general proof for the steady state case, while Varga [1961] outlines the mathematics of their solution.

Solution of the time dependent equations is by no means easy, and analytic approaches are out of the question for studies involving detailed geometries or configurations consisting of many materials. Analytic solutions for simple models, however, can provide further insight into the mathematical understanding of the problem, provided appropriate simplifications are possible. The analytic solutions then obtained are invaluable for testing numerical procedures at early stages of code development.

One such simplification of the equations involves the separation of variables

\[ \phi(r,t) = \Phi(r) T(t) . \]  

(3.1.11)

The validity of this is often assumed in safety and kinetics studies [Glasstone and Edlund 1952] and it leads eventually to the point
reactor kinetics model. Equation (3.1.7) is a convenient mathematical approximation without rigorous mathematical or physical justification. Despite one alarming problem, [Yasinsky and Henry 1965] the separation is thought to yield suitable approximations to reactor transients when a detailed knowledge of regional flux variation is not required. Analytic solution of the point kinetics equations is possible for certain types of reactivity variation. Non-linear algebraic equations arise, however, within the 'inhour equation'. These frequently require reversion to numerical techniques so the necessary parameters can be found. For a discussion on the use of such kinetic models in studying reactivity, feedback, and reactor stability see Hetrick [1971]. Even for point kinetics studies, analytic technique are generally inadequate when reactivity changes are given by a constantly changing function, or when non-linearities arise.

Analytic solutions for the spatial problem involving neutron sources and non-multiplying media are obtained readily for certain simple geometries involving a one group model. Solutions for criticality problems with fissile materials may be obtained with a little more effort. For example, the analytic solution of a homogeneous reactor is obtained for simple geometries with elementary techniques. Just surrounding this reactor with a reflective material requires a more sophisticated approach [Garabedian 1962]. For more than one group, a further separation of variables (space and energy) may be attempted. The basic textbooks on reactor physics treat such approaches extensively. (Considerable attention has been devoted to two-group studies, e.g. Meem [1964], because they have more chance of analytical treatment, and therefore were appropriate in the early days of reactor studies when computers were still in their infancy, and because they provide a reasonable introduction to the whole subject.)
Once the number of materials, the geometrical complexity of the structure, or the number of energy groups, starts increasing the analytic approach breaks down and some form of numerical solution is mandatory.

3.2 CHOICE OF A FINITE DIFFERENCE NUMERICAL APPROACH

The time dependent and independent forms of the multigroup diffusion equation may be solved by various numerical schemes. A fairly recent survey of current techniques is given by Adams [1977]. The reasons behind the choice of the time honoured finite difference technique, despite attractions inherent in other approaches is discussed here.

The finite element method has had remarkable success in the solution of partial differential equations arising in several fields of mathematical physics. The technique arose from generalisations of frame analysis studies, in engineering design. Its first application to continuum mechanics was not surprisingly in stress related studies (Turner et al [1956]). Its acceptance in non-engineering studies was at first slow, perhaps due to initial absence of mathematical respectability. Mathematicians by and large ignored the method until its utility was established in numerous engineering applications. The publication by Strang and Fix [1973] indicated its eventual wide acceptance by applied mathematicians, while Ciarlet [1978] denotes a higher point in its mathematical abstraction.

The alternative approach through the finite element technique was considered when work on the code POW3D commenced, but was rejected because of insufficient evidence as to its superiority (if any).
Kaper et al. [1972], Kang and Hansen [1973] and Hansen and Kang [1975] suggest the method may be attractive for reactor physics, while Barry [1974a and 1974b] reports success with Hermite bicubic approximating functions for similar elliptic problems. The main attractions of a finite element approach appear to be

(i) the ease with which irregular geometries are handled,

(ii) the automatic handling of the internal (non-essential) boundary conditions by the variational approach [Mikhlin 1964],

(iii) the ease with which continuously varying cross sections are accommodated,

(iv) possible improvements from the use of dispersed rather than point total (removal) cross sections,

(v) the better approximations possible at corners and interfaces [Babuška and Kellog 1972].

In the final analysis, however, a working three-dimensional code was required, and caution suggested the finite difference scheme be employed. Uncertainty over the properties of matrices arising in finite element approximations and the applicability of the new iterative schemes (then in embryo form), to them removed any doubt about the choice. This was supported by Young [1975] who indicated the need for further investigation into the performance of such schemes. This was particularly appropriate at a time when direct solution processes were very much the commonplace option for finite
element procedures. It appeared such schemes would not be appropriate for the extremely large systems envisaged [Hageman 1975]. It was anticipated that should MINI, the new iterative technique described in this work fail, then relaxation techniques, although somewhat slower than may be desired, at least would serve as an effective backup with the finite difference approach.

Some recent work on conjugate gradient methods [Concus and Golub 1975 and Concus et al. 1976] could have suggested an alternative approach, but was disregarded, due to disappointing experiences with the original conjugate gradient method in neutron diffusion studies [Barry, Paine, Pollard 1975]. In that study, a traditional version was implemented to handle the matrices generated by the older two-dimensional code POW [Pollard 1974], and it performed excellently on homogeneous models, as did the implementation by Reid [1972] for two Laplacian operator problems. For even fairly simple reactor models, such as Moata (Chapter 8), this level of performance was not repeated, and relaxation schemes appeared far preferable. At the time this project commenced, the work of Meijerink and van der Vorst [1977] and Kershaw [1977] was unknown. A considerable time elapsed (while the new iterative technique (MINI) was developed and implemented in a two-dimensional code), before the decision was made to again use the conjugate gradient method with its recent refinements. Initially the decision was prompted by Doherty [1977] and Eastwood [1977]. A general purpose conjugate gradient subroutine [Doherty ibid.] was incorporated in POW. It demonstrated the advantages of using the approximate inverse to correct the solution estimate after every conjugate gradient iteration. The subroutine was too general,
however, to be the main solution driver in a nuclear code. Consequently two subroutines, specially designed for neutron diffusion studies, were developed and implemented for two and three-dimensional geometries in POW3D. In both instances, the conjugate gradient schemes were introduced after the relaxation and MINI schemes were operational.

It was decided to ignore developments in more advanced methods of reactor analysis (cf. synthesis methods, Henry [1975]) as unsuitable (due to subjective choices that are necessary) for the general purpose code.

3.3 **FINITE DIFFERENCE DISCRETISATION OF THE MULTIGROUP DIFFUSION EQUATION**

Methods for discretisation of partial differential equations abound in numerical analysis textbooks. The details of discretisation of the multigroup diffusion equation consequently are delayed until the appendices, where they are included primarily for completeness. The method of time discretisation, however, may be of more interest for this particular problem (not covered in standard texts) and is summarised in Appendix A. The resulting difference scheme is given by equation (A.10). Various schemes to solve the time dependent neutron diffusion problem are available Hansen [1967]. The time dependent problem is not computationally trivial, however, because of widely varying time constants inherent in the physics.

The spatial discretisation is described in Appendix B for a single energy group equation in three-dimensions. The seven-point edge flux method (as opposed to the centre mesh flux) is used because it gives greater accuracy in approximating the diffusion operator. (The only non-standard aspect involves satisfying the internal
boundary conditions.) With the fine details as given in equations (B3.6, B3.10 and B3.11) the multigroup form of the approximation for the time dependent diffusion equation (3.1.1) is

\[ g_{ijk;p}^1 g_{ijk+1;p} + g_{ijk;p}^2 g_{ijk+1k;p} + g_{ijk;p}^3 g_{ijk-1k;p} \\
+ g_{ijk;p}^4 g_{ijk-1k;p} + g_{ijk;p}^5 g_{ijk+1k;p} + g_{ijk;p}^6 g_{ijk-1k;p} \\
+ \sum_{\ell} g_{ijk;p}^7 + \sum_{\ell} \left[ \sigma_{rg}(m_{\ell}, \bar{t}_p) + 2/(v g d t) \right] v_{ijk} \left\{ g_{ijk;p}^8 \right\} g_{ikj;p} \\
- \sum_{\ell} \left[ \sigma_{gg'}(m_{\ell}, \bar{t}_p) + \chi(2)(\delta t) \frac{\nu}{k} \sigma_{fg'}(m_{\ell}, \bar{t}_p) \right] v_{ijk} g_{ikj;p} \\
= - g_{ijk;p-1}^1 g_{ijk+1;p-1} - g_{ijk;p-1}^2 g_{ijk+1k;p-1} \\
- g_{ijk;p-1}^3 g_{ijk-1k;p-1} - g_{ijk;p-1}^4 g_{ijk-1k;p-1} \\
- g_{ijk;p-1}^5 g_{ijk+1k;p-1} - g_{ijk;p-1}^6 g_{ijk-1k;p-1} \\
- \sum_{\ell} \left[ \sigma_{rg}(m_{\ell}, \bar{t}_p) - 2/(v g d t) \right] v_{ijk} \left\{ g_{ijk;p-1}^8 \right\} g_{ikj;p-1} \\
+ \sum_{\ell} \left[ \sigma_{gg'}(m_{\ell}, \bar{t}_p) + \chi(1)(\delta t) \frac{\nu}{k} \sigma_{fg'}(m_{\ell}, \bar{t}_p) \right] v_{ijk} g_{ikj;p-1} \\
+ \sum_{d} \chi_d g_{\lambda_d} \delta t \frac{d}{d} g_{ijk;p-1} + g_{ijk}(\bar{t}_p) \right\} (3.3.1)

and for (3.1.2) is

\[ d C_{ijk;p} = d C_{ijk;p-1} C^{-\lambda_d \delta t} + \delta t \sum_{\ell} \sigma_{fg'}(m_{\ell}, \bar{t}_p) \]
where \( g=1,2,\ldots,G; \) \( d=1,2,\ldots,D \) \( \phi_{ijk;p} \) is the sought after group \( g \) flux solution taken as an average for a mesh box about a point \((i,j,k)\) at time \( t_p \), \( dC_{ijk;p} \) and \( gS_{ijk}(\bar{t}_p) \) respectively are volume integrals of precursor concentrations and external sources over a mesh box.

As a general purpose code, POW3D, handles problems with a 0,1,2 and 3 spatial dimensions. Except for the three-dimensional form some modifications to the difference equations (3.3.1) may be necessary to approximate leakage in the omitted direction(s). The missing term in the leakage operator is approximated by

\[
-\frac{\partial}{\partial u} \cdot D_u \frac{\partial}{\partial u} \phi(r) = D_u B_u^2 \phi(r) ; \quad u=x,y,z ;
\]

where \( B_u^2 \) is the geometric buckling. (For a homogeneous rectangular parallelepiped with extrapolated boundaries given at \( a, b \) and \( c \), the full buckling is \( (\pi/a)^2 + (\pi/b)^2 + (\pi/c)^2 \), Duderstadt and Hamilton [1976].) The corresponding flux profile is \( \cos(\pi x/a) \cos(\pi y/b) \cos(\pi z/c) \). To allow for the leakage in the omitted direction(s) \( u \), the removal cross sections \( \sigma_{rg}(m_z,t_p) \) in equation (3.3.1) are adjusted

\[
\sigma_{rg}(m_z,\bar{t}_p) = \sigma_{rg}(m_z,\bar{t}_p) + \frac{B_u^2}{3\sigma_{tr} g(m_z,\bar{t}_p)} .
\]

For historical reasons POW3D is also designed to support two-dimensional \((r,z)\) geometry calculations, but details of the discretisation are not given here. The solution of the resulting linear system of equations is undertaken in an identical fashion to an \((x,y)\) geometry calculation.
At the heart of a numerical solution of the time dependent
difference equations is solution of a large system of linear equations

\[ A \phi(p) = S, \]  

(3.3.2)

where \( S \) is given by the right hand side of (3.3.1) i.e. it depends
upon quantities like \( \phi^{(p-1)} \), \( \xi^{(p-1)} \) and any external source. \( A \) is a
matrix representing the left hand side of (3.3.1). Note the
eigenvalue \( \frac{1}{K} \) and the fundamental eigenvector appearing in the kinetics
equation (3.3.1) are determined in a criticality calculation which is
undertaken before the time dependent solution is commenced.

3.3.1 The Steady State Case

The steady state eigenvalue (or criticality) problem can be
expressed in matrix terms as

\[ M\phi = \frac{1}{K} F\phi, \]  

(3.3.3)

where \( M\phi \) is a finite difference approximation (obtained as detailed in
Appendix B) for

\[-\nabla \cdot D_{ng}(r) \nabla \phi_{g}(r) + \sigma_{rg}(r) \phi_{g}(r) - \sum_{g} \sigma_{gg'}(r) \phi_{g'}(r),\]
and $F_{x}$ is the equivalent approximation for

$$\chi_{g} \sum_{k} \frac{1}{k} \sigma_{fg'}(r) \phi_{g'}(r) \quad (g=1,2,\ldots,G; \ g'=1,2,\ldots,G) .$$

The matrices for the time dependent and eigenvalue problems are related closely through

$$A = M - F \ , \quad (3.3.4)$$

providing the temporal components of $A$ are ignored. The equation (3.3.4) appears in more detailed form as

$$\begin{align*}
\begin{bmatrix}
-vD_{1} + \sigma_{r1} & -\sigma_{12} & -\sigma_{13} & \cdots \\
-\sigma_{21} & -vD_{2} + \sigma_{r2} & -\sigma_{23} & \\
-\sigma_{31} & -\sigma_{32} & -vD_{3} + \sigma_{r3} & \\
\vdots & \vdots & \vdots & \\
\end{bmatrix}
\begin{bmatrix}
\phi_{1} \\
\phi_{2} \\
\phi_{3} \\
\vdots \\
\phi_{G}
\end{bmatrix}
\end{align*}$$

$$= \frac{1}{k} \begin{bmatrix}
vx_{1} \sigma_{f1} & vx_{1} \sigma_{f2} & \cdots \\
vx_{2} \sigma_{f1} & vx_{2} \sigma_{f2} & \\
vx_{3} \sigma_{f1} & vx_{3} \sigma_{f2} & \\
\vdots & \vdots & \\
\end{bmatrix}
\begin{bmatrix}
\phi_{1} \\
\phi_{2} \\
\phi_{3} \\
\vdots \\
\phi_{G}
\end{bmatrix} \quad (3.3.5)$$
The matrices $M$ and $F$ need not be as full as shown. For fast reactor studies the upscattering terms $\sigma_{ij}^\cdot j > i$ are zero, while for thermal reactors $\chi_g = 0$ in the lower groups. When $M$ is lower triangular, the matrix equations (in the group sense) are explicit, so far as the numerical procedure normally employed to solve them is concerned (i.e. no upscatter passes are necessary). For thermal reactor studies, not all $\sigma_{ij}^\cdot j > i$ are zero and it is common to have extensive upscattering from many thermal groups.

For the diagonal submatrices of $M$, in (3.3.3)

$$M_{gg} = -\nabla \cdot D_g \nabla + \sigma_{rg} \quad (g=1,2,\ldots,G),$$

the following important mathematical properties [Wachspress 1966] apply:

(i) $M_{gg}$ is irreducible, \hfill (3.3.6)

(ii) $m_{ij} \leq 0$ for all $i \neq j$, \hfill (3.3.7)

(iii) $m_{ij} = m_{ji}$, \hfill (3.3.8)

(iv) $m_{ii} \geq \sum_{j \neq i} |m_{ij}|$, \hfill (3.3.9)

with strict inequality for at least one $i$, and column diagonal dominance then follows from condition (iii).
For the complete matrix $M$ of (3.3.3), $\sigma_{ij} \neq \sigma_{ji}$ (i\neq j); consequently, there is no symmetry relation equivalent to (3.3.8). Row diagonal dominance (3.3.9) no longer applies. The matrix, however, is column diagonally dominant

\[ m_{ii} \geq \sum_{j \neq i} |m_{ji}|. \]  

(3.3.10)

This may be verified by considering the removal cross section of an arbitrary diagonal component of $M$. The removal cross section

\[ \sigma_{rg} = \sigma_{ag} + \sum_{g \neq i} \sigma_{g'g} \]  

(3.3.11)

involves two components, the absorption and out-scattering cross sections respectively. Summation of the column terms of $M$, (corresponding to the $g^{th}$ diagonal element) leaves only the absorption term which is positive, hence column diagonal dominance follows immediately.

The existence and uniqueness of a positive solution for equation (3.3.3), with a corresponding real eigenvalue greater in modulus than all other eigenvalues of the matrix equation has been demonstrated by the application of Perron-Frobenius theory of non-negative matrices [Birkhoff and Varga 1958].

Solution of the eigenvalue matrix equation involves the determination of the largest modulus eigenvalue and its corresponding eigenvector. The simplest approach to this is through the power method [Wachspress 1966] where
\[ \phi^{(n+1)} = \frac{1}{k(n)} M^{-1}F\phi(n), \]

\[ \frac{1}{k(n+1)} = \frac{1}{k(n)} \frac{\langle w, M^{-1}F\phi(n) \rangle}{\langle w, M^{-1}F\phi(n+1) \rangle}, \]  

(3.3.12)

for an arbitrary weighting vector \( w \). The required eigenvalue \( |k_i| > |k_j|, i=2,3,...,N_xN_yN_zG \).

If the initial estimate of the eigenvector corresponding to this eigenvalue, \( \phi_1^{(0)} \), is written in terms of the eigenvectors of \( M^{-1}F \)

\[ \phi_1^{(0)} = \sum a_i \phi_i, \]

then after \( n \) iterations with (3.3.12), the approximation \( \phi_1^{(n)} \) to \( \phi_1 \) is of the form

\[ \phi_1^{(n)} \approx c_1[\phi_1 + \sum_{p=2}^{NG} c_p (k_{2p})^n \phi_p], \]

where \( c_1 \) and \( c_2 \) are constants. The rate of convergence of the iterative scheme for eigenvector and eigenvalue iterations is determined by the dominance ratio \( \frac{k_2}{k_1} \) (where \( k_2 \) is the second highest (modulus) \( k \)-eigenvalue of \( M^{-1}F \)).

Convergence is slow for most problems, and is notoriously so for those with a high dominance ratio. For large reactors this is typically of the order of 0.95 or larger implying very slow
convergence. Fast reactors are more compact and have lower dominance ratios (Ferguson and Derstine [1977]). Even so these may be as high as 0.9.

Because convergence is slow, extrapolation is normally employed to accelerate convergence. In practice it is more attractive to extrapolate the neutron fission sources rather than the flux. This is achieved by multiplying both sides of the eigenvalue equation (3.3.3) by $FM^{-1}$. The theory of Chebyshev extrapolation is based on the assumption that all eigenvalues of $FM^{-1}$ are real. These assumptions are not necessarily true. In practical situations, however, experience shows that extrapolation really works for most problems [Fladmark 1972]. In POW3D the approach is identical to that adopted in POW, where weighted averages of three earlier fission sources are used.

Determination of the largest eigenvalue by the Chebyshev accelerated extension of (3.3.12) requires calculation of

$$M^{-1} F \phi (n-1) \ ,$$

(3.3.13)

while solution of the time dependent source problem (3.3.2) requires calculation of

$$A^{-1} \bar{S} \ .$$

(3.3.14)
Properties (3.3.6, 3.3.9 and 3.3.10) are sufficient to demonstrate \( M^{-1} > 0 \), and hence solutions are positive. Of course \( M^{-1} \) is never computed, and iterative techniques normally are used to solve the linear systems of equations. The row diagonal dominance of \( M \) is lost for the multigroup form but column diagonal dominance remains. Hence the Gauss-Seidel technique is the standard method to date for solution of the expression (3.3.13), because lack of symmetry precludes the calculation of optimal relaxation parameters in iterative processes. (Solution for 3.3.13 is carried out by block techniques. Fortunately, however, more sophisticated techniques are available for the inner blocks.) The new iterative technique MINI (to be introduced in Chapter 4), does not require symmetry and provides an appropriate and novel approach for hastening convergence of the operation (3.3.13).

3.3.2 Back To Kinetics

Unfortunately, even the property of column diagonal dominance is not guaranteed for the time dependent problems (3.3.14). The diagonal terms are modified by

\[
\sum_{\ell} \left[ \frac{2}{(v_d^\delta t)} - \chi_{(2)g} \frac{v}{k} \sigma_{fg}(m_{\ell}, \bar{t}_p) \right] v_{ijk}^\ell,
\]

where \( v_{ijk}^\ell \) denotes the volume of the 'box' surrounding the \( ijk^{th} \) mesh point. The negative off-diagonal terms are each increased (in a negative sense) by

\[
\sum_{\ell} \chi_{(2)g} \frac{v}{k} \sigma_{fg}'(m_{\ell}, \bar{t}_p) v_{ijk}^\ell.
\]
Consequently \( \sum_{i \neq j} |a_{ij}| \) differs from \( \sum_{i \neq j} |m_{ij}| \) by

\[
\left| \sum_{k} \left[ \frac{2}{v_{g} \delta t} - \chi(2) g^{\frac{v}{k}} \sigma_{fg}(m_{2}, \bar{e}_{p}) \right] v^{k}_{ijk} \right| - \left| \sum_{k} \chi(2) g^{\frac{v}{k}} \sigma_{fg} \right|
\]

Although the absorption component of the total cross section contains a fission removal component \( \sigma_{fg} \), it is insufficient to ensure that the solution to (3.3.2) is positive or that the iterative process for (3.3.14) even converges. The mathematical properties (3.3.6, 3.3.9 and 3.3.10) fortunately are sufficient rather than necessary conditions, and practical experience has shown the solution is always positive and the Gauss-Seidel method convergent. For the kinetics case, it would be possible if necessary, to adjust \( \delta t \) to ensure diagonal dominance, but the situation for fissile source problems is a little more difficult because there is no time dependent term to strengthen the diagonal. Once again, practical studies show no problems in obtaining a converged result for stable configurations.

3.4 COMPUTATIONAL STRUCTURE OF POW3D

The code POW3D is designed to function as shown in the following description. The schematic representation embodies concepts of programming and mathematics, to give the overall flow of logic. Not all the concepts involved have yet been explained, in later chapters these will be defined (MINI Chapters 4, 5; ICCG Chapter 5; energy group and region rebalance Chapter 6). Some of the stages look
remarkably simple, such as calculating the source components. This is rather misleading, as experience showed the time savings obtained by reworking this section were as great as those obtained by employing better iterative procedures. This section of the code has been worked over so many times that it is no longer possible to recognise the original physical intent of the calculation from the code. Here one discovers the dilemma, of choosing between efficient code, or well structured, self documenting and understandable instructions. Recent trends in computer science (Dahl, Dijkstra and Hoare [1972]) would come down emphatically in terms of the latter, however, the former is chosen in POW3D because the time savings are just so enormous that the range of problems handled by POW3D is enhanced greatly.
procedure main

\( t = t_0 \)

calculate \( t_p \), cross sections and coefficient matrix

if kinetics calc. then \( R_g = \text{rhs eqn (3.3.1)} \) for \( g=1,2,\ldots,G \)

if source calc. then \( R_g = \text{external source} \) for \( g=1,2,\ldots,G \)

else \( R_g = 0 \) for \( g=1,2,\ldots,G \)

generate trial fluxes \( \phi_g \); \( g=1,2,\ldots,G \)

calculate fission source \( F = \sum_{g' \neq g} \sigma_{gg'} \phi_g \)

if eigenvalue calc. then renormalise fluxes to given power level

if not kinetics then \( p = 0 \)

loop kinetics from 1 to \( p \) \( (t_f = t_0 + \sum_{p=1}^{p} \delta t_p) \)

loop outer = 1 to maximum outers

calculate number of upscatter passes (upmax)

*** POSSIBLE COARSE MESH SOLUTION OF OUTER CALCULATION ***

if joint group-space rebalance then begin collapse

\( \hat{\phi}_g \rightarrow \phi_g, \hat{M}_{gg'} \rightarrow M_{gg'} \)

\( \hat{S} \rightarrow S \); \( g=1,2,\ldots,G; g'=1,2,\ldots,G \)

call OUTER (\( \hat{M}, \hat{\phi}, \hat{S}, \text{upmax}, \text{coarse mesh} \))

end

*** FINE MESH CALCULATION OF OUTER CALCULATION ***

CALL OUTER (\( M, \phi, S, \text{upmax}, \text{finemesh} \))

calculate fission source \( F = \sum_{g} \sigma_{fg} \phi_g \); \( g=1,2,\ldots,G \)

if eigenvalue calc. then calculate \( k \)

if converged then finish for eigenvalue and source calcs.
if search and partial convergence then adjust parameters and calculate necessary matrix terms

extrapolate $F$ with Chebyshev method

if eigenvalue calc. then renormalise $\phi_g, g=1,2,\ldots,G$

end loop outer

update precursor concentration

t = t + \delta t

adjust parameters for kinetics variations

recompute necessary coefficient matrix terms procedure OUTER ($M, \phi, S, upmax, mesh$)

end loop kinetics

procedure OUTER ($M, \phi, S, upmax, mesh$)

loop $\uplambda$ from 1 to $upmax$

g_1 = 1, g_f = G, \Delta g = 1

if $\uplambda \neq 1$ then $g_1 = up$ (first group with upscatter into it)

if adjoint then $g_1 = G, g_f = 1, \Delta g = -1$

loop from $g = g_1$, to $g_f$, step $\Delta g$

if ($g = up$ and $\uplambda = 1$ and convergence difficult, mesh fine)

then apply energy rebalance $\rho_g, g'=1,2,\ldots,G$

if MINI SOURCE then $S_g = R_g + X_g F + \Sigma g' \sigma gg' \phi_g'$

$= M_{gg} + \Sigma g' \sigma gg' \phi_g'$

else $S_g = R_g + X_g F + \Sigma g' \sigma gg' \phi_g'$ (GS source)
if (convergence difficult and method ≠ SLOR and mesh fine)
    then region rebalance \( \phi_g \)

if (convergence difficult and method = SLOR and \( \omega \) determined and mesh fine)
    then region rebalance \( \phi_g \)

if (coarse mesh option)
    then solve \( M_{gg} \phi_g = S_g \) by direct means

if (fine mesh option) then
    solve \( M_{gg} \phi_g = S_g \) by SLOR, MINI or ICCG, or combinations of same

end loop \( g \)
end loop \( \nu_{up} \)
end procedure OUTER
CHAPTER 4

METHOD OF IMPLICIT NON-STATIONARY ITERATION (MINI) AND OTHER ITERATIVE APPROACHES

'A soldier on the march is hemmed in and borne along by his regiment as much as a sailor is by his ship. However far he has walked, whatever strange, unknown and dangerous places he reaches, just as a sailor is always surrounded by the decks, masts and rigging of his ship, so the soldier always has around him the same comrades, the same sergeant-major, the same Company dog, and the same commanders.'

L. Tolstoy, 'War and Peace'.

4.1 INTRODUCTION

The solution of the multigroup, multidimensional, time dependent neutron diffusion equation is broken down into the solution of a number of intermediate layers, as outlined in Chapters 2 and 3. At the heart of the solution procedure for all problems, there is an inner equation, which for a single energy group is basically the elliptic equation

\[- \nabla \cdot (D \nabla \phi) + \sigma_r' \phi = s, \quad (4.1.1)\]

where \( \sigma_r' \) denotes the removal terms. Elsewhere in this work, equation (4.1.1) is referred to frequently as a source problem, to distinguish it from the eigenvalue form (3.3.3). The discretisation procedure employed (Appendices A and B), leads to a set of linear equations
where $A$ is a sparse, real, irreducible, symmetric, positive definite (i.e. Stieltjes) block tridiagonal matrix with elements

$$a_{ij} = a_{ji} \quad i=1,2,...,N; j=1,2,...,N,$$

$$a_{ij} \leq 0 \quad i \neq j,$$  \hspace{1cm} (4.1.3)

$$a_{ii} \geq -\sum_{j \neq i} a_{ij} \quad \text{(with inequality for at least one } i \text{)}$$

and source elements which are non-negative (and not all zero),

$$s_i \geq 0.$$  \hspace{1cm} (4.1.4)

It can be shown that $A^{-1}$ has no negative elements [Varga 1961] and consequently all elements of the unknown flux

$$\phi = A^{-1} s$$  \hspace{1cm} (4.1.5)

are positive, in keeping with physical reality. The elements of $A$ are assumed to be formed by a natural ordering of successive points, as described in Chapter 3. For the two-dimensional set of equations, an iterative procedure based on solving one line at a time is used commonly, while for three-dimensional problems, a two-layer iterative scheme was anticipated. This approach is described fully in (Sections
5.5 and 5.6); however, for ease of presentation in this chapter, point methods are discussed first and then extended to the more commonly used forms.

Perhaps the simplest effective procedure for solving equation (4.1.2) is the Gauss-Seidel (GS) iterative scheme (4.1.6). For familiarity with conventional matrix treatments, $x$ will be used instead of $\phi$ to represent unknown quantities,

$$
x^{(n)}_i = (s_i - \sum_{j=1}^{i-1} a_{ij} x^{(n)}_j - \sum_{j=i+1}^{N} a_{ij} x^{(n-1)}_j) / a_{ii}, \quad (4.1.6)
$$

$i = 1, 2, \ldots, N$.

The index $n$ is an iteration counter, while all improperly formed sums are taken as zero. Because of the properties (4.1.3) the GS approach takes any trial solution $x^{(0)}$ and updates this so that $x^{(n)}$ converges to $x$, the required solution [Faddeeva 1959]. If all the elements of the trial solutions are non-negative

$$
x^{(0)}_i > 0, \quad i = 1, 2, \ldots, N, \quad (4.1.7)
$$

then all elements of subsequent iterative solutions are also non-negative,
4.4

\[ x_i^{(n)} > 0, \quad i=1,2,\ldots,N; \quad n=1,2,\ldots, \]

because each component of equation (4.1.6) is positive. This is in keeping with the matrix properties of \( A \) (4.1.3). Although this positive preserving property is not essential for an iterative process, it is considered to be desirable, and MINI was specifically formulated to enforce it. (MINI in its most refined form includes special tests to guard against the somewhat unlikely, but possible, situation where lack of numerical precision can lead to a zero or negative right hand side in equations similar to (4.1.6)).

The GS method is slow for reactor problems, where \( N \) is large (10,000 or more) and the largest magnitude eigenvalue, \( \lambda_{GS} \), of the iteration matrix is quite close to 1. Convergence is enhanced considerably by a simple extension, Successive Over-Relaxation (SOR). The procedure is

\[
x_i^{(n)} = \omega [s_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(n)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(n-1)}
\]

\[ - a_{ii} x_i^{(n-1)} (\omega - 1)/\omega ]/a_{ii} \quad i=1,2,\ldots,N \quad , \quad (4.1.8)\]

where the optimum extrapolation parameter is given as

\[
\omega_{opt} = 2/[1 + \sqrt{1 - \lambda_{GS}}] \quad , \quad [\text{Young 1954}] \quad (4.1.9)\]

and \( \lambda_{GS} \) is the largest (absolute value) eigenvalue of the GS point
iteration matrix. From equation (4.1.8) it is seen, that unlike the GS method, SOR may produce negative intermediate solutions, but this has not inhibited its use.

Methods based on successive over-relaxation are common in reactor codes. Generally they involve the solution for a block of unknowns rather than individual points. The block is often constructed from a line of mesh points within the original grid which produces a tridiagonal system lending itself to efficient direct solution. It may be extended to involve groups of lines, having a faster rate of convergence [Varga 1960], but more direct computational overheads. Instead of lines, methods involving far more intricate patterns of grid points are possible [Evans and Benson 1967], (or even simpler, red-black ordering, which has its own attractions).

A few basic results for iterative schemes are presented. These have some bearing on MINI, and will be used again when coarse mesh rebalance is discussed in Chapters 6 and 7.

For an iterative scheme of the form

$$\tilde{x}^{(n)} = M \tilde{x}^{(n-1)} + g$$

(4.1.10)

the error vector $e^{(n)}$ after $(n)$ iterations is defined as
4.6

\[ e(n) = x(n) - \hat{x}, \quad (4.1.11) \]

where \( \hat{x} \) is the exact solution satisfying \( \hat{x} = M\hat{x} + g \) (and of course \( A\hat{x} = s \)). Substituting (4.1.11) into (4.1.10) gives

\[ e(n) = M^n e(0). \quad (4.1.12) \]

For simplicity of presentation, consider the iteration matrix \( M \) does not have an eigenvector deficiency. The eigenvectors then form a complete set, and the initial error may be expanded in terms of basis vectors \( u \) by

\[ e(0) = \sum_{i=1}^{N} a_i u_i, \]

which on substitution into (4.1.12) leads to

\[ \|e(n)\| = \left\| \sum_{i} a_i \lambda_i^{(n)} u_i \right\| \leq \rho^n \|e(0)\|, \]

where \( \lambda_i \) are the eigenvalues of the iteration matrix and \( \rho \) is the spectral radius of \( M \) defined as

\[ \rho = \max_{i} |\lambda_i|, \quad (i=1,2,...,N). \]

With the limitation that \( M \) is not deficient in its eigenvalue structure, the error reduction for the iterative scheme may be studied
directly in terms of the different eigenvalues and eigenvectors, provided they are identifiable. For the Jacobi scheme it is possible to demonstrate the eigenvalues occur in plus and minus pairs, and for very simple matrix operators, to identify the eigenvalues and their corresponding eigenvectors. Once the switch is made to a relaxation scheme such analysis does not apply. The relaxation factor ($\omega$) is related to the eigenvalues of the Jacobi and SOR iterative matrices by

\[
(u + \omega - 1)^2 = u \omega^2 \lambda^2 ,
\]

[Young 1971], where $\lambda$ and $u$ are the eigenvalues of the Jacobi and SOR matrices respectively. Consequently

\[
\mu_i^\pm = \frac{\omega^2 \lambda_i^2}{2} - \omega + 1 \pm \omega \left| \lambda_i \right| \sqrt{\frac{\omega^2 \lambda_i^2}{4} - \omega + 1} ,
\]  
(4.1.13)

so the eigenvalues $\mu_i$ may be real or complex depending upon $\omega$, and the optimal value of $\omega$ from (4.1.13) satisfies the relation (4.1.9). The behaviour of the eigenvalues $\mu_i$ in terms of $\omega$ is best shown graphically, as in Figure 4.1, for SOR ($1 < \omega < 2$). The eigenvalues $\mu_i$ are shown for a suboptimal value of $\omega$. Some of the eigenvalues are real; for each real eigenvalue $\mu_i^-$ within the circle of radius $\omega - 1$ there is a corresponding $\mu_i^+$ lying outside; while complex pairs are distributed around the circle [Young 1971]. As $\omega$ is increased for any real pair $\mu_i^+$ and $\mu_i^-$, the circle is expanded, $\mu_i^-$ increases while $\mu_i^+$ decreases until for $\omega$ sufficiently large both eigenvalues meet. Further increases in $\omega$ cause them to separate again and move around
Distribution of SOR eigenvalues on the complex plane

(Similar to Young [1971])

Figure 4.1
the complex circle. When $\omega = \omega_{\text{opt}}$, as given by (4.1.9), all eigenvalues lie on the circle and the spectral radius of the SOR iterative matrix is

$$\mu(\text{SOR}) = \omega_{\text{opt}} - 1 .$$

All eigenvalues of the SOR matrix are of equal magnitude. It is no longer possible to identify neatly each eigenvector (even if it were known) with a corresponding eigenvalue.

The extrapolation parameter $\omega$, usually is determined by some earlier GS iterations [Faddeeva 1959], or iterations performed with an underestimate for $\omega$, using analysis of two successive iterations [Wachspress 1966] or three successive iterations [Pollard 1975]. POW3D uses the latter when the SLOR option is requested in place of the default method (MINI).

The numerical estimate $\omega$ for $\omega_{\text{opt}}$ is likely to be in error by a small amount. The spectral radius of the SOR iteration matrix is sensitive to the choice of $\omega$, the typical variation is of the form given in Figure 4.2, where an underestimate is more harmful than an overestimate. Consequently, the numerical algorithm [Pollard 1973], now employed in POW3D is designed to over-estimate, rather than underestimate $\omega$. 
Variation of the spectral radius of SOR iterative matrix with $\omega$ (similar to Varga [1962])

Figure 4.2
It may take as many as 30 non-optimum time independent iterations before the optimum value of $\omega$ is determined. For reactor source and non-search type criticality problems, a succession of calculations are performed, where the matrix $A$ (and hence $\omega$) are not altered because changes in the neutron flux can affect only the source, (the outer iterative loop) and estimates of particular group fluxes (the upscatter iterative loop). Hence there is no need to repeat the initial effort of estimating $\omega_{opt}$.

When ICCG is employed as an alternative solution strategy in POW3D the same observation applies to the necessity of recomputing the partial $\text{LDL}^T$ decomposition (see Chapter 5).

For kinetics and search type criticality problems on the other hand, material cross sections may be changing with each time or search iteration step, requiring re-evaluation of $\omega$ (or $\text{LDL}^T$), a not insignificant overhead, particularly for $\omega$. To avoid recalculation of $\omega$, a method was sought that required little memory of previous iterations. To drive the three-dimensional kinetics code efficiently, the method should also require as few iterations as possible.

Essentially the approach adopted arises from observations of the improvement the GS technique (4.1.6) offers over the Jacobi method.
\[ x_i^{(n)} = (s_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(n-1)}) - \sum_{j=i+1}^{N} a_{ij} x_j^{(n-1)}) / a_{ii}. \]

The traditional improvements to GS have sought explicit adjustments in terms of the current point, or points already updated in the current iterative pass. These are basically one parameter methods, although two parameter techniques Hadjidimos [1980] are possible.

If the second term \( \sum_{j=i+1}^{N} a_{ij} x_j^{(n-1)} \) in the GS method (4.1.6) could be updated to reflect what is happening at the \( n^{th} \) iteration, it seems reasonable to anticipate a further improvement in the rate of convergence. Consequently, the thrust is to make the term not updated yet in the GS method implicitly dependent on the unknown presently being sought, that is \( x_i^{(n)} \). The simplest way to do this is with the term

\[ \sum_{j=i+1}^{N} a_{ij} x_j^{(n-1)} \frac{x_i^{(n)}}{x_i^{(n-1)}}, \]

(4.1.14)

and numerical experimentation showed this often hastened convergence, but the method is unreliable, because sometimes the process converged to an incorrect result. The implicit idea, however, has been successfully applied in accelerating convergence of the alternating direction implicit method, with implicit buckling correction, ADI-B\(^2\) [Hageman and Yasinsky 1969]. An alternative to the simple multiplying
factor \( \frac{\text{x}_i^{(n)}}{\text{x}_i^{(n-1)}} \) was then sought in this work.

Implicit methods based on Stone's [1968] strongly implicit method have been devised [Wang 1978], but these are very different to the MINI approach used here.

4.2 POINT MINI APPROACH

The essence of the implicit approach used with MINI, is to introduce extrapolating parameters, \( \gamma^{(n)}_{ij} \), one for every non-zero matrix element \( a_{ij} \), with \( j > i \), so that a better estimate of \( x_j \) is obtained than \( x_j^{(n-1)} \). It is possible to replace \( x_j^{(n-1)} \) in the GS method (4.1.6) by the term

\[
x_j^{(n-1)} + \gamma^{(n)}_{ij} (x_i^{(n)} - x_i^{(n-1)}) \frac{x_j^{(n-1)}}{x_i^{(n-1)}}
\]

however, the simpler combination

\[
x_j^{(n-1)} + \gamma^{(n)}_{ij} (x_i^{(n)} - x_i^{(n-1)})
\]

is found to be superior in practice. Our MINI equations are then

\[
\sum_{j=1}^{i} a_{ij} x_j^{(n)} + \sum_{j=i+1}^{N} a_{ij} [x_j^{(n-1)} + \gamma^{(n)}_{ij} (x_i^{(n)} - x_i^{(n-1)})] = s_i
\]

\[
i = 1, 2, \ldots, N
\]

(4.2.1)
from which we obtain the explicit relation for $x_i^{(n)}$

$$x_i^{(n)} = \frac{[s_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(n)} - \sum_{j=i+1}^{N} a_{ij} (x_j^{(n-1)} - y_j^{(n)} x_i^{(n-1)})]}{a_{ii} + \sum_{j=i+1}^{N} a_{ij} y_j^{(n)}}$$

(4.2.2)

\[i=1,2,...,N\]

where again improperly formed sums are taken as zero, and the rules for computing the $y_j^{(n)}$ are yet to be defined. To satisfy the goal that all intermediate solutions are positive (assuming positive initial estimates) the following stringent, but not absolutely necessary condition may be applied to the extrapolating parameters

$$0 \leq y_j^{(n)} < \min(1, x_j^{(n-1)}/x_i^{(n-1)}) \quad (4.2.3)$$

When we do this, and on account of the negativity of off-diagonal terms, the effective diagonal term is reduced in (4.2.2) compared with the GS approach. The sufficiency of condition (4.2.3) may be verified by inspection of (4.2.2), and in this work the extrapolating parameters are required to satisfy at least (4.2.3). Writing the MINI iterative procedure in terms of the iteration increments
\[ \delta_i(n) = x_i(n) - x_i(n-1) , \quad (4.2.4) \]

and the iteration residuals

\[ R_i(n) = s_i - \sum_{j=1}^{i-1} a_{ij} x_j(n) - \sum_{j=i+1}^{N} a_{ij} x_j(n-1) , \]

gives

\[ \delta_i(n) = R_i(n) / (a_{ii} + \sum_{j=i+1}^{N} a_{ij} \delta_j(n)) , \quad i=1,2,...,N. \]

For comparison the GS and SOR iteration increments are

\[ \delta_i(n) (GS) = R_i(n) / a_{ii} \]

and

\[ \delta_i(n) (SOR) = \omega \delta_i(n) (GS) . \quad (4.2.5) \]

A point relationship similar to (4.2.5) can be written to express the MINI iteration increment in terms of the GS increment

\[ \delta_i(n) = \omega \delta_i(n) (GS) , \quad (4.2.6) \]

where
\[ \omega_i^{(n)} = \frac{a_{ij}}{a_{ii} + \sum_{j=1}^{N} a_{ij} \gamma_{ij}^{(n)}}. \] (4.2.7)

From the restriction on \( \gamma_{ij}^{(n)} \) given by (4.2.3), the increment extrapolation parameters \( \omega_i^{(n)} \) are bounded

\[ 1 \leq \omega_i^{(n)} \leq \frac{a_{ij}}{a_{ii} + \sum_{j=1}^{N} a_{ij}}. \] (4.2.8)

The upper limit on the SOR extrapolation parameter \( \omega \) in (4.2.5) is 2, however, there is no such upper bound on the MINI form of the extrapolation parameter in (4.2.8). MINI can consequently take big leaps for some iterations (and it usually does during early stages).

4.3 ESTIMATION OF MINI EXTRAPOLATION PARAMETERS

Rules for determining the \( \gamma_{ij}^{(n)} \) are formulated. The procedures evolved throughout the 'experimental mathematics' stage of the project. The impetus to consider a graphics approach came from the convergence pattern for the simple cobweb model of market stabilisation [Lipsey 1972]. This coupled with the availability of a pleasing interactive FORTRAN system [Clancy 1978], the ability to write fully interactive FORTRAN programs [Cawley and Trimble 1977], and the availability of a friendly graphics package [Trimble 1978] decided the approach. Without the use of computer graphics it is doubtful whether MINI would ever have been refined, and probably would have remained just another interesting idea. Eventually a short
motion picture was produced. This vividly demonstrated the ability of MINI to move in leaps and bounds to the correct solution. Its production was facilitated by software and hardware developed at the AAEC's research establishment. These facilities permitted the computer to drive the movie-camera from the mathematical subroutines written in FORTRAN [Turner 1982].

The rules for computing $\gamma_{ij}^{(n)}$ are established using three criteria:

(i) the MINI extrapolation parameters are assumed to be varying slowly between iterations

i.e. $\gamma_{ij}^{(n)} \approx \gamma_{ij}^{(n-1)}$ ,

(from hereon $\gamma_{ij}^{(n)}$ will imply normally an estimate based on results from the previous iteration - a non-linear iterative process of estimation is not assumed),

(ii) to be bounded

$0 \leq \gamma_{ij}^{(n)} \leq \min(1, x_j^{(n)}/x_i^{(n)})$ ,

(iii) to provide the simple extrapolation
\[ x_j^{(n)} = x_j^{(n-1)} + \gamma_{ij}^{(n)} \delta_j^{(n)} . \]

The third criterion may conflict with the second at times, so the third is extended to accommodate the second thus

\[ x_j^{(n-1)} + \gamma_{ij}^{(n)} \delta_j^{(n)} = p_{ij}^{(n)} x_j^{(n)} + (1-p_{ij}^{(n)}) x_j^{(n-1)} , \quad p_{ij}^{(n)} \in [-1,1] \]

with \( p_{ij}^{(n)} = 1 \) if possible, otherwise it is reduced to meet the requirement imposed through the second criterion. For all cases

\[ \gamma_{ij}^{(n)} = \frac{p_{ij}^{(n)} \delta_j^{(n)}}{\delta_i^{(n)}} . \]

For \( p_{ij}^{(n)} = 1 \) the idea is straightforward, in other cases a resort to an heuristic approach is necessary. Sometimes a large ratio of \( \delta_j^{(n)}/\delta_i^{(n)} \) arises, when the solution at the point \( i \), unlike the rest of the solution is almost correct - maybe the trial solution was nearly correct here (or a very small incremental change in the solution occurred). It is then preferable to use a small magnitude value instead of the large one calculated, because what is happening at the point \( i \) cannot really be used to predict what is going to happen at the point \( j \). In these instances, an 'upended' ratio is used instead, and this also tends to maintain a degree of continuity as \( n \) changes for values near 1. In summary, the value \( p_{ij}^{(n)} \) is such that

\[ \gamma_{ij}^{(n)} = \begin{cases} \frac{|\delta_j^{(n)}/\delta_i^{(n)}|}{|\delta_j^{(n)}/\delta_i^{(n)}|} & \text{if } |\delta_j^{(n)}/\delta_i^{(n)}| \leq 1 \\ |\delta_i^{(n)}/\delta_j^{(n)}| & \text{otherwise} \end{cases} \]  

(4.3.1)
then \( y_{ij}^{(n)} = \begin{cases} \frac{y_{ij}^{(n)}}{x_{i}^{(n)}} & \text{if } 0 < x_{i}^{(n)}/x_{i}^{(n)} \\ x_{j}^{(n)}/x_{i}^{(n)} & \text{otherwise} \end{cases} \) (4.3.2)

except that \( y_{ij}^{(0)} = 0 \), (a GS iteration) begins the process.

The additional computation beyond the GS and SOR approaches is not negligible; however, with the line and plane extensions for two and three dimensions respectively, the overhead is tolerable (Sections 4.8 and 4.9). The additional storage over SOR required for MINI is a temporary array for the iteration increments \( \delta_{i}^{(n)} \), \( i=1,2,...,N \).

4.4 CONSISTENCY OF MINI

Following Young [1971], let \( J(A,s) \) denote the set of solutions of (4.1.2) and \( J(I-G,k) \) represent the solutions of the related linear system

\[(I-G)x = k\]

formed from the linear stationary iterative method

\[x^{(n)} = Gx^{(n-1)} + k\] (4.4.1)

Then the iterative method defined by (4.4.1) is consistent with the system (4.1.2) if \( J(A,s) \subseteq J(I-G,k) \), reciprocally consistent if \( J(I-G,k) \subseteq J(A,s) \) and completely consistent if both consistency and reciprocal consistency (i.e. \( J(A,s) = J(I-G,k) \)) apply.
It will be demonstrated first, that if the $\gamma_{ij}^{(n)}$ are frozen (i.e. do not vary from iteration to iteration), then the stationary linear system arising from MINI of the form (4.4.1) is completely consistent. The matrix $A$ may be expressed

$$A = L + B + U,$$

where $B$ is a diagonal matrix, $L$ and $U$ represent lower and upper triangular forms respectively. Writing the general stationary relation (4.4.1) in terms of the MINI definition (4.2.2) we obtain

$$x^{(n)} = -(L+B+\Gamma)^{-1}(U-\Gamma)x^{(n-1)} + (L+B+\Gamma)^{-1}z^{(n-1)} + (L+B+\Gamma)^{-1}z,$$  \hspace{1cm} (4.4.2)

where $\Gamma$ is also a diagonal matrix whose components are

$$\Gamma_{ii} = \sum_{j=1}^{N} a_{ij} \gamma_{ij}.$$  \hspace{1cm}Young [1971] establishes that provided (4.1.2) is solvable (i.e. $A$ is non-singular) then (4.1.2) is completely consistent with (4.4.1) if and only if a non-singular matrix $M$ exists such that

$$G = I - MA,$$  \hspace{1cm} (4.4.3)

and

$$k = M z.$$  \hspace{1cm} (4.4.4)

It can be verified that providing $(L+B+\Gamma)$ is non-singular, a matrix $M$ exists such that
and this $M$ satisfies conditions (4.4.3) and (4.4.4). The second condition is immediately obvious from (4.4.2). To establish the first (4.4.3), consider

$$G = -(L+B+\Gamma)^{-1} (U-\Gamma) ,$$

multiplication of both sides by $M^{-1}$ gives

$$M^{-1} G = -(U-\Gamma)$$

$$= (L+B+\Gamma) - (L+B+U)$$

$$= M^{-1} - A .$$

Multiplying both sides of the transformed result by $M$ gives

$$G = I - MA ,$$

and the result is established, provided $M$ is non-singular. With $\chi_{ij} < 1$ (for all $i,j$) this holds, in keeping with conditions (4.1.3). For the type of matrix $A$ considered in the neutron diffusion problem (Chapter 3) this restriction can be extended to include the strict inequality $\chi_{ij} < 1$. 
Now consider a non-stationary iterative method, where at each step the relation

$$\chi^{(n)} = G_n \chi^{(n-1)} + k_n, \quad n \geq 1$$  \hspace{1cm} (4.4.5)

applies. After n such steps

$$\chi^{(n)} = G_n \chi^{(0)} + \hat{k}_n, \quad n \geq 1$$  \hspace{1cm} (4.4.6)

where

$$G_n = G_n G_{n-1} G_{n-2} \cdots G_1$$

$$\hat{k}_n = k_n + G_n k_{n-1} + G_n G_{n-1} k_{n-2} + \cdots + G_n G_{n-1} \cdots G_2 k_1.$$

The non-stationary iterative form (4.4.6) is satisfied by MINI, where $\chi^{(n)}(j>i,i=1,2,\ldots,N)$ may vary from iteration to iteration. The proof that the non-stationary form of MINI is completely consistent has not been achieved yet, at best one can demonstrate that it satisfies Young's definition of consistency for non-stationary iterative methods. The method (4.4.6) is defined to be consistent with (4.1.2) if for $\chi^{(n)} \in \int(A_1\mathcal{S})$ then $x^{(n)}* \in \int(A_1\mathcal{S})$ for all $n* \geq n$.

The consistency of (4.4.6) follows immediately from the theorem of Young [1971] stated here without proof.
Theorem. If the non-stationary method (4.4.6) is obtained by the use of (4.4.5) and if for each fixed value of $j=1,2,...$ the linear stationary method defined by

\[ x^{(n)} = G_j x^{(n-1)} + k_j \quad n \geq 1 \]

is consistent, then the method (4.4.6) is consistent.

A formal analysis of the fore-runner to MINI, given by (4.1.14) is difficult because of the non-linear nature of the iteration process, but the lack of consistency was discovered in numerical experimentation.

4.5 CONVERGENCE OF MINI

A proof of convergence for MINI is yet to be found. Numerical experimentation shows that MINI has never failed to converge for matrices with properties (4.1.3), and frequently is convergent (even if only slowly), for matrices when these conditions are relaxed significantly. A summary is given, however, of what has been shown for MINI.

From the MINI equations (4.2.1) and the definition (4.2.4), the iterative process for iteration increments may be expressed in matrix form as
\[ Q(n) \xi(n) = z - A \xi(n-1) \quad , \] (4.5.1)

where

\[
Q(n) = \begin{bmatrix}
    a_{11} + \sum_{j=2}^{N} a_{1j} \gamma_{i_j}^{(n)} & 0 & 0 & \cdots & 0 \\
    a_{21} & a_{22} + \sum_{j=3}^{N} a_{2j} \gamma_{i_j}^{(n)} & 0 & \cdots & 0 \\
    \vdots & \vdots & \ddots & \ddots & \vdots \\
    a_{N1} & a_{N2} & a_{N3} & \cdots & a_{NN}
\end{bmatrix}
\]

(i.e. \( Q(n) = L + \Gamma(n) \), extending earlier notation). Now

\[ Q(n-1) \xi(n-1) = z - A \xi(n-2) \quad , \]

which when subtracted from equation (4.5.1) gives

\[ \xi(n) = T(n) \xi(n-1) \quad , \]

where the MINI iteration matrix is
\[ T(n) = [Q(n)]^{-1}[Q(n-1) - A] \quad (4.5.2) \]

Essentially \( Q(n) \) is an approximation for \( A \) when it is used as a matrix multiplier for a class of vectors; the 'closer' \( Q(n) \) is to \( A \) the 'smaller' \( T(n) \). This is not strictly essential however, because \( T(n) \) may alternate 'big', 'small' and yet the combined effect could be to reduce the increment vector \( \tilde{\delta}(n) \). At first glance it seems attractive to take as the convergence criteria

\[ \rho(T(n)) < 1 \quad , \quad (4.5.3) \]

where \( \rho \) denotes the spectral radius of the matrix \( T \). Unfortunately the establishment of this condition proves elusive.

Even if (4.5.3) could be established, it is not sufficient (or even necessary) to guarantee convergence of the non-stationary form of MINI, all it would show, is that a stationary method based on MINI is convergent. (In other words assuming asymptotic \( \gamma \)'s exist the method would be convergent.) To verify this claim consider two subsequent iterations

\[ \tilde{\delta}(n) = T(n)\tilde{\delta}(n-1) \quad (4.5.4) \]

and
\[ \dot{\phi}(n+1) = \tau(n+1) \dot{\phi}(n) \]  

(4.5.5)

with \( \dot{\phi}(n-1) \) expanded in terms of the eigenvectors of \( \tau(n) \) (assumed to be complete)

\[ \dot{\phi}(n-1) = \sum_i a_i \dot{\psi}_i(n) , \]

then (4.5.4) is written

\[ \dot{\phi}(n) = \sum_i b_i \dot{\psi}_i(n) \]

for \( b_i = a_i (n) \lambda_i \) and where \( (n) \lambda_i \) denotes an eigenvalue of \( \tau(n) \). The second iteration (4.5.5) then is

\[ \dot{\phi}(n+1) = \sum_i b_i \tau(n+1) \dot{\psi}_i(n) \]  

(4.5.6)

If each eigenvector \( \dot{\psi}_i(n) \) is itself expressed in terms of the assumed complete set of eigenvectors \( \dot{\psi}_j(n+1) \) belonging to \( \tau(n+1) \)

\[ \dot{\psi}_i(n) = \sum_j c_j \dot{\psi}_j(n+1) , \]

(4.5.6) becomes
\[ \phi^{(n+1)} = \sum_j c_j^{(n+1)} \lambda_j^{(n)} \sum_i a_i^{(n)} \lambda_i^{(n)} u_j^{(n+1)}, \]

where \( (n) \lambda_j \) are the eigenvalues associated with each eigenvector \( u_j^{(n)} \) of \( T(n) \). Even if \( \max \{|(n) \lambda_i|\} < 1 \) there is no guarantee that any \( \| (n+1) \lambda_j \sum_i a_i^{(n)} \lambda_i \| < 1 \). An alternate choice of iteration norm such as \( \rho^{1/2}(T^*T) \) would be more appropriate.

Two convergence possibilities are considered now, for the first, an effectively stationary form of MINI is examined.

(i) An asymptotic situation is assumed to prevail in order to establish some plausible conditions for convergence. If

\[ \gamma^{(n)}_{ij} = \gamma^{(n+1)}_{ij} = \gamma_{ij}, \]

then \( Q(n) = Q \) and \( T(n) = T \) would be independent of \( n \). Let \( \lambda \) be a largest magnitude eigenvalue of \( T \) and \( \hat{x} \) an eigenvector such that

\[ T \hat{x} = \lambda \hat{x}. \quad (4.5.7) \]

Substituting (4.5.7) in (4.5.2) consequently gives

\[ [A - Q + \lambda Q] \hat{x} = 0. \quad (4.5.8) \]

Suppose a non-negative vector \( \tilde{s} \) exists such that the solution of
4.28

\[ A \bar{x} = s \]

consists of all positive 'minimum ratio' elements defined by

\[ \frac{\bar{x}_j}{\bar{x}_i} < 1 \quad \text{for all } j > i \text{ with } a_{ij} \neq 0 \]

and for any other \( s \), \( \frac{\bar{x}_j}{\bar{x}_i} < x_j/x_i \). Then if the acceptable choice

\[ \bar{\gamma}_{ij} = \frac{\bar{x}_j}{\bar{x}_i} \quad (< \min(1, x_j/x_i)) \quad (4.5.9) \]

is made, the relation

\[ A \bar{x} = Q \bar{x} \]

follows. The solution vector \( \bar{x} \) is consequently also an eigenvector \( \hat{x} \) of equation (4.5.8), corresponding to \( \lambda = 0 \). For matrices of small order, it has been discovered that all other eigenvalues are also zero, for the choice of \( \gamma_{ij} \) given by equation (4.5.9) - and it is conjectured this is so for any order matrix of the type under investigation. Asymptotically, as the iteration proceeds, the \( \gamma_{ij} \) tend to the values given by equation (4.5.9) (as in Section 4.7) and convergence is assured because the required (stationary) condition on \( \rho(T) < 1 \) is met.
Consider the simple example

\[
A = \begin{pmatrix}
1 & -a & 0 \\
-a & 1 & -b \\
0 & -b & 1
\end{pmatrix},
\]

with \( \bar{s} = (1,0,0)^T \), where T denotes transpose. This leads to

\[
\bar{y}_{12} = a/(1-b^2)
\]

and

\[
\bar{y}_{23} = b.
\]

(ii) An asymptotic situation may not exist, for which case numerical studies have demonstrated the \( y_{ij} \) must not exceed 1 for convergence. (Cases with \( y_{ij} = \) random number in (0,1) converge even if slowly.)

Although the convergence of MINI cannot be established for all \( y \) in the range (0,1), it is possible to demonstrate its convergence under the impracticable restrictive condition of stationary \( |y| < \frac{1}{2} \). Following Doherty [1977], consider the splitting of the symmetric matrix A.
$A = B - E - ET$, \\

where $B$ is the diagonal and $E ( = -L)$ is the lower triangular portion of $A$. The MINI process may be expressed as

$$Q \chi^{(n)} = z + (ET - \Gamma) \chi^{(n-1)},$$

i.e. $$(B - E - \Gamma) \chi^{(n)} = z + (ET - \Gamma) \chi^{(n-1)}, \quad (4.5.10)$$

where $\Gamma$ is a diagonal matrix containing the $\sum_{j>i}^{n} a_{ij} \chi_{ij}$ components of (4.2.2). The true solution $\chi$ satisfies

$$(B - E - \Gamma) \chi = z + (ET - \Gamma) \chi \quad , \quad (4.5.11)$$

Subtracting (4.5.11) from (4.5.10) gives

$$(B - E - \Gamma) e^{(n)} = (ET - \Gamma) \rho^{(n-1)} \quad , \quad (4.5.12)$$

where $e^{(n)} = \chi^{(n)} - \chi$. The expression (4.5.12) may be written as

$$(B - E - ET)e^{(n)} + (ET - \Gamma) e^{(n)} = (ET - \Gamma) \rho^{(n-1)}$$

and consequently
\[ A \tilde{e}(n) = (E^T - \Gamma)(\tilde{e}(n-1) - \tilde{e}(n)) \]
\[ = -(E^T - \Gamma)\tilde{\delta}(n), \quad (4.5.13) \]

where \( \tilde{\delta}(n) = \tilde{e}(n) - \tilde{e}(n-1) \). An alternate adjustment to (4.5.12) gives

\[ A \tilde{e}(n-1) = -(B - E - \Gamma)\tilde{\delta}(n). \quad (4.5.14) \]

Multiplication of (4.5.13) and (4.5.14) by \( \tilde{\delta}(n)^T \) and their combination gives

\[ \tilde{e}(n-1)^T A \tilde{e}(n-1) - \tilde{e}(n)^T A \tilde{e}(n) = \tilde{\delta}(n)^T (B - 2\Gamma)\tilde{\delta}. \quad (4.5.15) \]

It is easy to establish \( \chi < \frac{1}{2} \) is a sufficient condition to ensure convergence. Choose \( \tilde{e}(0) \) to be an eigenvector of \((B - E - \Gamma)^{-1}(E^T - \Gamma)\)

then

\[ \tilde{e}(1) = (B - E - \Gamma)^{-1} (E^T - \Gamma) \tilde{e}(0) \]
\[ = \lambda \tilde{e}(0) \]

and
4.32

\[ \delta^{(1)} = -(1-\lambda) \xi^{(0)} . \]

For this choice of \( \xi^{(0)} \) (4.5.15) reduces to

\[ (1-|\lambda|^2) \xi^{(0)T} A \xi^{(0)} = |1-\lambda|^2 \xi^{(0)T} (B-2\Gamma) \xi^{(0)} . \quad (4.5.16) \]

If \( \lambda \) were unity then \( \delta^{(1)} \) would be the null vector and by (4.5.14) \( A \xi^{(0)} = 0 \). Because \( A \) is positive definite and \( \xi^{(0)} \) is not a zero vector by definition, \( \lambda \) cannot be unity. For all \( |\gamma_{ij}| < \frac{1}{2} \) both sides of (4.5.16) are positive and hence \( |\lambda| < 1. \)

4.6 COMPUTATIONAL PROCEDURE

A FORTRAN segment formalising the MINI method is presented. In this

(i) \( A(I,J) \) denotes matrix elements \( a_{ij} \) of the supplied matrix of order \( N \) assumed to be present in fast memory,

(ii) \( X(I) \) denotes on entry trial solution elements \( x_i^{(0)} \) and on exit converged results \( x_i^{(n)} \),

(iii) \( \text{DELTA}(I) \) denotes elements of a temporary vector for storage of elements \( \gamma_i^{(n)} \) which are zero on entry,
(iv) MAX denotes a limit to the number of iterations, and

(v) ERR denotes a required error limit (say ERR = 10^{-4}) such that the solution is considered converged if

\[ \left| \frac{\delta_i^{(n)}}{x_i^{(n-1)}} \right| \leq ERR, \quad i=1,2,\ldots,N. \]

C MINI SOLUTION OF THE LINEAR EQUATIONS A X = s

DO 1 NIT=1,MAX
ERROR=-1.
DO 2 I=1,N
V=A(I,I)
U=S(I)-V*X(I)
IF(I.EQ.1)GO TO 4
I1=I-1
DO 3 J=1,I1
3 U=U-A(I,J)*X(J)
4 W1=0.
W2=0.
IF(I.EQ.N) GO TO 10
I1=I+1
DO 5 J=I1,N
AIJ=A(I,J)
IF(AIJ.EQ.O.) GO TO 5
XI=ABS(DELTA(I))
IF(XI.EQ.O.) GO TO 6
XJ=ABS(DELTA(J))
IF(XJ.GT.XI) GO TO 7
C NORMAL
GAMMA=XJ/XI
GO TO 8
C UPEND
7 GAMMA=XI/XJ
8 XI=X(I)
XJ=X(J)
IF(XJ.GT.XI) GO TO 9
C DOWNHILL
C 0.999999 ... FACTOR TO ASSURE NON-ZERO RHS
GIJD=XJ/XI
IF(GAMMA.GT.GIJD)GAMMA=GIJD*0.999999
9 W2=W2+AIJ*GAMMA
6 W1=W1+AIJ*X(J)
5 CONTINUE
V=V+W2
10 DELTA(I)=(U-W1)/V
4.34

C THIS TEST SAVES SOME ARITHMETIC WHEN FAR FROM CONVERGED
IF(ERROR.LT.ERR)ERROR=ABS(Delta(I)/X(I))
X(I)=X(I)+Delta(I)
2 CONTINUE
IF(ERROR.LE.ERR) GO TO 11
1 CONTINUE
C NOT CONVERGED IN MAXIMUM NUMBER OF ITERATIONS
C SET APPROPRIATE ERROR FLAG
C FINISH
11 CONTINUE

4.7 1-DIMENSIONAL MODEL PROBLEM

Because the matrix is tridiagonal, a direct solution is immediately possible for a one-dimensional reactor study. This is the normal method of solution for such a reactor idealisation, however, for purposes of comparing MINI with SOR, a one-dimensional reactor model problem is solved here with iterative methods. Throughout this work the phrase 'model problem' is used to signify a somewhat simplified problem that is not the representation of an actual reactor, but a small mathematical abstraction, possessing similar matrix properties, on which basic ideas may be tested first.

One such model problem gives the matrix

\[
A = \begin{pmatrix}
1 & -a & 0 & 0 \\
-a & 1 & -b & 0 \\
0 & -b & 1 & -a \\
0 & 0 & -a & 1
\end{pmatrix}
\]

for which
\[ y_{12} = \frac{a(1-a^2)}{(1-(a^2 + b^2))}, \quad (4.7.1) \]

\[ y_{23} = \frac{b}{(1-a^2)} \text{ and } y_{34} = a \]

corresponding to \( \vec{s} = (1,0,0,0)^T \). With

\[ \lambda_{GS} = a^2 + \frac{b^2}{2} + b \sqrt{a^2 + \frac{b^2}{4}} \]

\( \omega_{\text{opt}} \) is calculated using equation (4.1.9).

MINI is compared with GS and SOR in a series of trial calculations. The number of parameters is reduced by taking \( b=1-a \). Three right hand sides \( s_1=(1,0,0,0)^T, \ s_2=(1,1,1,1)^T \) and \( s_3=(0,0,0,1)^T \) are taken with the trial solution \( x_0=(1,1,1,1)^T \) and an error limit of \( 10^{-4} \). The results in Table 4.1, designated SORU and SORO, correspond to SOR calculations carried out with the underestimate,

\[ \omega = 2 - 1.1(2 - \omega_{\text{opt}}) \]

and the overestimate

\[ \omega = 2 - 0.9(2 - \omega_{\text{opt}}) \]

of the extrapolation parameter respectively. In practice \( \lambda_{GS} \) is estimated during the SOR method and this typically adds 10 to the
### TABLE 4.1
THE NUMBER OF ITERATIONS REQUIRED TO CONVERGE 1D MODEL PROBLEM

<table>
<thead>
<tr>
<th></th>
<th>GS</th>
<th>SOR</th>
<th>SORU</th>
<th>SORO</th>
<th>MINI</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a = 0.01)</td>
<td>267</td>
<td>53*</td>
<td>46*</td>
<td>59*</td>
<td>6</td>
</tr>
<tr>
<td>(\lambda_{GS} = 0.9803)</td>
<td>267</td>
<td>35</td>
<td>44</td>
<td>24</td>
<td>6</td>
</tr>
<tr>
<td>(\omega_{opt} = 1.754)</td>
<td>267</td>
<td>53*</td>
<td>46*</td>
<td>60*</td>
<td>8</td>
</tr>
<tr>
<td>(a = 0.1)</td>
<td>42</td>
<td>15</td>
<td>19</td>
<td>18</td>
<td>7</td>
</tr>
<tr>
<td>(\lambda_{GS} = 0.8299)</td>
<td>41</td>
<td>14</td>
<td>18</td>
<td>13</td>
<td>7</td>
</tr>
<tr>
<td>(\omega_{opt} = 1.416)</td>
<td>41</td>
<td>15*</td>
<td>18*</td>
<td>18*</td>
<td>8</td>
</tr>
<tr>
<td>(a = 0.5)</td>
<td>19</td>
<td>10</td>
<td>13</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>(\lambda_{GS} = 0.6595)</td>
<td>21</td>
<td>10</td>
<td>14</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>(\omega_{opt} = 1.260)</td>
<td>20</td>
<td>11</td>
<td>14</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>(a = 0.9)</td>
<td>71</td>
<td>17</td>
<td>22</td>
<td>16</td>
<td>7</td>
</tr>
<tr>
<td>(\lambda_{GS} = 0.9051)</td>
<td>71</td>
<td>19</td>
<td>24</td>
<td>18</td>
<td>8</td>
</tr>
<tr>
<td>(\omega_{opt} = 1.529)</td>
<td>74</td>
<td>20</td>
<td>25</td>
<td>19</td>
<td>8</td>
</tr>
<tr>
<td>(a = 0.99)</td>
<td>500</td>
<td>48</td>
<td>59</td>
<td>47</td>
<td>8</td>
</tr>
<tr>
<td>(\lambda_{GS} = 0.9901)</td>
<td>463</td>
<td>48</td>
<td>58</td>
<td>34</td>
<td>8</td>
</tr>
<tr>
<td>(\omega_{opt} = 1.819)</td>
<td>504</td>
<td>50</td>
<td>62</td>
<td>41</td>
<td>11</td>
</tr>
</tbody>
</table>
number of SOR iterations. Errors in the estimate of $\lambda_{GS}$ may also lead to an under- or overestimate of $\omega$ compared with the optimal value.

The calculations are carried out in single precision on an IBM 3031 using an interactive FORTRAN system (Clancy [1977]). The system provides for interactive movie display (Cawley and Trimble [1977]) of $\chi(n)$ as $n$ changes and the behaviour of any process can be watched for 'roughness' etc.

The results in Table 4.1 suggest that MINI, at least for this simple case, is acceptable when compared with GS and SOR methods, even allowing for the additional computation it requires. MINI requires fewer iterations because the asymptotically calculated $\gamma_{ij}$ in each case are essentially the same as the values calculated from equation (4.7.1). For more substantial problems the savings of MINI compared with SOR are not always so great. (The results marked * for relaxation methods indicate negative intermediate solutions.)

4.8 LINE MINI APPROACH

When the matrix $A$ arises from a two-dimensional representation of a reactor, a line approach is more appropriate than the previous point methods. The successive line over-relaxation method (SLOR), Young [1971], is successfully used in many reactor codes. Before the three-dimensional code was designed, one of the early codes, POW [Pollard 1974] was used as a test vehicle for assessing the worth of
MINI, and results of this experiment were reported [Barry and Pollard 1977]. Some are repeated in Chapter 8. The 5-point finite difference operators are set up as shown in Figure 4.3. For each grid point the finite difference approximation used requires only one $y_{ij}^{(n)}$ (as defined by 4.2.1 and 4.3.2). The computational success obtained with two-dimensional MINI computations confirmed the resolve to consider a three-dimensional version in a nuclear code.

4.9 MINI FOR THREE-DIMENSIONAL PROBLEMS AND ENERGY GROUPS

The two-dimensional form of MINI may be extended to handle problems with three spatial dimensions. The matrix properties of a three-dimensional form are the same as those for two-dimensions (4.1.3). The solution proceeds by a double layer iterative scheme, the inner most layer being the solution of a $(x,y)$ plane, within the outer layer, which solves for all planes. The matrix properties (4.1.3) again guarantee a positive solution for $\lesssim$ non-negative, corresponding to physical reality.

MINI also may be extended to handle the energy group upscatter iterations which were described in Section 3.3. For this situation, the G group neutron diffusion equation (3.3.5) corresponding to one outer iteration again can be expressed in the general block matrix form.
Schematics of the line MINI approach

Figure 4.3
\[ A \phi = \xi, \]  

where the right hand side (or eigenvalue terms) are written as source terms for convenience. The detailed structure in matrix terms relevant to the MINI discussion is

\[
\begin{bmatrix}
A_{11} & 0 & & & & & & & \\
A_{21} & A_{22} & & & & & & & \\
 & & \ddots & & & & & & \\
A_{G'1} & \cdots & A_{G'G'} & A_{G'G'+1} & \cdots & A_{G'G''} & & & \\
 & & & \ddots & & & \ddots & & \\
A_{G1} & A_{G2} & & & & & A_{GG} & & \\
\end{bmatrix}
\begin{bmatrix}
\phi_1 \\
\phi_2 \\
\vdots \\
\phi_{G'} \\
\vdots \\
\phi_G \\
\end{bmatrix} = 
\begin{bmatrix}
\xi_1 \\
\xi_2 \\
\ddots \\
\xi_{G'} \\
\ddots \\
\xi_G \\
\end{bmatrix}.
\]

The \( A_{ij} \) are block submatrices, representing the finite difference discretisation in three-dimensional space which approximates the diffusion, removal and scattering components of (3.1.8). The \( \phi_i \) and \( \xi_i \) give the spatial distribution of fission production for each energy group. The first group into which upscattering occurs is denoted by \( G' \). The matrix and vector properties of \( A \) and \( \xi \) in (4.9.1) are

\[
a_{ii} > 0, \\
a_{ij} = 0, \quad i \neq j \\
a_{ii} \geq \sum_{j=1}^{i} a_{ji} \quad \text{(with inequality for at least one value of } i),
\]

with \( s_i \geq 0 \) (not all zero).  

(4.9.3)
The lack of symmetry for the group problem (4.9.1) rules out many otherwise attractive iterative schemes. Equation (4.9.1) being lower triangular is solved directly up to group $G' - 1$

$$A_{gg} \phi(n) = S_g - \sum_{j < g} A_{gj} \phi(n), \quad g \leq G' - 1$$

and GS iteration for the remaining groups

$$A_{gg} \phi(n) = S_g - \sum_{j < g} A_{gj} \phi(n) - \sum_{j > g} A_{gj} \phi(n-1), \quad g > G'.$$

As in the previous sections, for ease of presentation, the above neutron diffusion equation will be written in the simpler notation

$$\sum_{j=1}^{N} a_{ij} x_j = s_i, \quad i=1,2,\ldots,N, \quad (4.9.5)$$

and a generalised approach for the multi-layer system is outlined.

A block solution process for non-overlapping ordered partitions $m=1,2,\ldots,M$, is used. (For example, $m=1$ might denote an $x$-line with $(y,z)$ fixed, $m=2$ its immediate neighbouring line. Alternatively it may denote an $(x,y)$ plane, or a particular energy group.)
The following sets of indices are now introduced for a partial stage of an iteration pass \( n \) through (4.1.2) or (4.9.1):

- \( J_m \) = the set of indices \( j \) for elements \( x_j \) to be updated together as a block,

- \( J^-_m \) = the set of indices \( j \) for elements \( x_j \), that have been updated already,

\[
J^-_m = J_1 \cup J_2 \cup \cdots \cup J_{m-1}
\]

and

- \( J^+_m \) = the set of indices \( j \) for elements \( x_j \), that have not yet been updated,

\[
J^+_m = J_{m+1} \cup J_{m+2} \cup \cdots \cup J_M
\]

The basic block Gauss-Seidel iteration process may be written as

\[
J^-_m \ a_{ij} x_j^{(n)} + J_m \ a_{ij} x_j^{(n)} + J^+_m \ a_{ij} x_j^{(n-1)} = s_i , \quad \text{for } i \in J_m, \ m=1,2,\ldots,M,
\]

where the J's as used in (4.9.6) also denote partial summation over the index \( j \), (for example \( J^-_m = \sum_{j \in J^-_m} \)).
The MINI approach for hastening convergence of the basic process given by equation (4.9.6), provides a better estimate for the last term on the left hand side. As for the two-dimensional form, $x_j^{(n)}$ is replaced with the diagonally coupled term,

$$x_j^{(n-1)} + \gamma_{ij}^{(n)}(x_i^{(n)} - x_i^{(n-1)})$$

where $\gamma_{ij}^{(n)}$ are the extrapolating parameters detailed in Section 4.3. The MINI process is then given by

$$J_m a_{ij} x_j^{(n)} + (J_m a_{ij} \gamma_{ij}^{(n)}) x_i^{(n)}$$

$$= s_i - J_m a_{ij} x_j^{(n)} - J_m a_{ij} (x_j^{(n-1)} - \gamma_{ij}^{(n)} x_i^{(n-1)})$$

$$i \in J_m, \quad m=1,2,\ldots,M.$$ (4.9.7)

The block of equations for the set $J_m$ may be solved by a direct method, by a further iterative process, or even by recursive use of the MINI process.

The problems of interest all have positive solutions $x_i$. The MINI iterative process requires that even intermediate solutions $x_i^{(n+1)}$ are positive. As a step towards ensuring positive solutions, the right hand side of (4.9.7) should be positive. This is achieved through the sufficient condition
\[ 0 \leq \gamma_{ij}^{(n)} < \frac{x_{j}^{(n-1)}}{x_{i}^{(n-1)}} \quad (4.9.8) \]

To assure positive solutions for \( x_{j}^{(n)} \), \( j \in J_m \), on the left hand side of (4.9.7), the block matrix of coefficients with elements

\[ a_{ik} + (a_{m}^{*} a_{ij} \gamma_{ij}) \delta_{ik}, \quad i \in J_m, \ k \in J_m, \]

where

\[ \delta_{ik} = \begin{cases} 1 & \text{if } i=k \\ 0 & \text{otherwise} \end{cases} \]

must not contain excessively large \( \gamma \)'s. Although somewhat restrictive, let \( g_m \) be an upper \( \gamma \)-limit for the \( m \)th block provided that

\[ \gamma_{ij}^{(n)} < g_m, \]

the block matrix of coefficients will yield positive solutions. When the results are collected, the overall restriction is

\[ 0 \leq \gamma_{ij}^{(n)} < g_{ij}^{(n)} \quad (4.9.9) \]

where
A preliminary calculation of the block $\gamma$-limits ($g_m$) could be carried out, or $g_m$ could be lowered empirically if negative solutions occurred, however, here we use

$$g_m = 1 \quad , \quad (4.9.10)$$

in keeping with the earlier procedure (Section 4.2). This limit will be used throughout the calculations, even though occasional negative intermediate solutions may be found for non-symmetric matrices. Of course it is necessary for the process to converge, the restrictions to maintain positive intermediate solutions are only part of the overall restrictions. Condition (4.9.10) is then found to be desirable.

Leaving out the details given in Section 4.3, the working equations for the extrapolating parameters are

$$\gamma_{ij}(n) = \begin{cases} \overline{\gamma}_{ij}(n) & \text{if } \overline{\gamma}_{ij}(n) \leq g_{ij}(n) \\ g_{ij}(n) & \text{otherwise} \end{cases}$$

where
Here $\gamma_{ij}^{(o)}$ is either obtained from the last overall pass through the current layer, or it is taken to be zero (a GS iteration). The extrapolating parameters are not stored, but rather $\delta_{i}^{(n)}$ is saved after each iterative pass for use in the next round with (4.9.11).

4.10 CONVERGENCE

A similar treatment to that provided in Section 4.5 applies for the more general form. The solution process may be expressed in matrix form as

$$Q(n)\bar{\delta}(n) = \bar{\zeta} - A_{\bar{\zeta}}(n-1) ,$$

where the elements of the matrix $Q(n)$ are given as

$$q_{ij} = \begin{cases} a_{ij} & i \in J_{m}, j \in J_{m}^{-} \cup J_{m} \text{ but } i \neq j; \\ a_{ii} + J_{m}^{+} a_{ij} \gamma_{ij}^{(n)} & \text{for the diagonal term;} \\ 0 & \text{otherwise,} \\ m=1,2,\ldots,M. \end{cases}$$

For the asymptotic form associated with a 4x4 system in blocks of two, the eigenvalue relationship (4.5.7) and (4.5.8) exists with the equivalent coefficient matrix
\[
\begin{pmatrix}
-a_{13}y_{13} - a_{14}y_{14} + \lambda(a_{11} + a_{13}y_{13} + a_{14}y_{14}) \\
\lambda a_{21} \\
\lambda a_{31} \\
\lambda a_{41}
\end{pmatrix}
\begin{pmatrix}
\lambda a_{12} \\
\lambda a_{32} \\
\lambda a_{42}
\end{pmatrix}
\begin{pmatrix}
a_{13} \\
a_{23} \\
a_{43}
\end{pmatrix}
\begin{pmatrix}
a_{14} \\
a_{24} \\
a_{44}
\end{pmatrix}
\]
for $\hat{\chi}$ an eigenvector of $T$ given by equation (4.5.2). Again if the $\gamma$'s are chosen satisfying

$$\gamma_{ij} = \hat{\chi}_j / \hat{\chi}_i \quad ,$$

(4.10.1)

(remembering, however, the restriction (4.9.9)), then direct substitution of (4.10.1) in the equivalent version of (4.5.8) gives

$$\lambda A \hat{\chi} = 0 \quad ,$$

hence all the eigenvalues must be zero since $A$ is invertible. The choice of $\gamma$'s given by equation (4.10.1) although plausible, seems somewhat remote until it is recognised that as iteration proceeds it is possible that

$$\delta^{(n)}_i \approx \hat{\chi}_i \quad , \quad i=1,2,\ldots,N \quad .$$

In this instance, the $\gamma$'s of equation (4.10.1) are calculated from the working equations (4.9.1). Further analysis, however, is possible.

Since asymptotically the $g$-restriction of (4.9.9), or rather the more exact form (4.9.8) is met automatically, the restriction reduces to
\[ 0 \leq \gamma_{ij} < \frac{x_j}{x_i} \]

That is

\[ 0 \leq \frac{x_j}{x_i} < \frac{x_j}{x_i} \]

Suppose there exists a non-negative vector \( \mathbf{s} \) such that the solution of

\[ A \mathbf{x} = \mathbf{s} \]

consists of all positive 'maximum ratio' elements. That is, for any other choice of \( \mathbf{s} \),

\[ \frac{x_j}{x_i} < \frac{x_j}{x_i} \text{ for } i \in J_m, j \in J^+, a_{ij} \neq 0 \]

We could then make the acceptable choice

\[ \bar{\gamma}_{ij} = \frac{x_j}{x_i} \]

which gives

\[ A \bar{\mathbf{x}} = Q \bar{\mathbf{x}} \]

The solution vector, \( \bar{\mathbf{x}} \), is then also an eigenvector \( \hat{x} \) of
corresponding to $\lambda = 0$. Of course such a vector $\vec{z}$ might not exist.

A simple example of the non-symmetric type of matrix in mind is

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix},$$

which will be used in blocks of two to solve $A \vec{x} = \vec{z}$ with GS and MINI iterations. We find that $\vec{\xi} = (0,1,0,0)^T$, and $\vec{\gamma}_{23} = 1$, which is the only $\gamma$ required. It is interesting to calculate the eigenvalues of the iteration matrix (cf. equation (4.10.2)) when

$$\gamma_{23} = \bar{\gamma}_{23} - \varepsilon.$$

They are $\{0,0,0,2\varepsilon/(1+2\varepsilon)\}$, hence the GS method ($\gamma_{23} = 0, \varepsilon = \bar{\gamma}_{23}$) has the eigenvalues $\{0,0,0,2/3\}$. It is noted that any $0 < \gamma \leq 1$ will give faster convergence than the GS method.

Table 4.2 shows the results from a numerical experiment with the same matrix $A$ for different iterative methods, and with different constant vectors, $\vec{z}$.
TABLE 4.2
NUMBER OF ITERATIONS TO CONVERGE SIMPLE 4x4 PROBLEM

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block GS</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>2*</td>
</tr>
<tr>
<td>Block MINI</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Point GS</td>
<td>35</td>
<td>36</td>
<td>36</td>
<td>33</td>
</tr>
<tr>
<td>Point MINI*</td>
<td>17</td>
<td>17</td>
<td>19</td>
<td>17</td>
</tr>
</tbody>
</table>

*Elements x_3 and x_4 of trial solution are correct.

+Almost identical successive over-relaxation (SOR).

TABLE 4.3
VARIATION OF LARGEST MAGNITUDE ITERATION EIGENVALUE

<table>
<thead>
<tr>
<th>n</th>
<th>(\lambda_p)</th>
<th>n</th>
<th>(\lambda_p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GS start</td>
<td>9</td>
<td>-0.5874</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.8257</td>
<td>10</td>
<td>-0.5984</td>
</tr>
<tr>
<td>2</td>
<td>0.7934</td>
<td>11</td>
<td>-0.6716</td>
</tr>
<tr>
<td>3</td>
<td>0.8257</td>
<td>12</td>
<td>-0.6097</td>
</tr>
<tr>
<td>4</td>
<td>0.6667</td>
<td>13</td>
<td>-0.5876</td>
</tr>
<tr>
<td>5</td>
<td>0.6489</td>
<td>14</td>
<td>-0.6426</td>
</tr>
<tr>
<td>6</td>
<td>-0.5805</td>
<td>15</td>
<td>-0.6398</td>
</tr>
<tr>
<td>7</td>
<td>-0.6256</td>
<td>16</td>
<td>-0.5879</td>
</tr>
<tr>
<td>8</td>
<td>-0.6750</td>
<td>17</td>
<td>-0.6149</td>
</tr>
</tbody>
</table>
\[ s_1 = (1,0,0,0)^T, s_2 = (0,1,0,0)^T, \]
\[ s_3 = (0,0,1,0)^T, s_4 = (0,0,0,1)^T. \]

A trial solution of \( x^{(0)} = (1,1,1,1)^T \) was used in all cases and the terminating condition was taken as
\[ |\delta_i^{(n)}/x_i^{(n-1)}| \leq 10^{-4}, \quad i=1,2,...,N, \]
which is also the one generally applied in the present work. A value of \( g^1 = 3/2 \) holds as a block \( \gamma \)-limit, although, as usual \( g^1 = 1 \) was used in the iterative process.

The results in Table 4.2 show that MINI satisfactorily solves the simple problem. The block MINI converges rapidly with the early achievement of an asymptotic \( \gamma \). On the other hand, no \( \tilde{s} \) exists for the point MINI as different element ratios, \( \bar{x}_{i+1}/\bar{x}_i \) require different \( \tilde{s} \)'s to achieve their minimum value. Consequently, no totally asymptotic situation is ever achieved except, in part, when \( \gamma_{34} = 1/2 \) is obtained. Analysis of the point MINI process for the simple problem shows that the minimum largest magnitude eigenvalue of the iteration matrix \( \lambda_L = \pm 1/\sqrt{3}(= \pm 0.5774) \) is obtained when \( \gamma_{12} = 1/2, \gamma_{23} = 1 \) and \( \gamma_{34} = 1/2 \). Table 4.3 shows the variation of the largest magnitude eigenvalue, \( \lambda_p \), of the point MINI process (assumed to be instantaneously stationary) based on the \( \gamma \)'s obtained at each stage of iteration \( n=1,2,...,17 \) during solution of \( A \bar{x} = \bar{s} \).
From Table 4.3 it is seen that the MINI process is persistently convergent and that $\lambda_p$ approaches the theoretical limit $\lambda_L$ to within a few per cent for some iterations.

In general, when there is no asymptotic situation, or when it is not achieved during the early stages, little can be proved about convergence. Instead numerical experiments by Doherty [1981] where several error norms were computed revealed MINI was not monotonically convergent - a usual assumption of analysis. The remainder of this Chapter and portions of Chapter 5 are devoted to the implementation of MINI and its relationship with a many parameter SOR iteration scheme. In Chapter 8, MINI is evaluated on a number of numerical experiments, mostly for real reactor calculations, which support the conjecture that MINI converges, provided the $\gamma$'s do not exceed unity.

4.11 IMPLEMENTATION OF MINI

The full details of how MINI is implemented in the code POW3D are given in Chapter 5. For now, a new rule is described to overcome a possible source of trouble when

$$\gamma_{ij}^{(n)} = \frac{x_j^{(n)}}{x_i^{(n)}} ,$$

as in this instance slightly less is required by inequality (4.9.9),
otherwise the entire right hand side of (4.9.7) might become zero. The FORTRAN coding of Section (4.6) anticipated this requirement.

The 'inner' solution process used to solve a 7-point edge mesh finite difference approximation to the basic group equation

\[-\nabla \cdot D_g \nabla \phi_g + \Sigma_{rg} \phi_g = s_g \tag{4.11.1}\]

is considered now. In (4.11.1) \(s_g\) denotes scattering and fission contributions based on previous estimates of the flux. The solution process iterates through the 'layers':

(i) z-direction 'between (x,y) plane' inners for the m-th pass,

(ii) '(x,y) plane' inners for the n-th pass (and m-th between plane pass), and

(iii) x line direct solution (n,m).

For ease of notation in the present description, it is assumed that \(D\) is constant as is the mesh width \(h_y\). Temporarily dropping group dependence, the numerical approximation is then equivalent to

\[- \frac{\partial}{\partial y} D \frac{\partial \phi}{\partial y} \approx \Sigma_y (-\phi_{j-1} + 2\phi_j - \phi_{j+1}) \]

where
\[ y = \frac{D}{h_y^2}, \]

and \( \phi_{j-1}, \phi_j, \phi_{j+1} \) are successive values for the flux along a line in the y-direction. As the iteration progresses in the y-direction, the flux \( \phi_{j+1} \) will not be available for the present pass \( n \), hence the normal approach is to approximate the y-direction leakage with

\[
- \left[ \frac{\partial}{\partial y} D \frac{\partial \phi}{\partial y} \right] (n, n-1) \approx \sigma_y \left( -\phi_{j-1}(n,m) + 2\phi_j(n,m) - \phi_{j+1}(n,m) \right),
\]

in which emphasis is given, on the left, to the appearance of the flux for the previous iteration, \( n \). The usual MINI approximation

\[
\phi_{j+1}(n,m) \approx \phi_j(n-1,m) + \gamma(n-1,m) (\phi_j(n,m) - \phi_j(n-1,m)),
\]

dropping the subscript \( j \) denoting the particular line from \( y \) and \( \phi \), produces

\[
- \left[ \frac{\partial}{\partial y} D \frac{\partial \phi}{\partial y} \right] (n, n) \approx - \left[ \frac{\partial}{\partial y} D \frac{\partial \phi}{\partial y} \right] (n, n-1)
\]

\[- \sigma_y \gamma(n-1,m) (\phi(n,m) - \phi(n-1,m)), \]

Similarly

\[
- \left[ \frac{\partial}{\partial z} D \frac{\partial \phi}{\partial z} \right] (n, m) \approx - \left[ \frac{\partial}{\partial z} D \frac{\partial \phi}{\partial z} \right] (n, m-1) \]
where \((\cdot, m-1)\) denotes results obtained from converged fluxes of the previous z-pass. (The diagonal leakage term always remains implicit as \(\phi(n,m)\).) Collecting the results, equation (4.11.1) becomes

\[
-\sigma_z \chi(\cdot, m-1) \left( \phi(n,m) - \phi(\cdot, m-1) \right),
\]

\[
\frac{\partial}{\partial x} D_g \frac{\partial \phi}{\partial x}(n,m) - \frac{\partial}{\partial y} D_g \frac{\partial \phi}{\partial y}(m)
\]

\[
\frac{\partial}{\partial z} D_g \frac{\partial \phi}{\partial z}(n,m,m-1)
\]

\[
+ (\sigma_{rg} - \sigma_{yg} \chi y_g - \sigma_{zg} \chi z_g) \phi_y(n,m)
\]

\[
= s_g - (\sigma_{yg} \chi y_g(n-1,m) \phi_y(n-1,m) + \sigma_{zg} \chi z_g(n-1,m) \phi_z(n-1,m)),
\]

which is also the result for a general mesh with appropriate changes to \(\sigma_{yg}\) and \(\sigma_{zg}\). Equation (4.11.2) is similar to the approximation normally obtained, except for the reduced removal terms and the compensating terms on the right.

The MINI approach (equation 4.11.2), like the successive line over-relaxation process [Young 1971], is an extrapolation of the basic GS method (where \(\chi = 0\)). Either MINI or SLOR may be used for any one direction, thus enabling a combination approach to be used for the whole reactor calculation. When SLOR is used for the (x,y) plane and MINI is used for the between planes (z), the optimum relaxation parameters may need continual redetermination because of the changing
removal term.

5. CONCLUSIONS

Equations (4.2.5, 4.2.6, and 4.2.7) suggest MINI is related to the general class of relaxation methods given by

\[(B + [\omega]L)x^{(n)} = ((I - [\omega]B - [\omega]U)x^{(n-1)} + [\omega]s)\] (4.12.1)

for \(A = L + B + U\), where instead of \([\omega]\) being a single relaxation parameter, it is now a square diagonal matrix. It is expected, that if all the relaxation parameters \(\omega_{ij}\) could be determined adequately, the iterative scheme (4.12.1) would be superior to SOR. MINI provides a means of computing these parameters.

The elements of \([\omega]\) are related to the \(\gamma\)'s by the relation (4.2.7)

\[\omega_{ii} = \frac{a_{ii}}{a_{ii} + \sum_{j > i} a_{ij} \gamma_{ij}}\]

or in matrix terms by

\[[\omega] = B[B + \Gamma]^{-1}\] (4.12.2)

where as before \(\Gamma\) is the diagonal matrix whose components are
\( \sum_{j>i} a_{ij} \delta_{ij} \). Substitution for \([\omega]\) in (4.12.1) and multiplication by \([B + \Gamma]^{-1}\) leads to

\[
(L + B + \Gamma)x(n) = (\Gamma - U)x(n-1) + \xi
\]

and then

\[
x(n) = -(L + B + \Gamma)^{-1}(U - \Gamma)x(n-1) + (L + B + \Gamma)^{-1}\xi,
\]

which is immediately recognised as the matrix form of MINI equation (4.4.2).

In fact, the ordinary SOR is essentially a special case of MINI, where all elements of \([\omega]\) are equal, stationary and bounded \(1 < \omega < 2\). The \(\chi's\) for MINI must satisfy

\[
\Gamma = -(I - [\omega])^{-1}B \quad (4.12.3)
\]

(The negative values arise from the negative off-diagonal elements of \(A\) and the \(\chi's\) turn out to be positive provided the \(\omega's\) exceed unity.)

This system is usually an underdetermined one, unless each grid point is connected by only one \(\gamma_{ij}\) to a forward point. In general there are infinitely many values of \(\gamma_{ij}\) leading to the SOR method.
The relationship (4.12.1) helps justify the restrictions placed on \( \gamma \) (viz. \( 0 \leq \gamma \leq 1 \)). The lower bound ensures \( \omega \geq 1 \), while the upper bound ensures \( \omega \geq 0 \). The relationship between \( \omega \) and \( \gamma \) is shown for the variation of a particular \( \gamma \) in Figure 4.4, where the point discontinuity \( \gamma_c \) satisfies \( \gamma_c \geq 1 \).

Despite the lack of a proof of MINI's convergence, the fact that for a somewhat restricted set of \( \gamma \)'s within the permitted range, the method corresponds to the established SOR technique augurs well for the method.

Although the setting for the idea of MINI arose originally from a suggestion in a talk by Noble [1976], where he referred to a concept due to Curtis of making a solution process more implicit, the multi-parameter system of \( \omega \)'s given by (4.12.1) closely relates the technique to the relaxation methods. The principal problems hindering the automation of the original human-directed relaxation techniques are by how much should the residual at each point be allowed to vary from zero and what restrictions should be placed on its fluctuating sign. Human practical experience was essential in the execution of the art [Shaw 1953]. Even though MINI evolved differently it comes some way to resolving this difficulty on a machine.
ω as a function of a particular γ

Figure 4.4
0, thou hast damnable iteration, and art, indeed, able to corrupt a saint. Thou hast done much harm upon me,...

Henry IV,pt.1

5.1 MOTIVATION

All too often, once a numerical technique has been formulated in mathematical terms, there are those who are inclined to dismiss the computational implementation as a trivial exercise. This is rather unfortunate, because what appears efficient in a mathematical sense may not be quite so promising when limitations of computer capacity and function are considered.

There is no unique 'best' implementation. What is 'best' is appropriate to the type of computing facility available, and the use to which the code will be subjected. The development of POW3D coincided with a major change in computer architecture at the AAEC's Research Establishment. POW3D was designed to function successfully in the old and new environments. As will become apparent here, the form in which a technique is best implemented for a particular problem on a computer, may make it appear quite different at first glance to the original mathematical definition. The practical implementations are relevant and as shown in the evolution of MINI (Chapter 4),
computational experiences may well influence the development of the original technique.

It is the author's observation from developing the code POW3D, that mathematical effort spent in implementing the most efficient iterative algorithms on this problem is wasted by comparison, should great care not be exercised also in writing the computer code, particularly that portion which evaluates the scattering and fission terms for the right hand side of the system (3.3.5) (i.e. the source terms).

To ensure the nuclear code functioned adequately, great care was taken to develop a monitoring system that ran under POW3D. This reported as to how the various components of the code made use of the valuable computational resources. It recorded the number of times each module was invoked, the central processing unit (CPU) time spent in execution of the module, and the elapsed ('wall clock') time. The latter reflecting both the I/O overhead and the sharing of resources under a multi-user environment.

Using this invaluable information the code was tuned; inefficient portions of the code being identified quickly and re-designed to perform in a more satisfactory manner.
POW3D is intended as a workhorse for reactor physics studies at the AAEC's Research Establishment. As such it constitutes a fundamental block of the AUS code system (Robinson [1975]). Because of its extreme importance and anticipated frequent use, no effort was spared in reworking sections of the code in which an excessive proportion of the computer time was spent.

The author believes that the actual implementation of the numerical techniques belongs within the discipline of mathematics and that responsibility for it should not be abrogated to computer science. Accordingly, the way in which MINI and ICCG function within POW3D is discussed in some detail. (The implementation of three-dimensional SLOR has many features in common with MINI, particularly so far as data structures are concerned for the third dimension. Consequently a discussion for that method is not pursued). ICCG is described first.

5.2 MODIFIED MATHEMATICAL DESCRIPTION OF ICCG

Attention is restricted to a system of linear equations of the form

\[ A \phi = s \]

corresponding to a one group subsystem of (3.3.5), which for a three-dimensional reactor can be written in the more explicit block form as
The three-dimensional domain is discretised by a regular grid system with \( N_x \), \( N_y \) and \( N_z \) points in the \( x \), \( y \), and \( z \) coordinate directions respectively. Components of the unknown flux \( \phi \) are arranged in a vector fashion. The organisation of the data corresponds to the following ordering of the spatial points

\[
\phi_{ijk} (((i=1,2,\ldots,N_x),j=1,2,\ldots,N_y),k=1,2,\ldots,N_z) .
\]

In general there is too much data for retention in fast memory at any one time, so it is structured in a block format. The basic blocks are submatrices \( A_{ij} \) which are square and of order \( N_x N_y \) for the coefficient matrix \( A \) of (5.2.1). Subvectors of the same order are used for \( \phi_i \) and \( s_i \). The submatrices \( A_{ij} (i\neq j) \) are diagonal, while the diagonal matrices \( A_{ii} \) are themselves block tridiagonal; the structure of each diagonal matrix being of a similar block form to that of the original matrix \( A \) in (5.2.1). This time there are \( N_y \) square submatrices of order \( N_x \). The off-diagonal matrices are again diagonal, while the diagonal block matrices have a tridiagonal structure. The overall structure is a direct consequence of the seven point finite difference
A mathematical definition of MINI was discussed at some length in Chapter 4. That for the ICCG is summarised here.

The conventional conjugate gradient method for solving \( Ax = b \) is expressed

Select \( x_0 \) as an initial approximation to \( x \)

\[
\begin{align*}
\mathbf{r}_0 &= \mathbf{b} - A\mathbf{x}_0 \\
\mathbf{t}_0 &= \mathbf{r}_0 \\
\alpha_i &= \frac{\langle \mathbf{r}_i, \mathbf{r}_i \rangle}{\langle \mathbf{t}_i, A\mathbf{t}_i \rangle} \\
\mathbf{x}_{i+1} &= \mathbf{x}_i + \alpha_i \mathbf{t}_i \\
\mathbf{r}_{i+1} &= \mathbf{b} - A\mathbf{x}_{i+1} \text{ or } \mathbf{r}_i - \alpha_i A\mathbf{t}_i \\
\beta_i &= \frac{\langle \mathbf{r}_{i+1}, \mathbf{r}_{i+1} \rangle}{\langle \mathbf{r}_i, \mathbf{r}_i \rangle} \\
\mathbf{t}_{i+1} &= \mathbf{r}_{i+1} + \beta_i \mathbf{t}_i.
\end{align*}
\]

The process converges in at most \( N = N_x N_y N_z \) steps, assuming no ill effects from rounding error, but adequate solutions occur usually for
Slight variation in specifications of the algorithm occur, involving different operation counts and storage overheads. For exact computer arithmetic the techniques have identical results, however, round-off error may well upset this. Reid [1971] discusses the original Hestenes and Stiefel approach and compares this with that due to Rutishauser. The original version is favoured when the ordinary definition of scalar product is used for the inner product of two vectors. This work departs from the recommended use of a recursive relation to estimate the residual, because slightly better accuracy is anticipated for large problems in implementations where single precision arithmetic is pushed to the limit.

The incomplete Choleski refinement involves an approximate factoring of

\[ A = L D L^T, \]

and the application of a subsequent approximation to \( A^{-1} \) in an attempt to correct the error in the solution at each stage of the conjugate gradient process. The new algorithm now appears as

Select \( x_0 \) as an initial approximation to \( x \)

\[ r_0 = b - Ax_0. \]
\[
t_0 = (LDL^T)^{-1}r_0
\]

\[
\alpha_i = \langle r_i, (LDL^T)^{-1}r_i \rangle / \langle t_i, At_i \rangle
\]

\[
x_{i+1} = x_i + \alpha_i t_i
\]

\[
r_{i+1} = b - Ax_{i+1} \text{ or } r_i - \alpha_i At_i
\]

\[
b_i = \langle r_{i+1}, (LDL^T)^{-1}r_{i+1} \rangle / \langle r_i, (LDL^T)^{-1}r_i \rangle
\]

\[
t_{i+1} = (LDL^T)^{-1}r_{i+1} + b_it_i
\]

Such formulations are concise within a mathematical framework. When the practical details for implementing the technique on even the most advanced computer systems are considered, however, they are not as explicit as they might be. This becomes apparent, particularly when fine mesh studies are undertaken for three-dimensional reactor configurations, with a view to comparing properties and performance of the various iterative methods on a number of criteria.

In Section 5.3 a mathematical description will be given for the ICCG technique in a form suited to the three-dimensional reactor physics problem, and appropriate for the implementation in POW3D. The formulation is based on the linear system (5.2.1). Following this basic mathematical definition, a computational description of the ICCG code in POW3D is provided (Section 5.5).
ICCG requires an approximate LDL$^\top$ decomposition of $A$. In POW3D the traditional Choleski decomposition approach is followed, whereby $L$ and $D$ are computed from

\[ l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} d_{kk} l_{jk} \]

\[ d_{ii} = 1/l_{ii}, \]

with the exception that no zero infill in $L$ is permitted; i.e. if $a_{ij}=0$ then $l_{ij}=0$.

This lack of infill dramatically reduces the storage otherwise required. For ICCG little additional storage is required and at the same time, the arithmetic overhead in (5.2.2) appears reasonable. In fact for the class of problem encountered here, the situation is even better than it first appears, because the appropriate off-diagonal elements of $L$ are those of the original matrix $A$. The result is established formally in the following theorem.

Theorem 5.1 Let $A$ be the matrix corresponding to the finite difference representation adopted in Appendix B, then decomposition of the matrix $A=LDL^\top$ given by

\[ l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} d_{kk} l_{jk} \quad \text{for } a_{ij} \neq 0 \]
= 0 \text{ otherwise}

and

\[ d_{ij} = \lambda_{ij}^{-1} \]

results in a lower triangular matrix \( L \), whose elements are

\[ \lambda_{ij} = a_{ij} \quad (i \neq j). \]

Proof. The proof is presented for the 7-point finite difference representation corresponding to the discretisation of the three spatial dimensional form. The result for fewer dimensions is established as portion of the larger proof.

For the 7-point finite difference representation, the elements \( a_{ij} \) of the matrix \( A \) that are non-zero can occur only in the following instances:

\[
\begin{align*}
  j &= i \\
  j &= i \pm 1 \\
  j &= i \pm s & \text{or} & i &= j \pm s \\
  j &= i \pm (s+t) & i &= j \pm (s+t)
\end{align*}
\]

where \( s \) and \( t \) are given by \( s=N_x \) and \( t=N_x N_y \), for the ordering of \( \Phi \) used here. Now, because no in-fill is permitted, unless a non-zero element occurs in the corresponding position in \( A \), the lower triangular elements \( \lambda_{ij} \) are zero except for the following three cases:
(i) \( i = j + 1 \),
(ii) \( i = j + s \),
(iii) \( i = j + s + t \).

The elements of \( L \) for case (i) are

\[
\mathcal{L}_{j+1}^j = a_{j+1}^j - \sum_{k=1}^{j-1} \mathcal{L}_{j+1}^k d_{kk}^j \mathcal{L}_{jk}^j.
\]

From the non-zero conditions applying to \( a_{ij} \), both \( \mathcal{L}_{j+1}^k \) and \( \mathcal{L}_{jk}^j \) can be non-zero only for \( k = j \). Because this combination is excluded from the summation it immediately follows that

\[
\mathcal{L}_{j+1}^j = a_{j+1}^j.
\]

The remaining cases (ii) and (iii) follow in a similar fashion.

A little thought, however, shows this property is lost for the conventional nine-point two-dimensional difference operator (or its three-dimensional equivalent).

5.3 IMPLEMENTATION OF ICCG WHERE DATA IS ORGANISED IN BLOCKS

The ICCG technique (Section 5.2) is described now, in a modified mathematical form, suited to the data block structure of the three-dimensional structure defined by (5.2.1). (Remember; it is only the data and not ICCG that is block structured.) In addition to the matrix \( A \) and vectors \( \phi \) and \( \xi \), six working vectors of length \( N_x N_y N_z \) are
introduced for convenience These are $t, d, r, v, u$ and $y$. The matrix
sub-blocks of $A$ are as described previously. The sub-blocks $L_{ij}$ and
$D_{ij}$ represent the corresponding submatrices appropriate to the
incomplete Choleski decomposition of $A$, which is presumed to be
computed previously. In addition the vector notation $\phi_k$ is used to
represent the $k^{th}$ subvector block of $\phi$ whose length is $N_x N_y$ while $\phi$
is considered to be ordered

$\phi = \{ \phi_1, \phi_2, \ldots, \phi_{N_z} \}$.

In the following description any improperly formed matrix or vector
operations are ignored.

---

SOLUTION OF $A \phi = s$ by ICCG

start perform incomplete LDL$^T$ decomposition of $A$
choose $\phi(0)$
initialize $t(0) = 0$, $d(0) = 0$, $\gamma(0) = 1$, $\alpha = \beta = 0$

loop $i = 1$ to Maximum iterations ($N_x N_y N_z$)

$t(i) = d(i-1) + \beta t(i-1)$
$\phi(i) = \phi(i-1) + \alpha t(i)$

if (any $\alpha t(i) > \varepsilon$) then set not converged flag

$z_1(i) = 0$

loop $k = 1$ to $N_z$

$t(k) = d(i-1) + \beta t(i-1)$
$z_{k+1} = z_{k+1}^\ast$
\[ \phi_{k+1}^{(i)} = \phi_{k+1}^{(i-1)} + \alpha_{k+1}^{(i)} \]

if (any \( \alpha_{k+1}^{(i)} > \varepsilon \)) then set not converged flag

if (\( k=N_z \) & converged) then finish.

\[ r_{k+1}^{(i)} = s_k - A_{kk} \phi_k^{(i)} - A_{k+1 k} \phi_{k+1}^{(i)} - r_k^{(i)} \]

\[ y_k^{(i)} = L_{kk}^{-1} (r_k^{(i)} - L_{k+1 k}^{(i)} y_{k-1}^{(i)}) \]

\[ r_k^{(i)} = A_{k+1 k} \phi_k^{(i)} \]

end loop \( k \)

\[ \beta_1 = \beta_2 = \beta_3 = \gamma^{(i)} = 0 \]

\( k=N_z \)

\[ \eta_k = L_{kk}^{-1} D_{kk}^{-1} y_k \]

loop \( \ell = 1 \) to \( N_z \)

\[ u_k = A_{kk} t_k + A_{kk-1} t_{k-1} + A_{k+1 k} t_{k+1} \]

\[ \beta_2 = \beta_2 + \frac{d_{k}^T u_k}{\phi_k} \]

\[ \beta_3 = \beta_3 + \frac{t_{k}^T u_k}{\phi_k} \]

\[ \gamma^{(i)} = \gamma^{(i)} + \frac{r_k^T \phi_k}{\phi_k} \]

\[ y_k = A_{kk} d_k + A_{k+1 k} d_{k+1} \]

\[ d_{k-1} = (L_{k-1 k-1})^{-1} (D_{k-1 k-1} y_{k-1} - L_{k-1 k} d_k) \]

\[ y_k = A_{kk} y_k + A_{k+1 k} d_{k+1} \delta_{k-1} \quad (A_{kk-1} \leftrightarrow A_{k+1 k}) \]

\[ \gamma_{k} = \gamma_{k} + \frac{A_{kk-1} d_{k-1}}{\phi_k} \delta_{k-1} \]

\[ \beta_1 = \beta_1 + \frac{d_{k}^T y_k}{\phi_k} \]

\[ \beta_2 = \beta_2 + \frac{t_{k}^T y_k}{\phi_k} \]

k = k - 1

end loop \( \ell \)

\[ \beta = \frac{\gamma^{(i)}}{\gamma^{(i-1)}} \]

\[ \alpha = \frac{\gamma^{(i)}}{(\beta_1 + \beta(\beta_2 + \beta(\beta_3)))} \]

end loop \( i \)

end
5.4 COMPUTATIONAL CONSIDERATIONS

The mathematical description of MINI (Chapter 4) and the modified version of ICCG (Section 5.3) fail to deal with the real limitations placed on their implementations, namely those arising from the lack of sufficient directly addressable storage. Even with the recent trends of reduced memory costs arising from technological changes, it is not possible to procure sufficient fast storage to hold all the data required for large three-dimensional models. Nor is it realistic to anticipate such a situation will arise in the near future, because reactor designers will attempt to incorporate more detail in their models as computer facilities increase. To show how the implementations of ICCG and MINI deal with problems of storage management, the two algorithms are described in a manner very similar to their implementation in POW3D.

The philosophy behind POW3D involves the assumption, that, for a particular energy group, sufficient fast storage always is available for any complete two-dimensional model, or for a two-dimensional subsection of a three-dimensional model. This is practical in view of present day scientific computer facilities. It is only in the third spatial dimension (and energy), that secondary data transfers become necessary. Suitably dimensioned arrays, to hold the two-dimensional forms in POW3D are generated dynamically under the VARRAY system [Cox in Pollard 1978]. This overcomes an otherwise basic weakness of FORTRAN which requires that the storage be allocated explicitly, a
significant draw back in a general purpose code. Should sufficient primary space within the requested region be unavailable to accommodate the two-dimensional section of the user's reactor model, the calculation is terminated. The user either may request a larger physical region or simplify his model.

The availability of virtual storage systems at first sight appears to take the responsibility for handling data transfers and the management of storage away from the scientific programmer. A blind acceptance of virtual storage presents problems because such systems are not universal. This affects the portability of any code which relies upon them exclusively. Limitations in the address range within such systems may present difficulties for very complex reactor calculations, particularly when the number of energy groups modelled is large. (The present limit for the IBM system is 8 Mbytes). In addition it is probably best to organise the data transfers oneself. This is particularly so when the most efficient organisation is known in advance. The alternative is reliance on a more general paging algorithm. In some cases these swap data in an inappropriate fashion. Such algorithms tend to retain pages used most recently and swap out the others. It is frequently the page that was dispatched to the disc, (because it was used some time ago) rather than one just used (and still in memory) which is required next in iterative approaches.
When initial planning for POW3D commenced, the AAEC operated an IBM 360/65 with 2 Mbytes of primary storage. It was known that this machine would not be replaced by an IBM 3031 (with virtual facilities) for at least three years. By then it was anticipated a significant proportion of the code would be operational. It was this knowledge, plus the desire to maintain a code with some portability to non-IBM compatible installations which greatly influenced the data structures chosen. To achieve this, the code had to run without relying on virtual memory. It was decided, however, not to ignore the potential of the virtual systems for later generation computers, particularly as this would be the environment in which POW3D would earn its keep. A method that attempts to exploit the best of the real and virtual worlds evolved from these constraints.

The language FORTRAN IV was selected as the principal language in which to develop POW3D for three reasons. The first is its acceptance by nuclear code users, and because in 1976, when this work commenced, there was virtually no reasonable alternative (for IBM installations at least). It also was envisaged that some existing routines (in FORTRAN) from older nuclear codes, could be introduced directly into POW3D. There were a number of drawbacks, however, concerned with the use of existing versions of IBM's FORTRAN compilers. These were primarily

(i) the inability to support free format input,
(ii) the limited capability of IBM's direct access routines,
(iii) the general slowness of IBM FORTRAN's Input/Output.
Because of the large amount of data (and its varying characteristics) which must be supplied by the physicist who eventually runs the code, 'free format' input is a most attractive scheme to reduce data preparation errors. This was achieved through use of SKAN [Pollard 1978], a system invoked by subroutine calls, in common use by scientists at the AAEC's Research Establishment.

The latter two problems presented more difficulty. Because the overhead in preformatting direct access work files with IBM FORTRAN is enormous, a series of very fast I/O routines designed for direct access files [Cawley 1977] were used instead. These are invoked by subroutine calls from FORTRAN and completely bypass the IBM FORTRAN I/O package (IBCOM).

The internal VIRTUL system of POW3D [Pollard 1976] was designed around these fast I/O routines written in assembler language. The VIRTUL system allows data to be fetched or stored as though it is being retained on secondary storage devices in a direct access fashion. If sufficient primary storage is available, the data is held there and the whole operation is transparent to the user of the system. The basic unit of data supported corresponds to one (x,y) plane full of information (an array of size \( N_x N_y \) in either single or double precision arithmetic) arising from the finite difference idealisation of the z coordinate direction. All the data transfers
are controlled through three indices; the type of data (i.e. coefficient, flux, source, \( \gamma \)'s), the relevant \( (z) \) plane indicator, and, if appropriate, the energy group.

Under the VIRTUL system all the spare available real fast storage for the region size specified is divided up into \((x,y)\) planes, and then allocated amongst the fluxes, coefficients etc., on a basis which the code writer may vary dynamically. Once all the available real store is used up, the remainder of the data is assigned to fast disc storage. The code writer is no longer concerned with knowing the ultimate destination or source of the data. An attempt by the code (under VIRTUL) to read (or write) data which has been assigned to real storage, simply results in a fast move operation transferring data to or from the appropriate POW3D working array. For data associated with disc storage, the fast I/O routines transfer the data directly between the user arrays and disc. This approach functioned successfully under the older machine architecture of the IBM 360/65.

For the newer architecture three successful approaches are available under the VIRTUL system. These do not require any alteration to the code itself.

(i) An arbitrarily sized region may be specified (from say 500K to the total real storage available to the user), and the data transfers performed under the same system with which the older architecture worked.

(ii) A small region may be specified and the virtual I/O facilities
of the IBM 3031 instead allowed to perform the data transfers when the VIRTUL system requests a transfer.

(iii) The specified region may be set equal to the complete range of virtually addressable storage (8 Mbytes) and the page interrupt system allowed to accommodate the data transfers.

No matter which option is in force, the performance of the code naturally benefits when the computer is lightly loaded on large problems.

Unless sufficient fast storage is available to hold a minimum number of working planes of information, it is inefficient to commence the calculation. For mathematical manipulation the code requires a number of working arrays (based on the x,y plane). These are in addressable memory. Prior to IBM's virtual operating system (MVS) the necessity of their presence prevented the code from commencing a calculation when memory was unavailable. Under MVS the requirements are identical, but the calculation may commence even if there is little chance of obtaining the memory. With such an environment there is always the possibility of page 'thrashing', when the machine is heavily loaded. For this reason timing comparisons reported in later chapters were performed over weekends when the loads were light.

The different iterative algorithms are implemented in an environment where the amount of directly addressable memory required for their efficient working is the same. They each use the space in
very different ways. For MINI and SLOR double precision arithmetic was deemed to be necessary, while at the design stage ICCG was expected to perform successfully with single precision operations in most sections. (The LDL\(\text{T}\) incomplete decomposition, however, is performed in double precision.) Because ICCG requires more work vectors than either SLOR or MINI, there are spatial as well as timing considerations for avoiding double precision arithmetic as much as possible.

5.5 A FULL COMPUTATIONAL DESCRIPTION OF THREE-DIMENSIONAL ICCG

A practical implementation of the ICCG algorithm is presented. The implementation is for the seven-point finite difference scheme, where the block structure (5.2.1) is used. The algorithm is based on the modified mathematical presentation of Section 5.3. It is possible to present the algorithm using a computer language, but this tends to obscure the essential features of the mathematics through the lack of suitable mathematical notation. Consequently, a description is given in a format that lies somewhere between a high level language and a mathematical definition. To describe the process adequately, the following data structures and notation are introduced.

\(\phi\) represents a storage vector (in VIRTUL memory) containing the three-dimensional flux, ordered in terms of the directly transferable subvectors.
\[ \Phi = \left\{ \phi_k ; k=1,2,\ldots,N_z \right\} . \]

Each \( \phi_k \) is itself ordered

\[ \phi_k = \left\{ \phi_{jk} ; j=1,2,\ldots,N_y \right\} \]

and at the lowest level

\[ \phi_{jk} = \left\{ \phi_{ijk} ; i=1,2,\ldots,N_x \right\} . \]

\( s \) represents the right hand side of (5.2.1), is stored in VIRTUL memory, and is structured like \( \Phi \).

\( e, d, t, y \) are work vectors held in VIRTUL memory also structured like \( \Phi \).

\( A \) represents the matrix of (5.2.1) held in VIRTUL memory. The matrix is block structured like (5.2.1) and each block may be identified directly and transferred; eg \( A_{k^1}, A_{k^2}, \) or \( A_{k^3} \).

\( e, d, t, x, z, x^*, r, b, u, v \) are temporary working vectors kept in real computer memory. They each represent a plane full of information \( (N_x \times N_y \) locations). \( e, d, \) and \( t \) should not be confused with the larger vectors of the same name held in VIRTUL memory). Nine vectors are used for the convenience of description, however, \( u, e \) and \( x^* \) may share the same physical locations, as may \( v \) and \( b \). Different names are selected, to assist understanding only. \( x, x^* \) and \( b \) are double precision vectors.

\( A_1, A_2, A_3 \) are three computer memory matrices each capable of holding an individual block submatrix of (5.2.1). To save storage and time throughout this work, only non-zero matrix elements are actually stored and transferred.
represents a fetch operation from VIRTUL to real memory. In each fetch operation, the information for a full \((x,y)\) plane is transferred to the appropriate vector or matrix. The transfer may be achieved by one of the following mechanisms within the VIRTUL system:

(i) a direct access read from disc storage,
(ii) a move from other directly addressable memory,
(iii) a move from virtual memory.

The fetch operation applies both to block subvectors:

\[ x_\rightarrow \phi \]  (the \(i^{th}\) \((x,y)\) plane of \(\phi\) is fetched)

or block matrices:

\[ A_{1} \rightarrow A_{k} \]  (the \(k^{th}\) off-diagonal block is fetched).

\[ \rightarrow \] represents a storage operation from real to VIRTUL memory. It follows a similar convention to \(\leftarrow\).

\[ = \] represents computational assignment

\(\{t_{j}\}_{i}\) denotes the \(i^{th}\) element of the \(j^{th}\) subvector of \(t\).

\(\{A_{1}\}_{ij}\) denotes the \((i,j)^{th}\) element of the block submatrix currently being held in \(A_{1}\).

\(\text{diag}(A)\) is a vector whose components are the diagonal elements of \(A\).

\([x]\) denotes a matrix formed with \(x\) as its diagonal

\(\Leftrightarrow\) denotes equivalence of storage.

\(\triangle\) denotes the lower triangular portion of a matrix.

\(\nabla\) denotes the upper triangular portion of a matrix.

Throughout the description any improperly formed summations, vector and matrix operations or data transfers are ignored. When this scheme was coded in FORTRAN great care was taken to avoid any unnecessary operations and storage requirements were minimised.
*** SOLVES $A\phi = s$ ***

*** ICCG 3-D IMPLEMENTATION ***

*** ASSUME ESTIMATES OF $\phi_k$ ($k=1,2,...,N_z$) HAVE BEEN MADE AND STORED ***

*** LDL$^T$ DECOMPOSITION OF A ***

\[
\begin{align*}
\text{loop } & \text{ k = 1 to } N_z \\
& A_1 + A_{k-1}, A_2 + Akk, A_3 + Ak+1 \\
& t \rightarrow \mathbf{e}_{k-1} \\
& \text{loop } i = 1 \text{ to } N_xN_y \\
& \mathbf{e}_i = \{\text{diag}(A_2)\}^2 - \sum_{j=1}^{i-1} (A_2)_{ij}^2 \mathbf{e}_j - (\text{diag}(A_1))_{i}^2 t_i \\
& \mathbf{e}_i = 1/\mathbf{e}_i \quad \text{ (D}_{ii} = 1/\mathbf{l}_{ii}) \\
& \text{end loop } i \\
& \mathbf{e} \rightarrow \mathbf{e}_k \\
& \text{(store diagonal D)} \\
& \text{end loop } k
\end{align*}
\]

*** CONJUGATE GRADIENT SECTION ***

\[
\begin{align*}
& \text{t = 0} \\
& \text{t} \rightarrow t_k, \text{ t d k } \quad k=1,2,...,N_z \\
& \alpha = \beta = 0 \quad \gamma(0) = 1 \\
& \text{loop } i = 1 \text{ to Maximum iterations} \\
& \mathbf{d} = \mathbf{d}_1, \mathbf{t} = \mathbf{t}_1, \mathbf{x} + \mathbf{\phi}_1 \\
& \mathbf{t} = \mathbf{d} + \beta \mathbf{t} \\
& \text{if (any } \mathbf{\alpha}_m/|x_m| > \varepsilon \quad m=1,2,...,N_xN_y \text{ then set not converged flag} \\
& \mathbf{x} = \mathbf{x} + \alpha \mathbf{t} \\
& A_2 + A_{11}, A_3 + A_{12} \\
& \mathbf{r} \rightarrow \mathbf{0} \\
& \text{loop } k = 1 \text{ to } N_z \\
& \mathbf{d} = \mathbf{d}_{k+1}, \mathbf{t} = \mathbf{t}_{k+1}, \mathbf{x} + \mathbf{\phi}_{k+1} \\
& \mathbf{t} = \mathbf{d} + \beta \mathbf{t}
\end{align*}
\]
\( t \rightarrow t_{k+1} \)

if \((\text{any } |\alpha_{m}||x^*| > \varepsilon; m=1,2,\ldots,N_x N_y)\) then set not converged flag

\( x^* = x^* + \alpha t \)

\( x^* \rightarrow \phi_k \)

if \((k \equiv N_z \text{ AND not converged flag unset})\) then finish

\( b \rightarrow s_k \)

\( r = b - A_2 x - A_3 x^* - \zeta \)

\( r \rightarrow \gamma_k \)

\( t \rightarrow y_{k-1} \)

\( \text{diag}(A_2) \rightarrow 1/\epsilon_k \) \hspace{1cm} \text{(fetch } \lambda_{i_1}; i=1,2,\ldots,N_x N_y) \)

\( d = \left[ \triangle(A_2) \right]^{-1} (r - A_1 t) \)

\( d \rightarrow y_k \)

\( A_1 = A_3 \) \hspace{1cm} \text{(}A_1 = A_{k+1} k \text{ by symmetry)}

\( A_2 \rightarrow A_{k+1} k+1 \), \( A_3 \rightarrow A_{k+1} k+2 \)

\( r = A_3 x \)

\( x = x^* \)

end loop \( k \)

\( \beta_1 = \beta_2 = \beta_3 = \gamma(i) = 0 \)

\( k = N_z \)

\( r \rightarrow \gamma_k \)

\( \text{diag}(A_2) \rightarrow 1/\epsilon_k \) \hspace{1cm} \text{(fetch } \lambda_{i_1}; i=1,2,\ldots,N_x N_y) \)

\( d = \left[ \nabla(A_2) \right]^{-1} \left[ \text{diag}(A_2) \right] r \) \hspace{1cm} \text{(}L^T\left)^{-1} D^{-1} \)

\( d \rightarrow d_k \)

loop \( \ell = 1 \) to \( N_z \)

\( t \rightarrow t_k \), \( r \rightarrow r_{k-1} \), \( u \rightarrow u_{k+1} \)

\( u = A_1 r + A_2 t + A_3 u \)

\( r \rightarrow r_k \)

\( \beta_2 = d^T u + \beta_2 \)

\( \beta_3 = t^T u + \beta_3 \)
\[ y(i) = y(i) + r^T d \]
\[ r = d_{k+1} \]
\[ y = A_2 d + A_3 r \]
\[ \text{diag}(A_2) = e_{k-1} \]
\[ r = y_{k-1} \]
\[ r = [\text{diag}(A_2)]^{-1} r \]
\[ A_3 = A_1 \]
\[ A_1 = A_{k-1} \quad k-2 \quad A_2 = A_{k-1} \quad k-1 \]
\[ x = \text{diag}(A_2) \quad \text{(save diagonal D)} \]
\[ \text{diag}(A_2) = 1/e_k \]
\[ x^* = [\text{diag}(A_2)]^{-1} (r - A_3 d) \]
\[ \text{diag}(A_2) = x \quad \text{(restore } A_2) \]
\[ v = v + A_1 x^* \]
\[ \beta_1 = \beta_1 + d^T v \]
\[ \beta_2 = \beta_2 + t^T v \]
\[ d = x^* \]
\[ d = d_{k-1} \]
\[ k = k-1 \]
\[ \beta = y(i)/y(i-1) \]
\[ \alpha = y(i)/(\beta_1 + \beta_2 \beta + \beta_3 \beta^2) \]
\[ \text{end loop } i \]
\[ \text{finish error condition} \]
\[ \text{end ICCG} \]
 POW3D offers ICCG as a possible means of solution for two-dimensional problems. The algorithm just outlined is more sophisticated than required for two-dimensional data structures, so separate code is used for efficiency. For such problems all the spatial information resides in directly addressable memory, while the solution is performed. The procedure used for two-dimensional calculations follows the conventional mathematical formulation (Section 5.2) and is not presented.

5.6 FULL COMPUTATIONAL DESCRIPTION OF 3-D MINI

The MINI process as outlined in Chapter 4 is a block procedure for solving the linear system of equations (5.2.1). In this respect it differs markedly from ICCG as implemented in POW3D. A description is now given of the double block process to solve the system (5.2.1).

The three-dimensional form of MINI requires the use of two distinct γ's at each grid point. One is related to the forward line (the 'in-plane' γ₁) and the other related to the forward plane (the out-of-plane' γ₂) as shown in Figure 5.1. (A third γₐ, not shown, is necessary for the energy group MINI). On the first pass it is possible either to set γ=0 and obtain a GS approach, or to calculate a somewhat arbitrary value of γ (perhaps from the last outer iteration). In POW3D, the default option is a GS start on the first outer iteration for each energy group of any new model. On subsequent outer iterations, knowledge remaining from the previous outer is used to obtain
$A\gamma_2$ required to improve the estimate of the forward plane flux point

$A\gamma_1$ required to improve the estimate of the forward flux point

γ relationships for 3-D problem

Figure 5.1
a true MINI iteration for the first inner iteration.

The inner-most (or two-dimensional) MINI block (like ICCG and SLOR) requires no data transfers in itself, because all the necessary data are established in computer memory by the outer block driver. A similar notation to that defined in Section 5.5 is employed for MINI. MINI, however, uses the following working vectors \( x, b, \Sigma, y, d \) and \( e \) of length \( N_xN_y \). In practice, however, it is more convenient to compute and use the elements of \( \Sigma \) as they are required, but the algorithm is more succinct in the form shown here.

*** SOLVES \( A\phi = s \) ***

*** 3-D MINI ALGORITHM ***

*** ASSUME ESTIMATES OF \( \phi_k \) (k=1,2,...,N_z) HAVE BEEN MADE AND STORED ***

\[ \text{loop } n=1 \text{ to Maximum iterations} \]
\[ b \leftarrow \Sigma_1 \]
\[ x \leftarrow \phi_1 \]
\[ A_2 \leftarrow A_{11}, A_3 \leftarrow A_{12} \]
\[ \text{loop } k=1 \text{ to } N_z \]
\[ y \leftarrow \phi_{k+1} \]
\[ d \leftarrow d_{k+1} \] (out of plane \( \delta \)'s for current plane \( k \))
\[ e \leftarrow \delta_{k+1} \] (out of plane \( \delta \)'s for next plane \( k+1 \))
\[ \text{for } k \neq N_z \] \[ \text{loop } \ell=1 \text{ to } N_xN_y \]
\[ \gamma_{\ell} = \min(|e_{\ell}|,|d_{\ell}|) \]
\[ \Gamma_{\ell} = \min(\gamma_{\ell}^2,|y_{\ell}|) \]
\[ \text{end loop } \ell \]
\[
\begin{align*}
\tilde{b} &= \tilde{b} - A_3 \tilde{y} + \Gamma^T A_3 \tilde{x} \\
\text{diag}(A_2) &= \text{diag}(A_2) + \text{diag}([\Gamma^T A_3]) \\
e &= \tilde{x} \\
d + \delta_k & \quad \text{(in plane } \delta\text{'s from previous pass of plane } k) \\

*** \text{ INVOKE PLANE MINI ROUTINE TO SOLVE } A \tilde{x} = \tilde{b} *** \\
\text{CALL PLANE MINI (} A_2 , \tilde{x} , \tilde{b} , \tilde{d} \text{)} \\

*** \text{ SOLUTION RETURNED IN } \tilde{x}, \text{ IN-PLANE } \delta\text{'s IN } \tilde{d} *** \\
d \rightarrow \delta_k & \quad \text{(store within plane } \delta\text{'s)} \\
\tilde{x} \rightarrow \phi_k & \quad \text{(store solution for } k\text{'th z plane)} \\
d &= \tilde{x} - \tilde{e} \\
d \rightarrow \delta^+ & \quad \text{(store out of plane } \delta\text{'s)} \\
\text{if (any } |d_x/e_x| > \varepsilon \text{) then set not converged flag} \\
A_1 &= A_3 , A_2 + A_{k+1} k+1 , A_3 + A_{k+1} k+2 \\
\tilde{b} &= \tilde{b} - \delta_k  \\
\tilde{x} &= \tilde{x} - A_1 \tilde{x} \\

\text{end loop } k \\
\text{if (not converged flag unset) then finish} \\
\text{end loop } n \\
\text{finish error condition} \\

\text{PROCEDURE PLANE MINI (} A , \tilde{x} , \tilde{b} , \tilde{d} \text{)} \\
*** A , \tilde{x} , \tilde{b} , \tilde{d} \text{ are block (line) addressable } *** \\

\text{loop } n: 1 \text{ to Maximum iterations} \\
\text{loop } j = 1 \text{ to } N_y \\
\text{for } j \neq N_y \text{ loop } i = 1 \text{ to } N_x \\
\bar{\gamma}_i &= \min(|\{d_{j+1}\}_i/\{d_j\}_i|, |\{d_j\}_i/\{d_{j+1}\}_i|) \\
\Gamma_i &= \min(\bar{\gamma}_i, |\{x_{j+1}\}_i/\{x_j\}_i|) \\
\text{end loop } i \text{ end loop } j
\[ b_j = b_{j-1} - A_{jj-1} x_{j-1} + \xi^T A_{jj+1} x_{j+1} \]
\[ \text{diag}(A_{jj}) = \text{diag}(A_{jj}) + \text{diag}([\xi^T A_{jj+1}]) \]
\[ d_j = (A_{jj})^{-1} b_j \quad \text{(solve tridiagonal system)} \]
if \( \text{any } |d_j| > \varepsilon \) then set not converged flag
\[ x_j = x_j - d_j \]
end loop j
if (not converged flag unset) then return
end loop n
return error condition
end procedure PLANE MINI
'and what a curious country it was. There were a number of tiny little brooks running straight across it from side to side, and the ground between was divided up into squares by a number of little green hedges, that reached from brook to brook.'

L. Carroll, 'Alice through the looking glass.'

6.1 INTRODUCTION

The multigroup neutron diffusion equations (3.3.5) are solved by a double iterative process (Chapter 5). The first (or inner step) effectively solves an inhomogeneous system of linear equations $A\phi = S$. The second (or outer step) involves the determination of the principal eigenvalue and eigenvector of $A\phi = \frac{1}{\lambda} F\phi$ by an iterative process (Section 3.3). The techniques of region rebalancing, as discussed in this chapter may be applied to accelerate further the convergence of either stage.

The methods of rebalancing are described from an historical perspective. Known properties of the reduced coarse mesh rebalance equations are reported, and a rather restricted justification for the effectiveness of the method is postulated in this chapter. Fourier analysis performed on some practical matrices is presented to support this in Chapter 7. Finally an account of its effectiveness with real reactor problems is reported in Chapter 9.
6.2

A major motivation for this section is to investigate the effect rebalance procedures have on accelerating convergence when newer forms of iteration are employed. Reactor physicists have known for a long time that the performance of more traditional iterative schemes is enhanced by coarse mesh rebalancing. McCallien [1976] discusses several working codes in which it has been gainfully employed for diffusion theory. Applications exist also in transport theory [Khalil et al 1979].

6.2 A VARIATIONAL APPROACH

The early attempts in reactor physics at what became known as coarse mesh rebalancing, involved simple procedures devoid of any mathematical sophistication. Probably the first use of such a correction to the neutron flux is reported by Roe [1954], who essentially sought a simple renormalisation factor to alter the magnitude of the flux estimate. The rebalance factor $R$ for $\phi(r)$ being obtained by estimating

$$ R = \frac{\iint sdVdE}{\iint (-\nabla \cdot \nabla \phi + \sigma_r \phi) dVdE}. $$

Following this, more elegant approaches with spatial variation in the rebalance factor were formulated directly from variational principles [Wachspress 1966]. Later more extensive forms of rebalancing became possible with the use of the weighted residual method rather than a specific variational formulation of the problem [Nakamura 1970, 1971,
For the original variational approach a system of $N$ linear equations is considered written in matrix form as

$$A \phi = \xi ,$$

(6.2.1)

where $A$ is obtained from a linear operator associated with the multigroup diffusion equation, and $\xi$ represents a known right hand side source term, formed from the multigroup equations producing the inhomogeneous discretisation (6.2.1). A corresponding adjoint system of equations

$$A^* \phi^* = \xi^*$$

(6.2.2)

exists. $A$ and $A^*$ are not necessarily the same. As is discussed later in this chapter they frequently represent self-adjoint blocks of the multigroup diffusion equations.

The solution vectors $\phi$ and $\phi^*$ are stationary nodes of the Roussopolos variational function

$$J(\phi^*,\phi) = \langle \xi^*,\phi \rangle + \langle \phi^*,\xi \rangle - \langle \phi^*,A\phi \rangle ,$$

(6.2.3)

where $\langle , \rangle$ denotes the inner product of the two vectors (corresponding to a volume integral in the case of the continuous form). Approximate
solutions to (6.2.1) and (6.2.2), (themselves approximations to the continuous form), may be obtained by expanding \( \phi \) and \( \phi^* \) in terms of a set of \( I \) appropriate trial (or basis) vectors \( x_i \) and \( x_i^* \), with arbitrary initial estimates included of \( \phi_0 \) and \( \phi_0^* \) respectively. The expansions are

\[
\phi = \phi_0 + \sum_{i=1}^{I} a_i x_i
\]

and

\[
\phi^* = \phi^*_0 + \sum_{i=1}^{I} a^*_i x^*_i
\]

The expansion coefficients are obtained by equating the partial derivatives of \( J \) with respect to these coefficients to zero

\[
\frac{\partial J}{\partial a_i} (\phi^*, \phi) = 0, \quad \frac{\partial J}{\partial a^*_i} (\phi^*, \phi) = 0,
\]

where \( i = 1, 2, \ldots, I \), and then solving the resulting set of equations. A selection of \( M \) orthogonal subspaces of \( \mathbb{R}^n \) is chosen with total dimension \( < n \), and the \( I \) trial vectors are defined over the subspaces so that the \( i^{th} \) trial vector \( x_i \) is expressed as

\[
x_i = x_{1i} \oplus x_{2i} \oplus \cdots \oplus x_{Mi}
\]

where \( x_{ji} \) denotes the projection of \( x_i \) onto the \( j^{th} \) subspace. This
necessitates the introduction of $M$ parameters for each of the $I$ basis vectors $x_i$ and the approximations for $\phi$ and $\phi^*$ are written

$$\phi = \phi_0 + \sum_{i=1}^{I} \sum_{j=1}^{M} a_{ij} x_i^*$$

and

$$\phi^* = \phi_0^* + \sum_{i=1}^{I} \sum_{j=1}^{M} a_{ij}^* x_j$$

respectively. The $a_{ij}$ in this sense are a set of $I.M$ coefficients and should not be confused with the matrix elements $a_{ij}$ of (6.2.1).

In addition to the source problem (equations 6.2.1 and 6.2.2), solution of the eigenvalue problem

$$A \phi = \frac{1}{\lambda} F \phi$$

and its adjoint may be accelerated by the variational approach. In this case the functional (6.2.3) is reduced as a result of writing the eigenvalue problem in the homogeneous form $A \phi - \frac{1}{\lambda} F \phi = 0$ to

$$J(\phi^*, \phi) = \langle \phi^*, A \phi - \frac{1}{\lambda} F \phi \rangle.$$ 

Again an approximate solution to (6.2.7) may be found through the use of trial functions and the relationship (6.2.4). In each case the original problem in $N$ unknowns has been reduced to one involving $I.M$ unknowns, and in reactor physics parlance such a reduced system is
known as the coarse mesh approximation. For the source form of the problem, application of (6.2.4) produces two reduced linear systems

\[ A \tilde{a} = \kappa \quad \text{and} \quad A^* \Phi = \kappa^* \quad , \tag{6.2.8} \]

where the elements of the matrix $A$ are

\[ a_{ij} = \langle x_i^*, A x_j \rangle \quad i=1,2,\ldots,I.M; j=1,2,\ldots,I.M \quad , \]

and those for $\kappa$ are

\[ \kappa_i = \langle x_i^*, s - A \phi_0 \rangle \quad i=1,2,\ldots,I.M \; . \]

In a similar manner the eigenvalue problem (6.2.7) is reduced to

\[ A \tilde{a} = \frac{1}{\lambda} \tilde{J} \tilde{a} - \kappa \quad , \]

and

\[ A \Phi = \frac{1}{\lambda^*} \tilde{J} \Phi - \kappa^* \quad , \tag{6.2.9} \]

where the elements of $A$, $\tilde{J}$ and $\kappa$ are given by

\[ a_{ij} = \langle x_i^*, A x_j \rangle \quad , \]
\[ f_{ij} = \langle \xi_i^*, Fx_j \rangle \]

and

\[ k_i = \langle \xi_i^*, A\phi_0 - \frac{1}{\lambda}\phi_0 \rangle \quad i=1,2,\ldots,1.M; \quad j=1,2,\ldots,1.M \]

Unfortunately the vector term \( k \) of (6.2.9) seems to indicate an inhomogeneous structure is generated from the homogeneous eigenvalue problem. When \( \phi_0 \) and \( \phi_0^* \) are chosen to be zero vectors (such as occurs all the time in this work for multiplicative rebalancing), the inhomogeneous term vanishes. For additive rebalancing further consideration of this phenomenon is delayed until Section 6.4.

Techniques based on rebalancing to accelerate the convergence of linear systems have had a long acceptance in reactor physics. Wachspress [1965, 1966] used coarse mesh techniques to accelerate the inner and outer iterations of the discrete multigroup neutron diffusion equation. Outside the nuclear field until recent times, the application of coarse mesh techniques was neglected apart from several notable exceptions such as Southwall [1935, 1940, 1946], and de la Vallee Poussin [1968]. Now, however, a somewhat related concept known as multigrid methods [Brandt 1977a, 1977b, 1979; Nicolaides 1977, 1979; McCormick 1979] has considerable potential. The multigrid methods, however, differ from the coarse rebalance techniques, particularly in that the former are not just two-step procedures.
It is interesting to speculate on this lack of interest elsewhere. It was probably the need to handle effectively the additional dimension of energy (c.f. Roe [ibid.]) which encouraged their early trial and acceptance in reactor physics, despite the considerable overhead in their implementation. All early experimentation was undertaken without a rigorous mathematical analysis to explain why the method might indeed work.

The value of coarse mesh rebalancing in practical nuclear studies is widely known and many variations in its form have arisen. At the AAEC a technique known as the SOKOLOV method was examined by Pollard [1968]. This is an iterative procedure which instead of directly satisfying the functional (6.2.3), achieves improvement through an averaging of functional corrections. One particular form of this method, for a particular matrix splitting and selection of dual bases vectors was incorporated into the code POW [Pollard 1974]. This method has been extended here to accommodate three spatial dimensions and energy, and was the first rebalance method attempted in the POW3D code. The version implemented corresponds to the disjunctive partitioning and region balance scheme discussed in Section 6.4.

Solution of (6.2.8) and (6.2.9) requires the appropriate matrices $A$ be non-singular. The non-singularity has been demonstrated for specific instances and choice of basis vectors [Wachspress 1966]. Most importantly, non-singularity of $A$ guarantees a unique solution, a property that will be exploited for several theorems in Section 6.7.
6.3 WEIGHTED RESIDUAL FORMULATION

The variational formulations may be approached through the rather more general framework of weighted residuals. Much of the pioneering work with the weighted residual approach in CMR is due to Nakamura [ibid.].

Once again an approximation of the form (6.2.6) is used for both the source (6.2.1) and eigenvalue problem (6.2.7). With this approximation the residual for the source problem is

\[ r = A(\phi_0 + \sum_{i=1}^{I} (\sum_{j=1}^{M} \oplus a_{ij}x_{ji})) - s, \quad (6.3.1) \]

and for the eigenvalue problem is

\[ r = A(\phi_0 + \sum_{i=1}^{I} (\sum_{j=1}^{M} \oplus a_{ij}x_{ji})) - \frac{1}{\lambda} F(\phi_0 + \sum_{i=1}^{I} (\sum_{j=1}^{M} \oplus a_{ij}x_{ji})). \quad (6.3.2) \]

The I.M coefficients \( a_{ij} \) are determined by making \( r \) orthogonal to a suitable set of weight vectors \( w_\ell \) (\( \ell = 1,2,\ldots, I.M \)) and obtaining the linear systems

\[ \sum_{i=1}^{I} \sum_{j=1}^{M} \langle w_\ell, A(\phi_0 + x_i) \rangle a_{ij} = \langle w_\ell, s \rangle, \quad (6.3.3) \]

and
for the source and eigenvalue problems respectively, where $x_i$ are defined by (6.2.5).

The weighted residual method allows sets of more appropriate basis functions to be introduced with weighting vectors that can be of significantly less computational complexity to achieve a better approximation for $\phi$ with reasonable economy. For convenience and to reflect the trial functions implemented in the nuclear code POW3D, attention is restricted to the case of $I = 1$; and henceforth $x_{1j}$ is written $x_j$.

Several varieties of basis vectors are available. The number of combinations they form with appropriate weighting vectors is considerable. First, the case of $\phi_0 = 0$ in (6.2.6) is considered; this choice gives

$$ \hat{\phi} = \sum_{j=1}^{M} a_j x_j ,$$

(where now, the direct sum notation is dispensed with and the $M$ subspaces are no longer necessarily mutually orthogonal). This form of approximation can be classified as a multiplicative coarse mesh
rebalance scheme, provided the $x_j$ are related directly to the previous estimate $\phi_0$ of $\phi$. To achieve this relationship, $M$ partitioning operators $P_j$ are introduced so that

$$x_j = P_j \phi_0 \quad (j=1,2,\ldots,M), \quad (6.3.5)$$

where the $P_j$ are chosen so that

$$\phi_0 = \sum_{j=1}^{M} P_j \phi_0,$$

i.e.

$$I = \sum_{j=1}^{M} P_j.$$

The $M \times M$ systems of linear equations (6.3.6 and 6.3.7) arising from this approximation and the weighted residual method are somewhat simpler than (6.3.3) and (6.3.4). They are

$$\sum_{k=1}^{M} \langle w_k, AP_k \phi_0 \rangle a_k = \langle w_k, \xi \rangle, \quad (6.3.6)$$

and

$$\sum_{k=1}^{M} \langle w_k, FP_k \phi_0 \rangle a_k = \left( \frac{1}{\lambda} \right) \sum_{k=1}^{M} \langle w_k, \xi \rangle a_k \quad (\lambda=1,2,\ldots,M). (6.3.7)$$
for the source and eigenvalue problems respectively.

The partitioning operator $P_j$ divides the spatial $(x, y, z)$ and energy dimensions up in some suitable form (hereafter the domain may be considered four-dimensional). In this work the partitioning is done so that the non-zero elements of $x_j$ are connected in a geometric sense. What constitutes the best choice of division possible, is open to conjecture. It seems sensible, however, to select the lines of spatial division in such a manner that their positioning reflects both the type of materials in the reactor under study and the form of the basis functions. Experience with practical neutron flux shapes is of enormous benefit in the allocation of the divisions. The code POW3D automatically positions the coarse mesh lines of subdivision so that the reactor fuel is the prime determinant of their position. Non-fissile materials and interfaces between materials are the secondary criteria considered for their allocation.

The choice of their location should vary with the degree of mathematical sophistication involved in the partitioning process. The code allows the knowing physicist to force his estimate of optimal coarse mesh positioning, and this proves helpful for experimental purposes. The automatic generation in POW3D was designed to accommodate the first form of partitioning described (i.e. the disjunctive type).
Three forms of partitioning are available in the code. For the first two, the approximation is of the multiplicative form (6.3.5). In the third, the vector $\phi_0$ is readmitted to give an additive form of approximation. The partitioning operator as used in (6.3.5), instead simply defines a set of prescribed vectors generally unrelated to the solution $\phi_0$ obtained so far.

In all cases the forms of partitioning selected generate sets of M basis vectors which are linearly independent. For the disjunctive methods the basis vectors are also mutually orthogonal and the $P_j$ are the projection operators.

6.4 PARTITIONING

A. Disjunctive partitioning

The spatial domain $D$, over which the discretised form of the diffusion equation is defined is divided into M spatial subdomains $D_m$. Any vector $\Phi$ of length N (where each element of the vector corresponds to a value of $\phi(r,E)$, defined over $D$) is partitioned so that each element of $\Phi$ belongs to one and only one subdomain $D_m$. Such subdivisions clearly satisfy the orthogonality condition, and the basis vectors are linearly independent.
The partitioning operator $P_j$ is simply a diagonal $NxN$ matrix, where certain elements are unity. For a one-dimensional problem, $P_j$ has a single block unit matrix structure. This structure is not necessarily preserved for higher dimensional problems as the non-zero elements may become scattered, because the structure then depends upon the spatial ordering of the elements of $\phi$. A two-dimensional geometric representation of disjunctive partitioning is given in Figure 6.1. The coarse grid divides the fine grid into four subdomains $D_1, D_2, D_3$ and $D_4$.

The coarse mesh is imposed upon the fine geometric mesh in such a way that it does not pass through any fine mesh point because each fine mesh point must be a member of only one subdomain $D_m$. Consequently the operations $P_1\phi_0$, $P_2\phi_0$, $P_3\phi_0$ and $P_4\phi_0$ simply pick out the values of $\phi_0$ defined over the four subdomains $D_1$, $D_2$, $D_3$ and $D_4$.

When a coarse grid is imposed in the fourth dimension (energy) a slightly different situation to that of Figure 6.1 emerges, because the discretisation in energy is achieved a little differently; (as shown in Figure 6.2).

The approximate solution sought (6.2.6), now is expressed as
Disjunctive partitioning coarse grid for 4 subdomains $D_m$

Figure 6.1
A disjunctive partitioning of the fourth dimension (energy)

Figure 6.2
It is obvious that the acceleration process in effect is seeking a scalar constant $a_j$ for each of the $M$ subdomains $D_m$. This is used to rescale the flux estimate $\phi_0$ over that subdomain. Hence the method is classified as a multiplicative form of rebalancing.

The procedure obviously will change the degree of continuity exhibited by $\phi(r_j, E)$ at the coarse mesh boundaries (which may or may not coincide with a material boundary) before the rebalance was applied. Significant variations in the neutron flux are more likely to occur near the physical boundaries, and this supports the rationale of positioning the coarse grid at such locations.

A more sophisticated partitioning scheme, commonly known as 'pyramid' partitioning is considered in an attempt to

(i) overcome the introduced discontinuity in $\phi$ at the coarse mesh boundaries, and

(ii) apply a more realistic form of correction across the subdomain $D_m$.

The impetus behind such a choice undoubtedly is due to developments in the approximation of functions, such as used in spline theory [Birkhoff and de Boor 1965; Rice 1969] and finite element
B. Multiplicative 'pyramid' partitioning

The spatial domain $D$ is divided again into $M$ spatial subregions $R_i$, but this time an element of $\phi_0$ may be common to more than one subregion (i.e. the eventual subdomains are not mutually orthogonal). The typical positions of coarse mesh lines for a simple two-dimensional geometry are shown in Figure 6.3. These divide the fine mesh to form 6 subregions $(R_i)$ which are combined to form 12 subdomains associated with the corners of $R_i$. Here any fine mesh point that lies on a coarse meshline of subdivision belongs to all the subregions associated with that line. The partitioning matrices $P_j$ ($j=1,2,...,M$) are again diagonal, but the non-zero terms are defined over the four adjacent subregions associated with each subdomain (for the two-dimensional form). For the partitioning operator defined over the domain $D_j$ (consisting of the four regions centred at the coarse grid mesh node coinciding with the $(k,\ell)$th fine mesh point) the non-zero diagonal elements of $P_j$ are defined by the two-dimensional analogue of the linear hat functions viz:

$$P_j(x,y) = \frac{(X_{k+1}-x)(Y_{\ell+1}-y)}{(X_{k+1}-X_k)(Y_{\ell+1}-Y_{\ell})}, \quad (6.4.2)$$

for $X_k \leq x \leq X_{k+1}$

and $Y_{\ell} \leq y \leq Y_{\ell+1}$
Coarse mesh grid for pyramid partitioning

Figure 6.3
\[
\frac{(X_{k-1} - x)(Y_{l+1} - y)}{(X_{k-1} - X_k)(Y_{l+1} - Y_l)}
\]
for
\[
X_{k-1} \leq x \leq X_k
\]
and
\[
Y_l \leq y \leq Y_{l+1}
\]

\[
\frac{(X_{k+1} - x)(Y_{l-1} - y)}{(X_{k+1} - X_k)(Y_{l-1} - Y_l)}
\]
for
\[
X_k \leq x \leq X_{k+1}
\]
and
\[
Y_{l-1} \leq y \leq Y_{l}
\]

These definitions (6.4.2) also identify indirectly the domain \( D_j \), as being that set of fine grid points associated with non-zero elements of \( P_j \). Again \( \sum_{j=1}^{M} P_j = 1 \). In passing, it is noted that the word 'pyramid' is not strictly correct due to the nonlinear nature of \( P_j(x,y) \) along the lines joining the points \((X_k,Y_l)\) and \((X_{k+1},Y_{l+1})\), however, its use is continued here even though the term pagoda is more
correct. It is obvious from (6.4.1) and (6.4.2) that because of the multiplicative nature of pyramid partitioning there are no first order discontinuities introduced in the 'improved' trial flux.

Both the disjunctive and pyramid forms of partitioning lead to an \( M \times M \) system of linear equations of the form

\[
\sum_{k=1}^{M} <w_k, A_P \phi_o> a_k = <w_k, s>, \quad (k=1, 2, \ldots, M),
\]

(6.4.3)

which may be expressed in matrix form as

\[
A^M a = s^M,
\]

(6.4.4)

where the superscript \( M \) is used to denote the multiplicative form.

A discussion on the properties desired for \( A^M \) and those actually achieved is delayed until Section 6.7.

C. Additive pyramid partitioning

In additive partitioning an improvement is sought to the trial flux \( \phi_o \) of the form

\[
\phi = \phi_o + \sum_{j=1}^{M} a_j x_j,
\]

with a subsequent saving of machine time for later iterations where
the trial vectors $x_j$ are no longer dependent on $\phi_0$. The choice of prescribed functions for $x_j$ is considerable. Settari and Aziz [1973] report on a single partitioning method (essentially that of de la Vallee Poussin [1968]), and extend it to a multiple partitioning method analogous to the form (6.2.5).

In this work, the piecewise continuous bilinear polynomials (hat functions) are selected as prescribed functions. These are defined over subdomains and are identical to those used with pyramid partitioning. Use of the additive form of correction with the weighted residual method leads to an $M \times M$ system of linear equations

$$
\sum_{k=1}^{M} \langle w_k, A x_k \rangle a_k = \langle w_k, s - A_0 \rangle (z=1,2,\ldots,M), \quad (6.4.5)
$$

which in turn is expressed in matrix terms as

$$
A^A a = s^A . \quad (6.4.6)
$$

For the eigenvalue problem $A \phi = \frac{1}{\lambda} F \phi$ the residual $r = A \phi - \frac{1}{\lambda} F \phi$ is weighted with $M$ weighting vectors to give

$$
\sum_{k=1}^{M} \langle w_k, A x_k \rangle a_k + \langle w_k, A \phi_0 \rangle = \quad (6.4.5)
$$
\[ = \frac{1}{\lambda} \sum_{k=1}^{M} \langle w_k, F x_k \rangle a_k + \langle w_k, F \phi_0 \rangle \quad (z=1,2,\ldots,M) \quad (6.4.7) \]

The homogeneous problem now appears in an inhomogeneous form (6.4.7). Nakamura [1976] gives one iterative scheme for solving (6.4.7). Of course the existing code cannot handle the inhomogeneous form directly. An alternative approach involves the introduction of an additional coefficient and subsequent weighting vector in the additive correction approximation to give

\[ \phi = a_0 \phi_0 + \sum_{j=1}^{M} a_j x_j . \]

This leads to an eigenvalue problem \( A A a = \frac{1}{\lambda} a \) of order \((M+1)\) where \( A_{ij} = \langle w_i, A \phi_j \rangle \) and \( j \omega = \langle w_i, F \phi_j \rangle \). This could be handled by the existing code, and because the solution of an eigenvalue problem is arbitrary by a multiplicative constant, the coefficient \( a_0 \) of \( \phi_0 \) could be rescaled to 1.

The prescribed trial vectors \( x_j \), for additive pyramid partitioning are identical to the diagonal of the corresponding pyramid partitioning matrix, (i.e. they are the \( n \)-dimensional piecewise linear polynomials), so for all \( j \)

\[ x_j = \left\{ \text{diag } P_j \right\} . \]

This leads to the following useful result.
Theorem 6.1  The matrix $A^A$ arising for the additive piecewise bilinear prescribed vectors is identical with that computed for multiplicative pyramid partitioning provided

(i) the same weighting functions are used, and

(ii) $\phi_0 = 1 = (1,1,\ldots,1)^T$ is used in the multiplicative form.

Proof  The $(k,k)^{th}$ element of $A^M$ is given by

$$A^M_{kk} = \langle w_k, A P_k \phi_0 \rangle,$$

and with $\phi_0 = 1$, $P_k \phi_0 = \text{diag} \{ P_k \}$ giving

$$A^M_{kk} = \langle w_k, A \text{diag} \{ P_k \} \rangle.$$

By (6.4.5) this is identical to $A^A_{kk}$, provided the weighting vectors are the same in each case.

What at first may seem a rather trivial and pointless result has considerable benefit when a nuclear code such as POW3D is used as an experimental tool as well as a production workhorse. By a few minor changes and the addition of one subroutine, the considerable body of code already written to implement the multiplicative correction form may be transformed to handle the additive case as well. (The implementation would be far from the most efficient possible, because
it requires approximately 5N additional multiplication operations to generate the two-dimensional form of the matrix $A^A$. It does, however, provide a convenient means of testing additive methods.

6.5 WEIGHTING

Four types of weighting vectors for both the source and eigenvalue form come to mind immediately:

(i) Galerkin weighting

$$w_A = P_A \Phi_0 \quad \text{(multiplicative)}$$

$$w_A = \text{diag}\{P_A\} \quad \text{(additive)}$$

(ii) Least squares weighting

$$w_A = AP_A \Phi_0 \quad \text{(multiplicative)}$$

$$w_A = A \text{diag}\{P_A\} \quad \text{(additive)}$$

(iii) Region balancing weighting

$$w_A = P_A^{1/2}$$

where $P_k$ is the disjunctive partitioning matrix
(iv) Pyramid weighting

\[ w_k = P_k \hat{1} , \]

where \( P_k \) is the pyramid partitioning matrix.

All combinations of basis and weighting vectors are possible. The discussion here, however, is limited primarily to those implemented within the code POW3D:

<table>
<thead>
<tr>
<th>Weighting</th>
<th>Partitioning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i) Region balancing</td>
<td>disjunctive partitioning</td>
</tr>
<tr>
<td>(ii) Region balancing</td>
<td>multiplicative pyramid partitioning</td>
</tr>
<tr>
<td>(iii) Region balancing</td>
<td>additive pyramid partitioning</td>
</tr>
<tr>
<td>(iv) Galerkin weighting</td>
<td>additive pyramid partitioning</td>
</tr>
</tbody>
</table>

Of these (i) was selected on historical grounds and ease of implementation, (ii, iii, iv) because they promised a greater return for the computational efforts involved in establishing the reduced system of equations, and (i, iv) following consideration of the difficulties of implementation or because of properties of the reduced matrix. In particular the combinations of Galerkin and pyramid weightings with the multiplicative forms were rejected on computational grounds, while matrix properties inherent with so many least squares formulations made that choice appear unattractive.
6.6 REBALANCING IN ENERGY FOR SOURCE PROBLEM

Some of the rebalancing techniques implemented and tested in POW3D are designed to achieve a rebalance only in the fourth dimension (energy). Energy rebalancing is handled through three separate approaches.

(i) At the outer iteration (Section 3.4), the coupled group source problem may be collapsed in a group sense, where energy simply is viewed as a fourth spatial variable. In this form disjunctive partitioning is the only feasible option available, because of the nature of energy discretisation (Section 6.4). At various times this form has been combined with both pyramid and disjunctive partitioning in the other spatial directions.

With the relatively small number of energy groups involved for most problems, there appears to be little advantage gained by such a rebalance. This is because the computational effort required to set up the rebalance equations is comparable to that expended in the absence of energy rebalancing and yet the time involved in solving the reduced equations is insignificant compared with the time spent in computing the reduced matrix (particularly for the pyramid forms of partitioning). Because of the limited advantages available with this option it has been removed from the current working version of POW3D.
(ii) A very simple yet effective form known as energy rebalancing is available. All the spatial (x, y and z) information is collapsed to a single point. This is achieved by a single disjunctive partitioning operation (i.e. $P_g = I (g=1,2,...,G)$) for each energy group equation in turn (and use of an associated weighting vector $w_g = 1$). Energy rebalancing is an operation which is performed once for every outer iteration. The rebalance operation is performed, when the first energy group of equation (3.3.5) with neutron upscatter into it is encountered.

In the case of the eigenvalue problem, after the fluxes for all energy groups have been rebalanced, they are renormalised to a predetermined source strength. Once again because of the relatively small number of energy groups there is no advantage to be had in further collapsing the energy groups onto themselves.

(iii) At the start of a specified outer iteration, any of the four forms of partitioning and weighting may be executed for the three spatial directions (x, y and z). The collapsed group equations are then solved as though an ordinary outer iteration was to be performed (note that the fine mesh equations are not solved at any stage during this part of process). The rebalanced group fluxes are used as estimates for the next outer iteration involving the fine geometry.
6.7 PROPERTIES OF THE COARSE MESH REBALANCE MATRICES

The coarse mesh rebalance equations, as generated in the code POW3D are of a considerably reduced order compared to the original fine mesh system. This is in keeping with a suggestion of Wachspress [1966] that the number of coarse mesh points in each coordinate direction be approximately equal to the square root of the number of fine mesh points in the same direction.

For a general purpose nuclear code, the only way to ensure that all the fine mesh systems of linear equations may be solved is through recourse to iterative techniques. The coarse mesh equations, however, are of a suitable order to warrant the employment of direct matrix solvers. The choice of one technique (to the exclusion of the other) is not expedient [Doherty 1980]. Despite the traditional hold iterative methods have in reactor physics problems, recent developments in the design and construction of efficient direct solvers [George 1977; George and Liu 1978; Duff and Reid 1979] may efficiently solve the coarse mesh problem where the sparsity of the matrix can be exploited. In POW3D, various solution processes for the coarse mesh system are available. Experimentation revealed that for particular forms of the reduced equations, matrix iterative techniques often worked despite the lack of any theory to guarantee their applicability. Failure of such techniques was sometimes observed, but more frequently their shortcoming was a very slow rate of convergence.
When iterative techniques are selected for both the coarse and fine mesh problems, there is an advantage to be gained should the solution subroutines be applicable to each system. Efficient use of common code requires that the different matrices have similar mathematical properties, particularly in so far as band structure is concerned, because only non-zero components are stored in POW3D.

The recursive use of iterative solution procedures is essential should any of the current two-step methods of coarse mesh improvement be incorporated into a multigrid procedure [Brandt 1977]. Such techniques require iterative methods at all stages of grid adjustment (with the possible exception of the coarsest grid).

To obtain some indication on the possible applicability of iterative methods, the properties of the coarse mesh systems of equations are reviewed. For simplicity, attention is restricted primarily to matrix equations of the inhomogeneous form $A\phi = s$ (i.e. a subset of the eigenvalue problem). The matrix $A$ may be either the full matrix corresponding to the left hand side of the multigroup eigenvalue problem (3.3.5) or a submatrix of the multigroup form corresponding either to a single energy group or a coupling of energy groups. The properties of $A$ vary depending upon which aspect of the neutron diffusion equation it represents.
When the matrix $A$ corresponds to the full multigroup equation, or represents a general coupling of groups from the full form, then $A$ is

(i) real,

(ii) diagonal elements are positive, off-diagonal elements are zero or negative,

(iii) column diagonally dominant.

For the instance where all energy groups involve upscatter $A$ is also irreducible.

When $A$ corresponds to the submatrix appropriate to a single energy group equation (or for a certain restricted coupling of energy groups) it is

(i) real symmetric,

(ii) irreducible,

(iii) diagonal elements are positive, off diagonal elements are zero or negative,

(iv) diagonally dominant (row and column).

The matrix properties of the reduced matrices $A^M$ and $A^A$ vary. They reflect those of the original matrix $A$, the type of partitioning and the type of weighting employed. Much of the original knowledge about the reduced matrices is due to Froehlich [1967], who established theorems appropriate to the method as formulated by Wachspress (which is basically a Galerkin technique) and for a rebalance weighting scheme. Some of the results have been strengthened by Nakamura [ibid.] and extended to different schemes. The properties now are
discussed at some length and an attempt is made to be rigorous in their formulation. Attention is restricted now to the single group equation or the full energy group system involving upscatter for all groups.

A. Disjunctive partitioning and region balance

For $\phi_0 > 0$ with use of disjunctive partitioning and region balance weighting $A^M$ has the following properties for the full multigroup case;

(i) $A^M$ has positive diagonal elements and zero or negative off-diagonal elements,
(ii) $A^M$ is irreducible,
(iii) $A^M$ is column diagonally dominant \( (6.7.1) \)

The non-positive property for off-diagonal elements follows from consideration of the $(\xi,k)^{th}$ element of $A^M$

\[
A^M_{\xi k} = \langle w_\xi, A p_k \phi_0 \rangle
\]

\[
= \sum_{i=1}^{N} \text{diag} \{p_i\} \sum_{j=1}^{N} a_{ij} \text{diag} \{p_k\} \phi_{oj} ,
\]

which for the partitioning operators under consideration

\[
= \sum_{i \in D_\xi} \sum_{j \in D_k} a_{ij} \phi_{oj} , \quad (6.7.2)
\]
where \( \Sigma \) signifies summation associated with those rows of the matrix \( A \) corresponding to points in the \( \xi \)th subdomain of \( D_\xi \) of \( D \).

With the disjunctive subdomains and for \( \xi \neq k \) all positive elements \( a_{ij} \) are excluded and the non-positive nature of the off-diagonal elements is established. For \((k,\xi \pm 1)\) and \((\xi \pm 1,\xi)\) the structure of \( A \) ensures (6.7.2) is strictly negative (for properly formed subregions) and its irreducibility follows because the directed graph of \( A^M \) is strongly connected [Varga 1962]. Property (iii) is demonstrated from a column summation over an arbitrary column \( k \).

\[
\sum_{\xi=1}^{M} \langle w_\xi, A P_k \phi \rangle
= \langle \sum_{\xi=1}^{M} \{ \text{diag } P_\xi \}, A P_k \phi \rangle
= \langle 1, A P_k \phi \rangle
= \sum_{i=1}^{N} \sum_{j \in D_k} a_{ij} \phi_{oj}
= \sum_{j \in D_k} \sum_{i=1}^{N} a_{ij}.
\]

Every \( \sum_{i=1}^{N} a_{ij} \geq 0 \), because \( A \) is column diagonally dominant, and so consequently is \( A^M \). Column dominance coupled with non-positive off-diagonal elements and strictly negative elements about the
diagonal guarantees that the diagonal elements of $A^M$ are positive.

Because the requirement of row diagonal dominance may be replaced with column diagonal dominance [Faddeeva 1959], the properties (6.7.1) are sufficient to guarantee convergence of the Gauss-Seidel type methods [Varga 1962; Young 1971]. These are sufficient grounds for expecting MINI to converge also, and numerical experimentation supports this supposition [Barry and Pollard 1978], as well as demonstrating MINI to be a very efficient solver of such systems. Consequently the direct solution process was avoided for this combination of partitioning and weighting in POW3D. With MINI the convergence accuracy in the iterative solution of the rebalanced inner fine mesh system was taken to be proportional to that expected of the current outer iteration.

The lack of symmetry of $A^M$ (even for the restricted case when $A$ itself is symmetric) follows from a consideration of $A^M_{zk}$ and $A^M_{kz}$;

$$A^M_{zk} = \sum_{i \in D_\ell} \sum_{j \in D_k} a_{ij} \phi_{ij}$$

$$A^M_{kz} = \sum_{i \in D_k} \sum_{j \in D_\ell} a_{ij} \phi_{ij}$$

and when the arbitrary nature of the trial flux $\phi$ is considered symmetry cannot be expected. Consequently the particular conjugate gradient code implemented in POW3D cannot be used for the coarse mesh
system of equations.

The properties of $A^M$ are extended in the next theorem for the case of additive disjunctive prescribed functions and region balance.

**Theorem 6.2** For $\phi_0 > 0$ then $A^A$ with disjunctive partitioning and region balance has the same properties for the full multigroup case, as $A^M$ (namely 6.7.1). For the case of any single energy group, $A^A$ is symmetric and hence row diagonally dominant.

**Proof.** The proof to the first part is obtained from the previous result, through the use of Theorem 6.1. (This result can be extended immediately to the case of any partitioning and weighting.) For the case in which $A$ arises from a single energy group

$$ A^A_{\lambda k} = \sum_{i \in D_{\lambda}} \sum_{j \in D_k} a_{ij} $$

and by the symmetry of $A$

$$ = \sum_{i \in D_k} \sum_{j \in D_{\lambda}} a_{ij} $$

$$ = A^A_{k \lambda} . $$

Finally, because $A^A$ is symmetric and column diagonally dominant, it is also row diagonally dominant.
It also follows from the properties proved for $A^M$ and $A^A$, that both $(A^M)^{-1}$ and $(A^A)^{-1}$ are non-negative matrices [Varga 1962]. From the positive nature of the inverse it immediately follows that the multiplicative correction scheme produces a positive rebalanced trial solution in keeping with the physical requirements placed upon the calculated neutron flux. Despite the non-negative nature of $(A^A)^{-1}$ a positive solution $\tilde{a}$ in (7.4.6) is not assured because the right hand side $<\mathcal{W}_k, \Delta \Phi_0>$ is not necessarily positive. As a consequence, the current formulation of MINI is not applicable to the case of the additive coarse mesh equations. (For MINI there is no simple restriction on $\gamma$ like (4.2.3), which will ensure a positive solution. Numerical experimentation, however, has indicated a solution is frequently returned despite this.) For a single energy group rebalanced equation, $A^A$ is symmetric and the conjugate gradient routine in POW3D could be applied directly because $A^A$ is also positive definite (Theorem 6.3).

B. Pyramid partitioning and region balance

In the formation of $A^M$ the properties of the original matrix $A$ unfortunately are not necessarily preserved. The $(\ell,k)^{th}$ element of $A^M$

$A^M_{\ell k} = \sum_{i \in D_k} \sum_{j \in D_k} a_{ij} \{ \Phi_k \odot \Phi_0 \}_j$

may be separated into two components
$$= \sum_{j \in D_k \cap D_l} \left\{ \rho \phi_o \right\} \sum_{i \in D_k} a_{ij} + \sum_{j \in D_k \cap D_l} \left\{ \rho \phi_o \right\} \sum_{i \in D_k} a_{ij}.$$
In experiments with practical calculations it was observed frequently (but not necessarily in all instances) that both MINI and GS techniques work. The rates of convergence, however, were often a little disappointing. In POW3D the solution process for this form of rebalance now is restricted to a direct technique.

C. Disjunctive partitioning and Galerkin weighting

Conditions similar to (6.7.1) apply in the case of disjunctive partitioning and Galerkin weighting, provided some additional conditions are placed on the initial estimate \( \Phi_0 \); namely \( \Phi_0 > 0 \) (which is physically sensible). Consequently \( A \Phi_0 = S_0 \geq 0 \), where not all the elements of \( S_0 \) are identically zero. In this case, however, column diagonal dominance of the resulting reduced matrix is replaced with row dominance. The results are established in a similar manner. Firstly non-positive off-diagonal terms;

\[
A_{\ell k}^M = \langle (P_\ell \Phi_0), A(P_k \Phi_0) \rangle
\]

\[
= \sum_{i \in D_\ell} \left\{ P_\ell \Phi_0 \right\} i \sum_{j \in D_k} a_{ij} \left\{ P_k \Phi_0 \right\} j ,
\]

for \( \ell \neq k \) the positive diagonal components of \( A \) are excluded from (6.7.3) and consequently \( A_{\ell k}^M \leq 0 \). Strict inequality \( A_{\ell k}^M < 0 \) follows directly for properly coupled subdomains \( (k, \ell \pm 1; k \pm 1, \ell) \) and the structure of \( A \) is reflected in \( A_{\ell k}^M \) ensuring irreducibility. Row diagonal dominance follows by a summation over any arbitrary row \( \ell \) of
$A^M$ viz:

$$\sum_k A^M_{kk} = \sum_k (P_k \phi_0)^T A(P_k \phi_0)$$

$$= (P_k \phi_0)^T A \phi_0$$

$$= (P_k \phi_0)^T S_0 \geq 0$$

and positive diagonal terms are a further consequence of this.

The somewhat restrictive condition $\phi_0 > 0$ usually is met in practice, because it is one of the physical conditions imposed upon the solution, however $A \phi_0 > 0$ does not eventuate necessarily. If either is not satisfied there is no assurance that $(A^M)^{-1} > 0$, with appropriate consequences for preservation of a positive solution with the multiplicative form of correction. The case of additive correction with disjunctive partitioning and Galerkin weighting is the same as that of disjunctive partitioning and region balance, and was discussed previously.

It is possible to establish a few results, for the particular case in which $A$ is symmetric and positive definite, which guarantee the suitability of relaxation and conjugate gradient methods provided Galerkin weighting is used. The restriction that $A$ itself is positive definite and symmetric is not as difficult to establish as it first appears to be. The condition of symmetry is assured provided
consideration is restricted to a single group equation from the
multigroup system. (It is only when two or more groups contribute
simultaneously to A that symmetry may be lost.) The properties of the
individual group equation matrix A (Section 4.1) are sufficient to
guarantee A itself is positive definite, through use of the following
theorem [Schwarz, Rutishauser and Stiefel 1973].

Theorem 6.3 An irreducible symmetric matrix that is weakly diagonally
dominant and has positive diagonal elements is itself positive
definite.

The validity of the Gauss-Seidel type of methods is guaranteed by
the following theorem [Young 1971] again stated without proof.

Theorem 6.4 For a symmetric matrix A with positive diagonal elements,
the SOR method converges if and only if A is positive definite and 0 <
\omega < 2.

All that is necessary now, is to establish \( \hat{A} \) is symmetric and
positive definite. The condition of symmetry and positive
definiteness also allow the conjugate gradient technique to be applied
[Bennett 1958]. It is further conjectured that MINI is potentially
suitable, provided the right hand side is non-negative. First the
symmetry of \( \hat{A} \) is established.

Theorem 6.5 Provided Galerkin weighting is used, and A is symmetric,
both $A^M$ and $A^A$ are symmetric, irrespective of the type of partitioning used.

**Proof.** In this and subsequent theorems it is sufficient to establish the result for $A^M$ alone, because the result for $A^A$ follows for the particular case of $\Phi_0 = 1$ by Theorem 6.1.

$$A^M_{ik} = \langle w_k, A P_k \Phi_0 \rangle$$

$$= \sum_{i \in D_k} \sum_{j \in D_k} \{P_k \Phi_0 \} i a_{ij} \{P_k \Phi_0 \} j$$

$$= \sum_{j \in D_k} \sum_{i \in D_k} \{P_k \Phi_0 \} j a_{ji} \{P_k \Phi_0 \} i$$

$$= \langle w_k, A^T P_k \Phi_0 \rangle$$

$$= A^M_{kk} .$$

**Theorem 6.6** For $A$ positive definite, then provided Galerkin weighting is used $A^M$ and $A^A$ have positive diagonal elements irrespective of the type of partitioning employed.

**Proof.** $A^M_{kk}$ may be expressed in the form
where \( y = P_k \phi \), and positiveness of \( A_{kk} \) immediately follows because \( A \) is positive definite and \( y \neq 0 \).

**Theorem 6.7** If \( A \) is a real positive definite matrix and Galerkin's method is used, then \( A^M \) and \( A^A \) are real positive definite matrices irrespective of the type of partitioning used.

**Proof.** \( A^M \) and \( A^A \) are obviously real for all forms of partitioning. To demonstrate the positive definite nature of \( A^M \) it is sufficient to show \( \varphi^T A^M \varphi > 0 \) for any \( \varphi \neq 0 \).

\[
\varphi^T A^M \varphi = \sum_i \sum_j \varphi_i A_{ij} \varphi_j
\]

\[
= \sum_i \sum_j \varphi_i (P_i \phi) A(P_j \phi) \varphi_j
\]

\[
+ \sum_i \varphi_i (P_i \phi) A \sum_j (P_j \phi) \varphi_j
\]

\[
= x^T A x ,
\]

where \( x = \sum_j (P_j \phi) \varphi_j \).
The \((P_j \varphi_0)\) are linearly independent and \(\varphi \neq 0\) so \(x \neq 0\). Consequently because \(x^T A x > 0\) by the positive definite nature of \(A\) it follows that \(\varphi^T A^M \varphi > 0\) for \(\neq 0\). The result for \(A^A\) follows immediately as a special case by Theorem 6.1.

6.8 EQUIVALENCE OF ADDITIVE AND MULTIPLICATIVE TECHNIQUES FOR INITIAL FLAT FLUX ESTIMATES

It is now proved (Theorem 6.8) for the special case (of some computational interest), where a flat flux estimate \(\varphi_0\) is considered, that the additive and multiplicative rebalance methods produce identical results.

**Theorem 6.8** If \(\varphi_0\) is an initial constant approximation to the solution of \(A \varphi = s\); and \(\varphi^M\) and \(\varphi^A\) represent improved estimates of \(\varphi_0\) after multiplicative and additive coarse mesh rebalancing respectively, such that

\[
\varphi^M = \varphi_0 \left[ \sum_{k=1}^{M} p_k c_k \right]
\]

and

\[
\varphi^A = \varphi_0 + \left[ \sum_{k=1}^{M} p_k d_k \right],
\]

where \(p_k\) are partitioning matrices (multiplicative rebalancing), and
$P_k$ are the correction vectors (additive rebalancing) related to $P_k$ by $P_i = \text{diag}[P_i]$; then $\tilde{\phi}^M$ and $\tilde{\phi}^A$ are identical when determined by the weighted residual method for the same weighting vectors. $c_k$ and $d_k$ are related by $c_k = \frac{a+d_k}{a}$, where $a = \{\phi_i\}_{i=1}^N$, $(k=1,2,...,M)$.

**Proof.** For any linearly independent set of weighting vectors $w_\ell$, $\\ell=1,2,...,M$; $c$ satisfies the $M \times M$ linear system below, generated by the weighted residual technique

$$
\sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle c_k = \langle w_\ell, s \rangle, \quad (\\ell=1,2,...,M).
$$

Substituting $\frac{a+d_k}{a}$ for the unique solution $c_k$ gives

$$
\sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle + \sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle \frac{d_k}{a} = \langle w_\ell, s \rangle.
$$

Consequently

$$
\sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle d_k = \langle w_\ell, s \rangle - \sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle.
$$

Hence

$$
\sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle d_k = \langle w_\ell, s \rangle - \sum_{k=1}^M \langle w_\ell, AP_k \phi_o \rangle.
$$
which is the system of linear equations arising from the additive form of the weighted residual technique. The required result is then immediate from the uniqueness of solution of (6.8.1).

Corollary. Theorem 6.8 also applies where $\phi^M$ and $\phi^A$ satisfy the rebalanced eigenvalue equation.

The proof of the corollary follows in a similar manner.

Should the estimate $\phi_0$ prove to be the actual flux $\phi$ then application of coarse mesh rebalance procedures does not perturb the estimate. This is proved in the following theorem for the multiplicative form; the additive case is somewhat analogous.

Theorem 6.9. Should the estimate $\phi_0$ be the actual solution $\phi$ of the inhomogeneous equation $A\phi = s$, then application of the multiplicative or additive coarse mesh rebalance procedure does not perturb the solution.

Proof. The multiplicative rebalanced estimate $\phi_1$ is expressed as

$$\phi_1 = \sum_{k=1}^{M} a_k P_k \phi_0$$
where $A_0 \phi_0 = s$, and $\phi_1$ satisfies the $M \times M$ system

$$
\sum_{k=1}^{M} \langle w_k, \Lambda \mathbf{P}_k \phi_0 \rangle a_k = \langle w_k, s \rangle \quad (\ell=1,2,\ldots,M) .
$$

(6.8.2)

It can be shown that $a_j = 1 (j=1,2,\ldots,M)$ is a solution of (6.8.2).

Substituting for all $a_j$, (6.8.2) becomes

$$
\sum_{k=1}^{M} \langle w_k, \Lambda \mathbf{P}_k \phi_0 \rangle = \langle w_k, s \rangle \quad (\ell=1,2,\ldots,M) ,
$$

and using $\sum_{k=1}^{M} \mathbf{P}_k = \mathbf{I}$, this reduces to

$$
\langle w_k, A \phi_0 \rangle = \langle w_k, s \rangle \quad (\ell=1,2,\ldots,M) .
$$

Because the solution to the reduced system is unique,

$$
\phi_1 = \sum_{k=1}^{M} a_k \mathbf{P}_k \phi_0 = \phi_0 .
$$

Corollary 6.9a. Should the estimate $\phi_0$ be a multiple of the actual solution $\phi$, then application of the multiplicative rebalance procedure returns the correct solution $\phi$.

Proof. Suppose $\phi_0$ is a multiple of $\phi$
i.e. $\phi_0 = c \phi$,
then the weighted residual technique for any partitioning and weighting is

$$
\sum_{k=1}^{M} \langle w_k, A P_k \phi_0 \rangle a_k = \langle w_k, s \rangle
= \langle w_k, \frac{1}{C} A \phi_0 \rangle.
$$

(6.8.3)

It is easily verified that $a_k = \frac{1}{c}$, $k=1,2,...,M$ satisfies (6.8.3)

$$
\sum_{k=1}^{M} \langle w_k, A P_k \phi_0 \rangle \frac{1}{C} = \langle w_k, A \sum_{k=1}^{M} P_k \frac{1}{C} \phi_0 \rangle
= \langle w_k, \frac{1}{C} A \phi_0 \rangle,
$$

and as a consequence of the uniqueness of the solution to (6.8.3) the result is established.

A subsequent corollary is given without proof.

**Corollary. 6.9b** Theorem 6.9 also applies in the case of the homogeneous eigenvalue problem.

Upon completion of any coarse mesh rebalance operation, be it additive or multiplicative, no advantage is obtained from a subsequent application of the same procedure. This result of some computational
interest is formalised in the following theorem.

**Theorem 6.10** If $\phi_1$ is the multiplicative or additive coarse mesh rebalanced solution to the estimate $\phi_0$ of $A\phi = s$, then any subsequent rebalanced solution $\phi_2$ obtained from $\phi_1$ with the same partitioning and weighting is identical to $\phi_1$.

**Proof.** The first coarse mesh multiplicative rebalance solution $\phi_1$ may be expressed as

$$\phi_1 = \sum_{k=1}^{M} (P_k \phi_0) a_k^1,$$

and this satisfies the weighted residual system of linear equations

$$\sum_{k=1}^{M} \langle W_k, AP_k \phi_0 \rangle a_k^1 = \langle W_k, s \rangle, \quad (k=1,2,\ldots,M).$$

If a second rebalance operation is performed starting with $\phi_1$ (instead of $\phi_0$), a second rebalanced vector $\phi_2$ is obtained which may be written as

$$\phi_2 = \sum_{k=1}^{M} (P_k \phi_1) a_k^2,$$

where $\phi_2$ satisfies the weighted residual relationship.
\[ \langle \omega_2, A \phi_2 \rangle = \langle \omega_2, s \rangle, \quad (\ell=1,2,\ldots,M) \] \quad (6.8.4)

Substituting for \( \phi_2 \) in (6.8.4) gives a system of linear equations

\[ \sum_{k=1}^{M} \langle \omega_2, A(P_k \phi_1) \rangle a_k^2 = \langle \omega_2, s \rangle, \quad (\ell=1,2,\ldots,M) \] \quad (6.8.5)

Let \( a_k^2 = 1+\varepsilon_k \) (k=1,2,...,M) be the solution to (6.8.5). Then

\[ \sum_{k=1}^{M} \langle \omega_2, A(P_k \phi_1) \rangle + \sum_{k=1}^{M} \langle \omega_2, A(P_k \phi_1) \rangle \varepsilon_k = \langle \omega_2, s \rangle \]

Consequently

\[ \sum_{k=1}^{M} \langle \omega_2, A(P_k \phi_1) \rangle \varepsilon_k = \langle \omega_2, s - A \sum_{k=1}^{M} P_k \phi_1 \rangle \]

\[ = \langle \omega_2, s - A \phi_1 \rangle \]

\[ = 0 \]

Therefore \( \varepsilon_k = 0 \), k=1,2,...,M and it follows that \( \phi_2 = \phi_1 \).

**Corollary.** Theorem 6.9 also applies in the case of the homogeneous eigenvalue problem.

6.9 **ANALYSIS OF THE COARSE MESH REBALANCING METHOD**

A motivation for using coarse mesh rebalance techniques, apart from some vague appeal to obtaining (hopefully) a better 'shaped' flux estimate, is that such a procedure may provide a means of removing low frequency eigenvector components from the error vector \( e = \phi - \phi_0 \).
The ability of standard iterative methods to remove high frequency error components in an efficient manner is discussed in Chapter (7). If coarse mesh techniques can take out the low frequency components quickly, then the rate of convergence of the iterative procedure may be improved significantly. It is difficult to analyse just how well the method succeeds in this aim, because of the non-linear nature of the transformations involved, however, for a somewhat restricted case, it is demonstrated that weighting vectors exist which are capable of completely removing the low frequency eigenvector error components. A similar treatment is given by Nakamura [1977] for the eigenvalue problem.

First the concept of order of an eigenvector is introduced. The low and high order eigenvectors are assumed to be associated with predominantly low and high order geometric frequencies respectively. A theorem supporting the premise that low eigenvector components may be removed from the residual error vector \( r = A\phi_1 - s \) where \( \phi_1 \) is the rebalanced vector is now established, for the particular case in which \( A \) is real symmetric and has distinct eigenvalues. The theorem may be extended to the error vector \( e = \phi - \phi_1 \) as well.

**Theorem 6.11** If low frequency eigenvectors associated with a real symmetric matrix \( A \) are used as weighting vectors for either multiplicative or additive coarse mesh rebalancing of the source problem \( A\phi = s \), then the low frequency error components are eliminated completely from the residual \( r = A\phi - s \).
Proof. The case of multiplicative correction is considered first. The real \(N \times N\) symmetric matrix \(A\) is assumed to have \(N\) distinct eigenvalues \(\lambda_i, i=1,2,\ldots,N\); with eigenvectors \(u_i, i=1,2,\ldots,N\). The \(u_i\) are considered to be ordered from low to high eigenvectors. The eigenvectors belonging to distinct eigenvalues are orthogonal [Faddeeva 1959], and form a linearly independent basis set for the \(N\) dimensional space. The correct solution \(\hat{\phi}\) may be expressed in terms of the \(N\) eigenvectors

\[
\hat{\phi} = \sum_{n=1}^{N} c_n u_n
\]  

(6.9.1)

hence

\[
S = A \hat{\phi} = \sum_{n=1}^{N} c_n \lambda_n u_n
\]

The multiplicative rebalanced vector \(\tilde{\phi}_1\) is expressed as

\[
\tilde{\phi}_1 = \sum_{k=1}^{M} a_k \tilde{f}_k
\]

where \(\tilde{f}_k = p_k \hat{\phi}\). \(f_k\) may also be expressed in terms of the eigenvectors as

\[
\tilde{f}_k = \sum_{n=1}^{N} b_{kn} u_n
\]

so the rebalanced vector \(\tilde{\phi}_1\), becomes
\[ \sum_{k=1}^{M} a_k \sum_{n=1}^{N} b_{kn} u_n \]

\[ = \sum_{n=1}^{N} a_n^1 u_n, \quad (6.9.2) \]

where \( a_n^1 = \sum_{k=1}^{M} a_k b_{kn} \). The rebalanced vector \( \phi_1 \) is obtained by the weighted residual method with weights \( \omega_\xi (\xi = 1, 2, \ldots, M) \) chosen to be the \( M \) low order eigenvectors of \( A \), so that

\[ \langle u_\xi, A \phi_1 - \xi \rangle = 0. \]

With the expansion (6.9.2) this produces

\[ \langle u_\xi, A \sum_{n=1}^{N} a_n^1 u_n \rangle = \langle u_\xi, \xi \rangle. \]

Hence

\[ \sum_{n=1}^{N} \langle u_\xi, a_n \lambda_n u_n \rangle = \sum_{n=1}^{N} \langle u_\xi, c_n \lambda_n u_n \rangle \quad (6.9.3) \]

and from the orthogonality relation of the distinct eigenvectors

\[ a_\xi = c_\xi \quad (\xi = 1, 2, \ldots, K). \quad (6.9.4) \]

Consequently the expansion for the residual is
\[ z = A\phi_1 - \xi \]

\[ = \sum_{n=1}^{N} (a_n^\lambda \gamma_n - c_n^\lambda \gamma_n) \]

\[ = \sum_{n=M+1}^{N} (a_n^\lambda \gamma_n - c_n^\lambda \gamma_n) , \]

by (6.9.4). Thus all the low frequency eigenvectors have been removed from the residual.

The theorem is established next for the case of additive rebalancing. The additive rebalanced vector is

\[ \phi_1 = \phi_0 + \sum_{k=1}^{M} a_k p_k , \]

where the \( p_k \) are given by \( p_k = \text{diag}\{p_k\} \), so by the analogue of the reasoning leading to (6.9.2)

\[ \phi_1 = \phi_0 + \sum_{n=1}^{N} a_n^\lambda \gamma_n . \]

The expression \( \xi - A\phi_0 \) can be expressed in terms of the eigenvectors as

\[ \xi - A\phi_0 = \sum_{n=1}^{N} c_n^\lambda \gamma_n . \]
Now the weighted residual technique with \( u_k, \xi = 1, 2, \ldots, M \) as weighting vectors gives

\[
\langle u_k, A \sum_{n=1}^{N} a_n u_n \rangle = \langle u_k, \xi - A \phi_0 \rangle,
\]

and hence

\[
\langle u_k, \sum_{n=1}^{N} a_n \lambda_n u_n \rangle = \langle u_k, \sum_{n=1}^{N} c_n u_n \rangle,
\]

and from the orthogonality condition the following relation

\[
a_k = \frac{c_k}{\lambda_k} \quad (\xi = 1, 2, \ldots, M). \tag{6.9.5}
\]

It immediately follows that \( r = A \phi_0 - \xi \)

\[
= A \sum_{n=1}^{N} a_n u_n - \xi + A \phi_0
\]

\[
= \sum_{n=1}^{N} a_n \lambda_n n u_n - \sum_{n=1}^{N} c_n u_n,
\]

hence

\[
r = \sum_{n=M+1}^{N} a_n \lambda_n n u_n.
\]
Corollary. Theorem 6.11 also applies to the error vector \( \Phi - \Phi_1 \) for both multiplicative and additive coarse mesh rebalancing.

Proof. The error vector \( e = \Phi - \Phi_1 \) from (6.9.1) and (6.9.2) is

\[
e = \sum_{n=1}^{N} c_n u_n - \sum_{n=1}^{N} a_n u_n,
\]

but by (6.9.4) for the multiplicative case

\[
a_\xi = c_\xi \quad (\xi=1,2,...,M)
\]

\[
e = \sum_{n=M+1}^{N} c_n u_n - \sum_{n=M+1}^{N} a_n u_n.
\]

The error vector \( e \) for additive rebalancing is

\[
e = \Phi - \Phi_0 - \sum_{n=1}^{N} a_n u_n
\]

but

\[
s - A\Phi_0 = A(\Phi - \Phi_0) = \sum_{n=1}^{N} c_n u_n
\]
\[ \Phi - \Phi_0 = \sum_{n=1}^{N} \frac{c_n}{\lambda_n} u_n, \]

consequently

\[ e = \sum_{n=1}^{N} \frac{c_n}{\lambda_n} u_n - \sum_{n=1}^{N} a_n u_n \]

and by (6.9.5)

\[ = \sum_{n=M+1}^{N} \frac{c_n}{\lambda_n} u_n - \sum_{n=M+1}^{N} a_n u_n. \]

It is immediately obvious from (6.9.3) that the theorem is a particular case of a more general result, namely, that the choice of weighting vectors used in the weighted residual section completely determines the order of the eigenvectors removed from the rebalanced vector. For example, if high order eigenvectors were used, then the high order terms would be removed from the residual. In view of the findings reported later in Chapter 7, this would appear a fairly pointless exercise.

The results for weighting vectors corresponding to low frequency eigenvectors is extended to the case in which the weighting vector is made up as a combination of all low frequency eigenvector components, rather than being unique eigenvectors.

Corollary. If the weighting vectors are linear combinations of the
low frequency eigenvectors, then the residual and error vectors after rebalancing do not include any of the low frequency components.

**Proof.** The weighting vectors are expressed as a linear combination of the linearly independent low frequency eigenvectors by

\[ \mathbf{w}_k = \sum_{k=1}^{M} e_{\xi k} \mathbf{u}_k \quad (\xi=1,2,\ldots,M) \]

and the M\times M matrix \( \mathbb{E} \) (with elements \( e_{\xi k} \)) is consequently non-singular. The residual relationship following the approach of Theorem 6.11 is

\[ \left\langle \mathbf{w}_k, \sum_{n=1}^{N} a_n \mathbf{v}_n \right\rangle = \left\langle \mathbf{w}_k, \sum_{n=1}^{N} c_n \mathbf{v}_n \right\rangle. \]

Expansion of the weighting vectors gives

\[ \left\langle \sum_{k=1}^{M} e_{\xi k} \mathbf{u}_k, \sum_{n=1}^{N} a_n \mathbf{v}_n \right\rangle = \left\langle \sum_{k=1}^{M} e_{\xi k} \mathbf{u}_k, \sum_{n=1}^{N} c_n \mathbf{v}_n \right\rangle \]

and orthogonality leads to

\[ \sum_{n=1}^{N} e_{\xi k} (a_k - c_k) \lambda_k = 0 \quad (\xi=1,2,\ldots,M). \]

By the non-singularity of \( \mathbb{E} \)
(a_k - c_k)\lambda_k = 0 \quad (k=1,2,\ldots,M),

and consequently for distinct non-zero eigenvalues \lambda_k

a_k = c_k.

Consequently

\[ r = \sum_{n=M+1}^{N} (a_n \lambda_n u_n - c_n \lambda_n u_n). \]

The result for \( e = \phi - \phi_1 \) follows in a similar way, as does the proof for additive rebalance techniques.

Unfortunately the eigenvectors of the matrix for all but the simplest instances are unknown, and consequently are unavailable for the CMR process. The weighting vectors actually selected for the four schemes come directly from the coarse mesh itself. Because the coarse mesh is capable of approximating low frequency geometric terms only, it is hoped that the weighting vectors will be composed predominantly of these. Consequently it is expected that CMR will remove them from the estimated solution.
7.1 INTRODUCTION

It was conjectured in Section 6.9 that CMR suppresses the low frequency error components of the trial vector solution. A simple theorem established the result for specific weighting vectors, but these unfortunately cannot be found in practice. The potential value of CMR used in conjunction with various iterative schemes will be investigated in this chapter.

Brandt [1977] notes that when the usual iterative procedures are applied, there is a rapid initial decrease in error, but after several iterations, the reduction is no longer sustained.

Iterative methods are known to decrease error in the estimates of the solution, asymptotically in proportion to the absolute magnitude of each eigenvalue, when the error is expanded in terms of the eigenvectors of the iteration matrix. On this basis, it is realistic to anticipate a rapid initial decrease in error associated with the
removal of the lower eigenvalue components of error, and that the other components will persist for some time. This expansion has limited use, however, because the eigenvectors and eigenvalues can be specified directly for only the simplest form of the differential operator.

Brandt considers the problem of convergence from another viewpoint, which is pursued throughout this Chapter. In this approach, error in the estimate of the solution is considered in terms of the various Fourier frequencies. The high frequency components of error are representable only on the fine grid system, where neighbouring points are coupled closely. It is postulated, that certain iterative techniques easily remove the high frequency components of error, but have considerable difficulty with the lower terms. Low frequency components are represented on a coarse mesh. It is anticipated that the solution obtained over this mesh will have a much smaller low frequency error component than the original estimate for the fine grid. It is hoped that when the coarse mesh solution is used to modify the fine mesh estimate, the low frequency error components that contribute to the fine mesh error vector are reduced. It is in the application of the coarse grid solution to the finer mesh, that CMR and Brandt's multigrid approach differ. In essence the multigrid technique uses the coarse mesh solution in an interpolative manner to obtain a fine grid estimate, whereas CMR uses it to adjust the previous fine grid estimate in either a multiplicative or additive process.
Knowledge of the rapid removal of high frequency terms motivated Brandt in his formulation of the multigrid algorithms. This knowledge is supported by what Brandt [1975] and Nicolaides [1975] describe as a 'local mode' error analysis for some relaxation approaches. In this analysis, a smoothing factor \( \mu(\theta) \) is determined. This is a measure of the decay induced in each Fourier component of error by an iterative step. The analysis is performed at a single grid point, far removed from the boundary. In carrying out the point analysis, it is important to realise that the matrix iterative process itself is not being analysed. Instead it is the change to the error in the solution approximation, (examined in terms of the Fourier components), as the iterative procedure passes through a particular grid point (or collection of grid points for a block process) that is studied. Consideration of a Brandt approach is justified to the extent, that in some way, it reflects the iterative process.

A similar analysis is performed here, based on the point version of MINI, which later is extended (by an additional assumption in keeping with Brandt [1977]), to accommodate block versions. Smoothing factors for other iterative schemes are quoted and compared.

The approach is somewhat artificial, in that it distorts the structure of the actual matrix involved by ignoring the boundary conditions (and consequently its eigenvector structure). It also involves some rather gross assumptions for the general case. It does present, however, a plausible indication of the effectiveness of
coarse mesh rebalance techniques for accelerating convergence. Although the local mode analysis technique comes in for some mathematical criticism of its general applicability, the insight it offers and its agreement with other analyses of error is sufficient justification to report on it here.

A more global approach involving Fast Fourier transforms is included, this does not suffer from some of the limitations in the 'local' approach. The results from this treatment support the 'local' mode findings.

7.2 'LOCAL MODE' FOURIER ANALYSIS

Consider a general equation of the form

\[-a \frac{\partial^2 u}{\partial x^2} - b \frac{\partial^2 u}{\partial y^2} = F(x,y),\]

subject to appropriate Dirichlet or Neumann boundary conditions. This equation can be approximated at any mesh point \((\alpha, \beta)\) situated away from the boundary by the following finite difference representation

\[a(-U_{\alpha+1\beta} + 2U_{\alpha\beta} - U_{\alpha-1\beta}) + b(-U_{\alpha\beta+1} + 2U_{\alpha\beta} - U_{\alpha\beta-1}) = h^2 f_{\alpha\beta}, \quad (7.2.1)\]

where an evenly spaced mesh is assumed about \((\alpha, \beta)\) and \(U\) represents the exact finite difference solution to (7.2.1).
Let \( \tilde{u} \) and \( \bar{u} \) represent the \((n-1)\) and \((n)\) iterate approximations respectively for \( U \) during a MINI iteration, where the solution procedure already has passed through the points \((\alpha, \beta-1)\) and \((\alpha-1, \beta)\). Applying MINI to equation (7.2.1) (and for convenience dropping the iteration dependence of \( \gamma \)) gives

\[
\begin{align*}
\tilde{a}(-u_{\alpha+1} + \gamma_{\alpha+1} (\tilde{u}_{\alpha\beta} - u_{\alpha\beta}) + 2\tilde{u}_{\alpha\beta} - \tilde{u}_{\alpha-1\beta}) \\
\bar{b}(-u_{\alpha\beta+1} + \gamma_{\alpha\beta+1} (\tilde{u}_{\alpha\beta} - u_{\alpha\beta}) + 2\tilde{u}_{\alpha\beta} - \tilde{u}_{\alpha\beta-1}) = h^2 \bar{F}_{\alpha\beta}. \quad (7.2.2)
\end{align*}
\]

The finite difference approximation (7.2.1) may be altered by the inclusion of two zero terms \( \gamma_{\alpha+1\beta}(U_{\alpha\beta} - U_{\alpha\beta}) \) and \( \gamma_{\alpha\beta+1}(U_{\alpha\beta} - U_{\alpha\beta}) \) respectively to

\[
\begin{align*}
\tilde{a}(-u_{\alpha+1\beta} - \gamma_{\alpha+1\beta} (U_{\alpha\beta} - U_{\alpha\beta}) + 2U_{\alpha\beta} - u_{\alpha-1\beta}) \\
\bar{b}(-u_{\alpha\beta+1\beta} - \gamma_{\alpha\beta+1\beta} (U_{\alpha\beta} - U_{\alpha\beta}) + 2U_{\alpha\beta} - u_{\alpha\beta-1}) = h^2 \bar{F}_{\alpha\beta}. \quad (7.2.3)
\end{align*}
\]

If the error between the solution \( \tilde{U} \) and the two subsequent iterates \( \tilde{u} \) and \( \bar{u} \) is written as

\[
y = \tilde{U} - \bar{u}
\]

and
\[ \bar{V} = \bar{U} - \bar{v} , \]

then subtraction of (7.2.2) from (7.2.3) gives

\[
a(-\nu_{\alpha+1\beta} - \gamma_{\alpha+1\beta} (\bar{v}_{\alpha\beta} - \nu_{\alpha\beta}) + 2\bar{v}_{\alpha\beta} - \nu_{\alpha-1\beta}) \\
+ b(-\nu_{\alpha\beta+1} - \gamma_{\alpha\beta+1} (\bar{v}_{\alpha\beta} - \nu_{\alpha\beta}) + 2\bar{v}_{\alpha\beta} - \nu_{\alpha\beta-1}) = 0 ,
\]

(7.2.4)

as an expression of error about the point \((\alpha, \beta)\).

The \(\chi\)'s are assumed to be constant in any direction for the following analysis (and hence positionally independent). Consequently \(\chi_{\alpha+1, \beta}\) and \(\chi_{\alpha, \beta+1}\) are written as \(\chi_x\) and \(\chi_y\) respectively, to indicate the dependence on direction alone. In reality, positional independence does not hold; so the analysis is necessarily restrictive. Now in a Fourier study, elements of the error vectors at the point \((\alpha, \beta)\) on subsequent iterations are

\[
\nu_{\alpha\beta} = \sum_{\theta_1} \sum_{\theta_2} A_{\theta_1\theta_2} e^{i(\alpha\theta_1 + \beta\theta_2)}
\]

and

\[
\bar{\nu}_{\alpha\beta} = \sum_{\theta_1} \sum_{\theta_2} \bar{A}_{\theta_1\theta_2} e^{i(\alpha\theta_1 + \beta\theta_2)}
\]

respectively.
Substitution into the linear relation (7.2.4) and use of the linear independence of the Fourier vectors to separate coefficients yields

\[ A_{\theta_1, \theta_2}(-a_{x}+2a-b_{y}+2b-ae^{-i\theta_1}+be^{-i\theta_2})+A_{\theta_1, \theta_2}(-a_{x}+a_{y}+be^{i\theta_1}+b_{y}e^{i\theta_2}) = 0 \]

for all \( \theta_1 \) and \( \theta_2 \). To measure the effect of damping on a particular Fourier term after one MINI iteration, a smoothing coefficient (or damping factor) \( \mu(\theta_1, \theta_2) \) is defined as the ratio of the absolute magnitude of the Fourier coefficients after an iteration to those before i.e.

\[ \mu(\theta_1, \theta_2) = \frac{|A_{\theta_1, \theta_2}|}{|A_{\theta_1, \theta_2}|} \]

For the point version of MINI

\[ \mu_{\text{MINI}}(\theta_1, \theta_2) = \frac{|ae^{-i\theta_1}+be^{-i\theta_2}-a_{x}+b_{y}|}{|2a+2b-a_{x}-b_{y}+ae^{-i\theta_1}-be^{-i\theta_2}|} \quad (7.2.5) \]

It must be stressed that the 'local mode' analysis above is a study of the five point linear relation (7.2.4) and not the full MINI process of Chapter 4. This form of analysis is capable only of giving some idea about what happens to the error components at mesh points away from the boundary. For most reactor systems the grids are relatively fine, so the majority of points fit into this category.
Any attempt to extend the explanation to the full process encounters problems that are identified in the following treatment [Doherty 1981].

Consider the linear system

\[ A \tilde{u} = \tilde{s} \]

with the following splitting of \( A \)

\[ A = M - N , \]

which allows the linear system to be expressed

\[ M \tilde{u} = N \tilde{u} + \tilde{s} . \]  \hspace{1cm} (7.2.6)

This then defines the iterative scheme

\[ \tilde{u}^{(n)} = M^{-1} N \tilde{u}^{(n-1)} + M^{-1} \tilde{s} . \]  \hspace{1cm} (7.2.7)

Subtracting (7.2.6) from (7.2.7) yields

\[ \tilde{u}^{(n)} - \tilde{u} = M^{-1} N (\tilde{u}^{(n-1)} - \tilde{u}) . \]  \hspace{1cm} (7.2.8)

The error vector at step \( n \) may be expanded in terms of the \( N \) eigenvectors \( w_i \) of \( M^{-1} N \) as
\[(u(n)-u) = \sum_i a_i^{(n)}w_i\]

or

\[(u(n)-u) = W \tilde{a}(n),\]

where \(W\) is a matrix whose columns are the eigenvectors \(w_i\) and the components of \(\tilde{a}\) are the coefficients of the error expansion. Provided the \(w_i\) are orthogonal eigenvectors, as for example in the point Jacobi scheme,

\[a^{(n-1)} = W^T(u^{(n-1)}-u)\]

\[a^{(n)} = W^T(u^{(n)}-u)\]

\[= W^T M^{-1}N(u^{(n-1)}-u)\]

\[= W^T M^{-1}NWa^{(n-1)}\]

\[= T_{a}^{(n-1)}. \quad (7.2.9)\]

If the eigenvectors of the iterative scheme are orthogonal, \(T\) is a diagonal matrix whose elements are expressed in terms of the eigenvalues of the matrix. Under these conditions it seems natural to analyse the iterative technique in terms of the coefficients of the eigenvector expansion of error. For the simple case of \(a\) and \(b\)
constant in (7.2.1), the orthogonal eigenvectors of the Jacobi scheme are the Fourier terms, and the local mode analysis is rigorously justified. For more complex elliptical systems, the analysis no longer holds, nor can one be sure it ever applies for MINI, due to the uncertainty about the eigenvectors of the iterative matrix. Consequently the separation of error components as performed in the point analysis (7.2.5), cannot be justified by analytical analysis of the iterative scheme, although there is ample empirical evidence for its success. The approach is pursued for insights it provides into the various iterative schemes evaluated.

The smoothing factor for SOR may be found in a similar way to which (7.2.5) was derived for MINI. At any mesh point \((\alpha, \beta)\) the new value for SOR, \(\hat{u}_{\alpha\beta}\), satisfies

\[
\hat{u}_{\alpha\beta} = \omega \bar{u}_{\alpha\beta} + (1-\omega)u_{\alpha\beta},
\]

(7.2.10)

where \(u_{\alpha\beta}\) denotes the value from the previous iteration and \(\bar{u}_{\alpha\beta}\) denotes the value obtained from solving the five point difference relationship involving the quantities \(\hat{u}_{\alpha-1\beta}\), \(\hat{u}_{\alpha\beta-1}\), \(\bar{u}_{\alpha\beta}\), \(u_{\alpha+1\beta}\) and \(u_{\alpha\beta+1}\). Equation (7.2.10) is transformed to yield

\[
\bar{u}_{\alpha\beta} = \left(\hat{u}_{\alpha\beta}-(1-\omega)u_{\alpha\beta}\right)/\omega
\]

and this is substituted in the SOR equivalent of (7.2.1) to give the five point relationship
Subtracting (7.2.11) from (7.2.1), writing \( y = U - u \) and \( \hat{v} = U - \hat{u} \) leads to

\[
a(-v_{\alpha+1\beta} + \frac{2\hat{v}}{\omega} u_{\alpha\beta} - \frac{2(1-\omega)}{\omega} u_{\alpha\beta} - \hat{v}_{\alpha-1\beta}) + \]
\[
+ b(-u_{\alpha\beta+1} + \frac{2\hat{u}}{\omega} u_{\alpha\beta} - \frac{2(1-\omega)}{\omega} u_{\alpha\beta} - \hat{u}_{\alpha\beta-1}) = F_{\alpha\beta} .
\]  

(7.2.12)

Again employing linear independence of the Fourier expansion in the relationship (7.2.12), the SOR smoothing factor

\[
\mu_{SOR}(\theta_1, \theta_2) = \frac{|ae^{i\theta_1} + be^{i\theta_2} + \frac{2a(1-\omega)}{\omega} + \frac{2b(1-\omega)}{\omega}|}{|2a - 2b - ae^{i\theta_1} + be^{i\theta_2}|} .
\]  

(7.2.13)

is obtained. Consideration of the point connection between MINI and SOR (Section 4.12) demonstrates the necessary degree of equivalence between (7.2.5) and (7.2.13). The GS smoothing factor follows either from (7.2.5) \( \gamma = 0 \), or (7.2.13) \( \omega = 1 \), and is

\[
\mu_{GS}(\theta_1, \theta_2) = \frac{|ae^{i\theta_1} + be^{i\theta_2}|}{|2a - 2b - ae^{i\theta_1} + be^{i\theta_2}|} .
\]
As indicated in Chapter 5, point methods are seldom used to solve the multi-dimensional fine mesh reactor calculations, because the rate of convergence with the block or line versions is considerably superior. For two-dimensional problems, a line version of MINI or SOR solves along each mesh line in turn. This requires the solution of a tridiagonal system of linear equations which is efficiently achieved through the Thomas [1949] algorithm. In three-dimensions, the blocking is achieved in terms of \((x,y)\) planes, which in turn are solved by the line MINI(LMINI), SLOR or ICCG techniques.

To obtain the smoothing factor for a three-dimensional problem solved with a block approach, consider the general form

\[
-a \frac{\partial^2 u}{\partial x^2} - b \frac{\partial^2 u}{\partial y^2} - c \frac{\partial^2 u}{\partial z^2} = F(x,y,z) \quad .
\] (7.2.14)

For any point \((\alpha,\beta,\delta)\), only one \(\gamma_{(z)}^{(\alpha,\beta,\delta+1)}\) is required for a block MINI(BMINI) process based on the \((x,y)\) plane. This \(\gamma\) relates the point in the current plane to the corresponding point \((\alpha,\beta,\delta+1)\) in the plane yet to be solved for. Consequently the three-dimensional block representation equivalent to (7.2.2) is

\[
a(-u_{\alpha+1\beta\delta} + 2u_{\alpha\beta\delta} - u_{\alpha-1\beta\delta}) + b(-u_{\alpha\beta+1\delta} + 2u_{\alpha\beta\delta} - u_{\alpha\beta-1\delta}) + c(-u_{\alpha\beta\delta+1} - \gamma_{\alpha\beta\delta+1}(u_{\alpha\beta\delta} - u_{\alpha\beta\delta}) + 2u_{\alpha\beta\delta} - u_{\alpha\beta\delta-1})
\]
where the solution at the points \((a+1, b, \delta)\) and \((a-1, b, \delta)\) has been obtained by a further process, whose details are assumed unimportant and are not reflected in (7.2.15). This leads to the following expression for the smoothing factor

\[ u_{\text{BMINI}}(\theta_1, \theta_2, \theta_3) = \frac{|e^{i\theta_3} - c\gamma_z|}{|2a+2b+2c-c\gamma_z - ae - i\theta_1 - i\theta_2 - i\theta_3|} \cdot \quad (7.2.16) \]

The equivalent expression for block SOR(SBOR) in three-dimensions is

\[ u_{\text{SBOR}}(\theta_1, \theta_2, \theta_3) = \frac{|a(1-w)[e^{i\theta_1} - 2 + e^{-i\theta_1}] + b(1-w)[e^{i\theta_2} - 2 + e^{-i\theta_2}] - ce^{i\theta_3} - 2c(1-w)|}{|a[1-w(e^{i\theta_1} - 2 + e^{-i\theta_1})] + b[1-w(e^{i\theta_2} - 2 + e^{-i\theta_2})] - 2c - ce^{i\theta_3}|}. \quad (7.2.17) \]

The two smoothing factors for the line versions of MINI(LMINI) and SOR(SLOR) appropriate to a two-dimensional operator, may be obtained similarly, and are given by

\[ u_{LMINI}(\theta_1, \theta_2) = \frac{|be^{i\theta_2} - b\gamma_y|}{|2a+2b-b\gamma_y - ae - i\theta_1 - i\theta_2|}. \quad (7.2.18) \]
and

\[ \mu_{\text{SLOR}}(\theta_1, \theta_2) = \frac{\left| a \frac{1-\omega}{\omega} \left[ e^{i\theta_1 - e^{i\theta_1}} - b e^{i\theta_2} - 2b \frac{1-\omega}{\omega} \right] \right|}{\left| a \frac{1-\omega}{\omega} \left[ e^{-i\theta_1} - e^{i\theta_1} \right] - b \frac{2b}{\omega} e^{-i\theta_2} \right|} \]  \hspace{1cm} (7.2.19)

respectively.

Inspection of (7.2.5), (7.2.13), (7.2.16), (7.2.17), (7.2.18) and (7.2.19) reveals in general, that while \( \theta_1 \) and \( \theta_2 \) increase for most \( a, b \) and \( c \), \( \mu(\theta) \) decreases and is less than 1. (No such strong smoothing is indicated for the six schemes when \( \theta \) is small, in fact as \( \theta \to 0, \mu(\theta) \to 1 \).

There are some minor violations of this phenomenon of smoothing, such as occur with point MINI, where \( \mu(\theta) \) is also dependent upon \( \gamma \). For the case of any \( a, b \), and \( \theta \) for \( \gamma_x = \gamma_y = 1 \)

\[ \mu_{\text{MINI}}(\theta_1, \theta_2) = \frac{\left| a e^{i\theta_1} + b e^{i\theta_2} - a - b \right|}{\left| a e^{-i\theta_1} + b e^{-i\theta_2} - a - b \right|} \cdot \]

The result is no reduction for any of the error component frequencies. Fortunately with MINI, although \( \gamma = 1 \) is permitted, it is a value that doesn't arise frequently in practice, and because it is recalculated after every iteration the probability of its persistence is even lower. (In fact for the non-symmetric point MINI example of Section 4.10 an asymptotic \( \gamma = 1 \) was obtained but convergence was relatively slow for MINI. Presumably the boundary \( \gamma \)'s \#1 were sufficient to
enable convergence to take place.) The same behaviour is observed for SOR, SLOR and SBOR as $\omega \to 2$. Brandt [1977] notes the Gauss-Seidel case for $a < b$, where $\mu(\frac{1}{2}, \theta) \to 1$ as $\omega \to 0$. An appropriately oriented line version, however, overcomes this shortcoming of the point iterative approach, as consideration of (7.2.19) reveals.

Examination of (7.2.16) and a three-dimensional analogue of (7.2.5) suggests that the block form of MINI is more smoothing than point MINI for the three-dimensional problem. A similar examination of (7.2.18) and (7.2.5) supports the same suggestion for line MINI over point MINI for the two-dimensional problem. When MINI was introduced in the code, it was anticipated that the block forms would have a higher rate of convergence. In the absence of any formal theory on convergence, this prediction based on local mode analysis is nice to have. The actual difference in smoothing for specific settings of $(\theta_1 = \theta_2)$ is shown in Figure 7.2 for MINI and LMINI. It is well known that blockings of the relaxation methods are more efficient, however, it is not immediately obvious (because of algebraic difficulties) that smoothing with the relaxation approaches is greater with blocking for all $\omega$ and $\theta$. Specific examples shown in Figure 7.4 suggest it is so for SOR and SLOR.

Through an examination of MINI (7.2.5), the local mode approach reveals that no smoothing occurs for $\gamma > 1$. This supports the upper restriction placed on $\gamma$ for MINI, as formulated in Chapter 4 and found to be desirable in practice. The theoretical upper bound on $\omega$ for SOR
is suggested similarly by the smoothing factor (7.2.13).

In Figure 7.1 the smoothing factor is plotted as a function of $\gamma$ and $\omega$ for the highest frequency component of error for point MINI and SOR methods. The results are shown for the operator $-\nabla^2$ on a 16x16 finite difference grid over a square region of side length $\pi$. Both point and line methods are displayed, and the expected advantage of the line approach is realised. The slight superiority of point MINI over the relaxation method is also demonstrated for the high frequency component, except at the ends of the parameter range, while for the line version the superiority of MINI appears more marked. The nature of the smoothing for point MINI and SOR confirms that (consistent with the assumption of a single $\gamma$ ) for any relaxation factor $\omega$ there exists a $\gamma$ which makes the two methods equivalent as predicted by the point theory (Section 4.2).

It is not possible with LMINI and SLOR, however, to identify directly an $\omega$ and a $\gamma$ in the allowable range which make the two methods equivalent. For a point analysis of the increment in the solution along a line, the SLOR and line Gauss-Seidel(LGS) techniques are related by

$$\delta_{\text{SLOR}} = \delta_{\text{LGS}},$$

(7.2.20)

where $\delta$ denotes the changes in the solution vector along the current line (or block). Let the tridiagonal matrix (associated with a
Smoothing factor as a function of \( \omega \) and \( \gamma \) for the high frequency error component \((n=\frac{14\pi}{15})\) of the operator \(-\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\) on a square region.

Figure 7.1
Smoothing factor as a function of $\gamma$ for the $\theta_1 = \frac{\pi}{15}$, $\theta_2 = \frac{7\pi}{15}$, $\theta_{14} = \frac{14\pi}{15}$ frequency error components of the operator $\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)$ on a square region of side $w$. Smoothing factors for the optimal relaxation methods are superimposed.

Figure 7.2
Smoothing factors for the two-dimensional operator with MINI and LMINI. The smoothing factor for the equivalent three-dimensional operator is included for BMINI. \( \theta_1 = \frac{\pi}{15}, \theta_5 = \frac{5\pi}{15}, \theta_7 = \frac{7\pi}{15} \) and \( \theta_{14} = \frac{14\pi}{15} \) components of error frequencies are displayed.

Figure 7.3
7.20

Smoothing factors for the two-dimensional operator with SOR and SLOR.

The smoothing factor for the equivalent three-dimensional operator is included for SBOR. \( \theta_1 = \frac{\pi}{15}, \theta_5 = \frac{5\pi}{15}, \theta_7 = \frac{7\pi}{15} \) and \( \theta_{14} = \frac{14\pi}{15} \) components of error frequencies are displayed.

Figure 7.4
particular line in the LGS process) arising at the intermediate stage be written

\[ L + D + U, \]

where the matrix \( D \) represents the diagonal term, and the matrices \( L \) and \( U \) are the lower and upper diagonal components respectively after MINI is applied between lines. The following identification between LMINI and LGS may be made

\[ (L + D + D)_{LMINI} = (L + D)_{LGS}, \quad (7.2.21) \]

where \( D \) is the additional diagonal term (or function of \( y \) i.e. \( d_{ij} = \sum_{j>i} a_{ij} y_{ij} \) due to the MINI connection to the forward line of points. Consequently, the connection between LMINI increments and LGS increments is

\[ \delta_{LMINI} = (I + (L + D)^{-1}D)^{-1}\delta_{LGS}. \quad (7.2.22) \]

It is clear from (7.2.22) that in general, each element of \( \delta_{LMINI} \) is no longer connected to a single element of \( \delta_{LGS} \) by a multiplicative constant. It is no longer possible to relate LMINI with SLOR by simple relationships such as (4.2.6) and (4.2.7) (even for a single line). If \([\omega]\) is taken to be a diagonal matrix whose non-zero terms are the over-relaxation parameters \( \omega \), consideration of equations (7.2.20) and (7.2.21) leads to a formal equivalence between a variation of LMINI and SLOR only if
\[ \tilde{D} = (L + D) ([\omega]^{-1} - I) \quad \] (7.2.23)

The matrix \( \tilde{D} \) arising in (7.2.23) is no longer diagonal as indicated in the following 3x3 example for a typical \((L+O+U)\) configuration. For

\[
\begin{pmatrix}
4 & -1 & 0 \\
-1 & 4 & -1 \\
0 & -1 & 4 \\
\end{pmatrix}
\]

then

\[
\tilde{D} = \begin{pmatrix}
4\left(\frac{1}{\omega} - 1\right) & 0 & 0 \\
-(\frac{1}{\omega} - 1) & 4\left(\frac{1}{\omega} - 1\right) & 0 \\
0 & -(\frac{1}{\omega} - 1) & 4\left(\frac{1}{\omega} - 1\right) \\
\end{pmatrix}
\] (7.2.24)

Because \( \tilde{D} \) is no longer diagonal, there is no real equivalence between LMINI and SLOR. The \( \tilde{D} \) matrix (7.2.24), however, suggests the relationship (7.2.23) may be satisfied only if LMINI is reformulated to include a more complex coupling of points in the forward line to the current point.
Returning to the results of 'local mode' analysis, the smoothing factor \( \mu(\theta_1, \theta_2) \) (with \( \theta_1 \) and \( \theta_2 \) again chosen equal) is plotted in Figure 7.2 for the same operator equation. This time three different frequency components are shown as functions of \( \gamma \), for point MINI and LMINI. The chosen components represent low \( (\theta_1 = \frac{\pi}{15}) \), middle \( (\theta_2 = \frac{7\pi}{15}) \) and high \( (\theta_{14} = \frac{14\pi}{15}) \) frequency components of the spectrum. The comparison suggests the high frequencies will be removed quickly by the iterative scheme, but the lower components will persist. The smoothing factors for SOR and SLOR (calculated with the optimal \( \omega \) in each case) are superimposed for the same frequency components. There appears to be little difference between point MINI and SOR for the lower frequency, but at higher frequencies the differences are more evident for a larger portion of the range of admissible \( \gamma \)'s. For the line version of MINI this tendency is even more pronounced. For point MINI and SOR there exists a fixed \( \gamma \) and \( \omega \) which make the two methods equivalent, as predicted. The greatest smoothing for the low frequency error of MINI coincides with the corresponding smoothing factor for optimal SOR. From the graph it is observed that for this particular value of \( \gamma \), the other high smoothing components for MINI also coincide with the optimal SOR smoothing factor. For line MINI, a similar coincidence is suggested for low frequency error, but it is not repeated for high frequency error at all, which again is not surprising in view of (7.2.24).
The results certainly show MINI handles the high frequency error terms more efficiently than it does the lower components.

In Figure 7.3 the smoothing factor is shown as a function of $\gamma$ when $\theta = \frac{\pi}{15}, \frac{5\pi}{15}, \frac{7\pi}{15}, \frac{14\pi}{15}$, for the point and line versions of MINI on a two-dimensional 16x16 grid with the previous operator. In addition a smoothing factor for the three-dimensional operator

$$-\left[ \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \right],$$

is included, where the solution process is block MINI (Section 5.6), on a 16x16x16 grid. In Figure 7.4 a similar situation is displayed for the smoothing factor of the successive over-relaxation forms, shown as a function of $\omega$. As detailed in Section 5.6, the solution of the three-dimensional spatial problem with relaxation and MINI is undertaken as a two level process, so the results reported for block MINI(BMINI) and successive block over-relaxation (SBOR) give the smoothing factors of the outer block process (or z direction iterations). From Figure 7.3, it is obvious (for all frequencies displayed), that the block MINI process for the third spatial dimension possesses superior smoothing qualities to those of a simpler line or point form on a two-dimensional problem with a grid of comparable order.
For the relaxation technique, the block form of relaxation on a three-dimensional problem also demonstrates more smoothing than SLOR or SOR on a comparable two-dimensional problem, for a significant portion of the $\omega$ range. As the mesh becomes more tightly coupled and the value of $\omega$ selected increases, the smoothing terms move together for various frequencies.

It also follows with the relaxation technique that for $\omega$ somewhat greater than $\omega_{\text{opt}}$, there appears to be only a small difference in the smoothing factor across the frequency spectrum predicted by local mode analysis. This is in keeping with SOR theory, where all eigenvalues of the iterative process become equal in absolute value for $\omega \geq \omega_{\text{opt}}$.

The maximum smoothing, with local mode analysis for the low frequency components in Figure 7.4 coincides with the value of $\omega$ predicted by theory for SOR, and is not far from it for SLOR. In Table 7.1 the optimal values of $\omega$ given by theory and predicted from the local mode analysis are displayed for various sized grid subdivisions.

A 'local mode' examination of the block form of MINI and the relaxation techniques on three-dimensional problems, confirms that these methods have their work distributed between iterating in the (z) direction and in the block (x,y) planes. The analysis in general suggests that BMINI has a greater ability to smooth in the (z)
### Table 7.1

$\omega_{opt}$ by theory and local mode analysis of lowest frequency component of error

<table>
<thead>
<tr>
<th>Grid size $n \times n$</th>
<th>SOR $\omega_{opt}$</th>
<th>SLOR $\omega_{opt}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\omega_{opt}$ theory</td>
<td>$\omega_{opt}$ local mode</td>
</tr>
<tr>
<td>4</td>
<td>1.23</td>
<td>1.26</td>
</tr>
<tr>
<td>8</td>
<td>1.48</td>
<td>1.49</td>
</tr>
<tr>
<td>16</td>
<td>1.69</td>
<td>1.69</td>
</tr>
<tr>
<td>32</td>
<td>1.82</td>
<td>1.82</td>
</tr>
<tr>
<td>64</td>
<td>1.91</td>
<td>1.91</td>
</tr>
<tr>
<td>128</td>
<td>1.95</td>
<td>1.95</td>
</tr>
</tbody>
</table>
direction than SBOR. The results of Chapter 8 will support this conjecture for some large reactor geometries. Because the block forms (BMINI and SBOR) compared to ICCG have to work harder in converging the (x,y) plane, it is anticipated there might be a trade-off against ICCG in the third-dimension. The results of Chapter 8 also lend some weight to this supposition, particularly for BMINI.

In comparing the two separate MINI processes for the three-dimensional calculations, it is not sufficient to compare the smoothing factors (7.2.16) and (7.2.18), because MINI in the (z) direction perturbs the diagonal terms of the (x,y) plane MINI matrix. The modified smoothing factor for the inner MINI process (based on this method) is

$$\nu^{\text{MOD}}_{\text{LMINI}}(\theta_1, \theta_2) = \frac{|b e^{i\theta_2} - b \chi_y|}{|2a+2b+2c-c \chi_z-b \chi_y-a e^{i\theta_1}-i \theta_1-a e-\chi y|}.$$  

A comparison of $\nu^{\text{MOD}}_{\text{LMINI}}(\theta_1, \theta_2)$ and $\nu_{\text{BMINI}}(\theta_1, \theta_2, \theta_3)$ with $(\chi_z=\chi_y)$ and $(a=b=c)$ for convenience, suggests that the (z) direction MINI is capable of removing higher frequency components faster than the inner plane MINI iterative stage, but that it has more trouble with the lower frequencies.

The modified smoothing factor for the inner relaxation step is obtained similarly; for this situation
The 'local mode' Fourier analysis of the many relaxation techniques, supports the contention that the performance of such schemes is limited by their inability to remove low frequency error components as rapidly as higher terms for suboptimal estimates of \( \omega \). Due to difficulties in obtaining a precise determination of \( \omega \), the performance of relaxation schemes may be enhanced (for low estimates) by a method which removes low order error components.

The Jacobi method is at variance with the above conclusion. It violates the observation that high frequency terms are removed most rapidly. The Jacobi smoothing factor for the two-dimensional problem considered here is

\[
\nu_j(\theta_1, \theta_2) = \left| \frac{ae^{i\theta_1} + ae^{-i\theta_1} + be^{i\theta_2} + be^{-i\theta_2}}{2a + 2b} \right|
\]

For \( \theta \) around the mid-range of frequencies, \( \nu_j(\theta_1, \theta_2) \) is a minimum, and is largest for \( \theta \to 0 \) and \( \theta \to \pi \). This is in keeping with an understanding of the eigenvalues of the Jacobi iterative matrix (for the simple case of \( a=b=1 \)). Nicolaides [1975], however, shows a weighted Jacobi approach is a more satisfactory choice if the technique is to be used as the basis of a multigrid system.
The Fourier evidence for MINI indicates a more general inability to cope with low frequency error components, relative to the higher frequencies.

Because the Fourier analysis is local in concept and does not accommodate boundary conditions, any interpretation of the results should be qualified accordingly.

In spite of the limitations previously mentioned, the local mode analysis suggests that MINI is a very suitable candidate to benefit from coarse mesh rebalancing, because of the greater relative difficulty it has in suppressing the low frequency error components. The same conclusion cannot be drawn for successive over-relaxation techniques when accurate estimates of $\omega_{\text{opt}}$ are available. Experience over the years with relaxation methods for practical reactor configurations, has shown that a considerable reduction in computational effort results when CMR is applied. It may be conjectured, that the benefit arises from either a reduction in low frequency error components when a suboptimal approximation to $\omega_{\text{opt}}$ is used, or from simply the better approximation to the true solution produced by CMR.

Unfortunately it is difficult to extend the 'local mode' approach to handle the conjugate gradient method, or any of its more recent variants. The conjugate gradient method is global in its orientation as the following formulation reveals for the $i^{th}$ iteration.
The coefficients $a_i$ and $e_i$ clearly are dependent upon the solution at all grid points, and the local mode approach is inapplicable.

In an attempt to overcome the shortcomings of the localised study, a recourse is made to Fourier transforms to permit a 'global' study of the frequency approach to convergence.

**7.3 A GLOBAL MODE FOURIER ANALYSIS**

To investigate in a global sense, the effect of the various iterative schemes on high frequency error components, the fast Fourier transform [Brigham 1974] is used to determine the Fourier coefficients of the error vector at various stages of the iterative process. For this work, the subroutine DHARM [IBM 1970] is used in a two-dimensional mode to obtain the inverse series
where \( X(j_1, j_2) \) denotes the error component of the trial vector at the \((j_1, j_2)\)th mesh point.

Two sample problems are analysed by the Fourier expansion. The first problem is

\[-\nabla^2 u(x, y) = f(x, y),\]

where the boundary condition \( u=0 \) on all sides of a square region of side \( \pi \) is applied. The right hand side \( f(x, y) \) is computed from a solution \( u(x, y) = \sin x \sin y \) over an equally spaced five-point finite difference approximation used on a 17x17 grid which includes the boundaries. A trial solution \( u_0(x, y) = u(x, y) + \sum_{k=1}^{15} \sin kx \sin ky \) was selected because it represented a wide selection of error frequencies of the same magnitude. A second problem

\[-\nabla^2 u(x, y) + \sigma(x, y)u(x, y) = f(x, y)\]

is selected subject to the same boundary conditions and the same trial solution. The right hand side \( f(x, y) \) is calculated numerically to conform to the solution \( u(x, y) = \sin x \sin y \) and \( \sigma(x, y) \) is a function (equivalent to a neutron loss) returning a random value in the range
To assist the interpretation of an otherwise massive number of Fourier coefficients in the error expansion

\[ v(j_1, j_2) = \sum_{k_1=0}^{15} \sum_{k_2=0}^{15} A_{k_1 k_2} e^{2\pi i \left( \frac{j_1 k_1}{16} + \frac{j_2 k_2}{16} \right)} \]

for \(0 \leq j_1 \leq 15, 0 \leq j_2 \leq 15\); the Fourier coefficients \(A_{k_1 k_2}\) are grouped and analysed by range. The range \((m_1, m_2)\) is defined to include all terms whose indices \(k_1\) and \(k_2\) satisfy

\[ m_1 \leq k_1 \leq m_2 \quad \text{for} \quad 0 \leq k_2 \leq m_2 \]

and \(m_1 \leq k_2 \leq m_2 \quad \text{for} \quad 0 \leq k_1 \leq m_1\).

The fraction that all the Fourier coefficients in the range \((m_1, m_2)\) form of all the Fourier components after an iterative step is shown as \(FF(m_1, m_2)\)

\[ FF(m_1, m_2) = \frac{\sum_{k_1=0}^{15} \sum_{k_2=0}^{15} |A_{k_1 k_2}|}{\sum_{k_1=0}^{15} \sum_{k_2=0}^{15} |A_{k_1 k_2}|} \]

i.e.\(\) \(FF(m_1, m_2)\)

\[ \left(7.3.1\right) \]

where the unspecified summation is taken to be over the range \((m_1, m_2)\).

The maximum error
\[ E_M = \max \left| u_{ij}^{(n)} - u_{ij}^{(n-1)} \right| \]

and average error

\[ E_A = \frac{1}{15} \sum_{i=0}^{15} \sum_{j=0}^{15} \left| u_{ij}^{(n)} - u_{ij}^{(n-1)} \right| / 256 \]

are reported also. Results are shown in Tables 7.2 and 7.3 after 1, 3, 5 and 7 iterative steps with the four schemes MINI, ICCG, SLOR, GS. For MINI, the \( \gamma \)'s are redetermined continuously by the algorithm described in Section 4.3. On the first MINI pass, a Gauss-Seidel iteration is used to commence the process. For SLOR, the optimal relaxation factor is used from the start. This is done because a numerical estimate of \( \omega_{\text{opt}} \) may not have been obtained by the seventh iteration and by then the GS process would have reduced the error too much for purposes of comparison. Although \( \omega \) may be obtained directly for problem 1, a numerical estimate [Pollard 1973] obtained from separate calculations is used in both problems.

The error in the initial estimate has a wide distribution of error components across the frequency spectrum. Consideration of \( FF(m_1, m_2) \) reveals a significant shift occurs in the proportion of the frequency terms that comprise the new error, after even one iteration with MINI(=GS), GS and ICCG. After three iterations for both problems considered, the proportion of the various frequency terms comprising the error stabilises, the high frequency components being almost insignificant. For problem 1 with SLOR, the distribution of frequency
TABLE 7.2
A FOURIER ANALYSIS OF FOUR ITERATIVE SCHEMES FOR PROBLEM 1

<table>
<thead>
<tr>
<th>Iterative Method</th>
<th>Iteration Count</th>
<th>$E_M$</th>
<th>$E_A$</th>
<th>EM</th>
<th>EA</th>
<th>FF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0,2</td>
<td>3,4</td>
<td>5,6</td>
<td>7,7</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>1</td>
<td>2.83</td>
<td>0.457</td>
<td>0.48</td>
<td>0.23</td>
<td>0.18</td>
</tr>
<tr>
<td>I</td>
<td>3</td>
<td>0.977</td>
<td>0.353</td>
<td>0.76</td>
<td>0.13</td>
<td>0.07</td>
</tr>
<tr>
<td>N</td>
<td>5</td>
<td>0.609</td>
<td>0.248</td>
<td>0.80</td>
<td>0.11</td>
<td>0.06</td>
</tr>
<tr>
<td>I</td>
<td>7</td>
<td>0.397</td>
<td>0.146</td>
<td>0.82</td>
<td>0.10</td>
<td>0.06</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td>1.31</td>
<td>0.413</td>
<td>0.57</td>
<td>0.21</td>
<td>0.13</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>0.597</td>
<td>0.277</td>
<td>0.77</td>
<td>0.14</td>
<td>0.06</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>0.401</td>
<td>0.183</td>
<td>0.79</td>
<td>0.13</td>
<td>0.05</td>
</tr>
<tr>
<td>G</td>
<td>7</td>
<td>0.266</td>
<td>0.119</td>
<td>0.80</td>
<td>0.13</td>
<td>0.05</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>5.02</td>
<td>0.602</td>
<td>0.26</td>
<td>0.24</td>
<td>0.30</td>
</tr>
<tr>
<td>L</td>
<td>3</td>
<td>2.53</td>
<td>0.390</td>
<td>0.26</td>
<td>0.25</td>
<td>0.29</td>
</tr>
<tr>
<td>O</td>
<td>5</td>
<td>1.09</td>
<td>0.183</td>
<td>0.27</td>
<td>0.24</td>
<td>0.29</td>
</tr>
<tr>
<td>R</td>
<td>7</td>
<td>0.436</td>
<td>0.087</td>
<td>0.27</td>
<td>0.23</td>
<td>0.30</td>
</tr>
<tr>
<td>G</td>
<td>1</td>
<td>2.83</td>
<td>0.457</td>
<td>0.48</td>
<td>0.23</td>
<td>0.18</td>
</tr>
<tr>
<td>S</td>
<td>3</td>
<td>1.06</td>
<td>0.375</td>
<td>0.79</td>
<td>0.12</td>
<td>0.06</td>
</tr>
<tr>
<td>S</td>
<td>5</td>
<td>0.770</td>
<td>0.318</td>
<td>0.85</td>
<td>0.09</td>
<td>0.05</td>
</tr>
<tr>
<td>S</td>
<td>7</td>
<td>0.625</td>
<td>0.272</td>
<td>0.85</td>
<td>0.08</td>
<td>0.05</td>
</tr>
</tbody>
</table>
### TABLE 7.3
A FOURIER ANALYSIS OF FOUR ITERATIVE SCHEMES FOR PROBLEM 2

<table>
<thead>
<tr>
<th>Iterative Method</th>
<th>Iteration Count</th>
<th>$E_M$</th>
<th>$E_A$</th>
<th>FF (0,2)</th>
<th>FF (3,4)</th>
<th>FF (5,6)</th>
<th>FF (7,7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>1</td>
<td>2.59</td>
<td>0.320</td>
<td>0.44</td>
<td>0.25</td>
<td>0.20</td>
<td>0.11</td>
</tr>
<tr>
<td>I</td>
<td>3</td>
<td>0.423</td>
<td>0.093</td>
<td>0.69</td>
<td>0.17</td>
<td>0.10</td>
<td>0.04</td>
</tr>
<tr>
<td>N</td>
<td>5</td>
<td>0.090</td>
<td>0.010</td>
<td>0.67</td>
<td>0.18</td>
<td>0.10</td>
<td>0.04</td>
</tr>
<tr>
<td>I</td>
<td>7</td>
<td>0.030</td>
<td>0.003</td>
<td>0.60</td>
<td>0.24</td>
<td>0.11</td>
<td>0.05</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td>0.989</td>
<td>0.222</td>
<td>0.50</td>
<td>0.24</td>
<td>0.15</td>
<td>0.11</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>0.103</td>
<td>0.030</td>
<td>0.55</td>
<td>0.28</td>
<td>0.13</td>
<td>0.04</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>0.013</td>
<td>0.004</td>
<td>0.56</td>
<td>0.28</td>
<td>0.12</td>
<td>0.04</td>
</tr>
<tr>
<td>G</td>
<td>7</td>
<td>0.002</td>
<td>0.0005</td>
<td>0.57</td>
<td>0.28</td>
<td>0.11</td>
<td>0.04</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>3.28</td>
<td>0.299</td>
<td>0.29</td>
<td>0.24</td>
<td>0.28</td>
<td>0.19</td>
</tr>
<tr>
<td>L</td>
<td>3</td>
<td>0.422</td>
<td>0.064</td>
<td>0.36</td>
<td>0.21</td>
<td>0.25</td>
<td>0.18</td>
</tr>
<tr>
<td>O</td>
<td>5</td>
<td>0.070</td>
<td>0.010</td>
<td>0.44</td>
<td>0.20</td>
<td>0.21</td>
<td>0.14</td>
</tr>
<tr>
<td>R</td>
<td>7</td>
<td>0.010</td>
<td>0.002</td>
<td>0.49</td>
<td>0.20</td>
<td>0.18</td>
<td>0.12</td>
</tr>
<tr>
<td>G</td>
<td>1</td>
<td>2.59</td>
<td>0.320</td>
<td>0.44</td>
<td>0.25</td>
<td>0.20</td>
<td>0.11</td>
</tr>
<tr>
<td>S</td>
<td>3</td>
<td>0.533</td>
<td>0.125</td>
<td>0.71</td>
<td>0.16</td>
<td>0.09</td>
<td>0.04</td>
</tr>
<tr>
<td>S</td>
<td>5</td>
<td>0.188</td>
<td>0.051</td>
<td>0.78</td>
<td>0.10</td>
<td>0.08</td>
<td>0.04</td>
</tr>
<tr>
<td>S</td>
<td>7</td>
<td>0.070</td>
<td>0.020</td>
<td>0.76</td>
<td>0.12</td>
<td>0.08</td>
<td>0.04</td>
</tr>
</tbody>
</table>
components remains fairly static, while for problem 2, a small relative decrease in the contribution made by the high frequency error components is observed. This is in keeping with the theoretical understanding and expectations of SLOR. In problem 1, the error components belong to the set of eigenvectors of the associated Jacobi iterative matrix, and these eigenvalues have a reasonably uniform distribution. A selection of $\omega = \omega_{opt}$ ensures that all the eigenvalues of the iteration matrix are transformed to lie on the complex circle of radius $\omega - 1$ (Section 4.1). Even allowing for small errors in the estimate used for $\omega_{opt}$ (which is designed specifically to favour high rather than low estimates), the distribution of Jacobi eigenvalues is such that most, if not all, of the transformed eigenvalues still fall on the circle.

When $\omega_{opt}$ is known exactly, the eigenvalues of the iterative scheme are identical in magnitude. This is confirmed in problem 1, and there would seem to be little point in searching for a scheme to facilitate the removal of low frequency error alone. A Fourier study involving a 10 percent lower estimate of $\omega_{opt}$, however, reveals a difference in the relative rate of error reduction.

In problem 2, a different situation arises, whereby a more selective elimination of frequency components is achieved through the iterative process. For less straightforward iterative matrices than those of problem 1, a clustering of eigenvalues is possible and errors in the precise determination of $\omega_{opt}$ may affect the number of
eigenvalues which are transformed to lie on the circle. This in turn will influence relative rates of decay of the Fourier terms. Under these circumstances, consideration of coarse mesh rebalance techniques may be worthwhile. Reactor physicists have traditionally used and benefited from such acceleration schemes in conjunction with SLOR.

7.4 COARSE MESH REBALANCING

The four coarse mesh rebalance schemes employed in POW3D, and described in Chapter 6 are subjected to the same global Fourier analysis previously applied to the iterative schemes. The results for two test problems are reported in Tables 7.4 and 7.5. The first problem involves the two-dimensional equation

\[ -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f(x,y), \] (7.4.1)

subject to the boundary condition \( u=0 \) on all sides of a square region of side \( \pi \). For the first problem, the solution \( u^1(x,y) = (\pi-x)(\pi-y)xy \) is selected and for the second \( u^2(x,y) = \sin x \sin y \). In each case the right hand side of (7.4.1) is calculated numerically on a 17x17 grid which includes the boundaries.

For the first example, the trial solution
<table>
<thead>
<tr>
<th>Method of CMR</th>
<th>$E_M$</th>
<th>$E_A$</th>
<th>FF</th>
<th>FR</th>
<th>$%$ of Terms Reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disjunctive Partitioning</td>
<td>0.13</td>
<td>0.04</td>
<td>0.81</td>
<td>0.09</td>
<td>0.07</td>
</tr>
<tr>
<td>Multiplicative Partitioning</td>
<td>0.06</td>
<td>0.03</td>
<td>0.77</td>
<td>0.13</td>
<td>0.06</td>
</tr>
<tr>
<td>Additive Pyramid Partitioning</td>
<td>0.67</td>
<td>0.30</td>
<td>0.79</td>
<td>0.11</td>
<td>0.05</td>
</tr>
<tr>
<td>Additive Pyramid Galerkin Weighing</td>
<td>0.24</td>
<td>0.04</td>
<td>0.67</td>
<td>0.17</td>
<td>0.10</td>
</tr>
</tbody>
</table>

$u = (\pi-x)(m-y)x$ 
$u_0 = u + \sin(x) \sin(y)$


<table>
<thead>
<tr>
<th>Method of CMR</th>
<th>( E_M )</th>
<th>( E_A )</th>
<th>FF</th>
<th>FR</th>
<th>% of terms reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disjunctive Partitioning</td>
<td>0.13</td>
<td>0.04</td>
<td>0.81 0.09 0.07 0.03</td>
<td>0.02 0.02 0.02 0.02</td>
<td>100 100 100 100</td>
</tr>
<tr>
<td>Multiplicative Pyramid Partitioning</td>
<td>0.06</td>
<td>0.03</td>
<td>0.77 0.13 0.06 0.04</td>
<td>0.01 0.01 0.01 0.02</td>
<td>100 100 100 100</td>
</tr>
<tr>
<td>Additive Pyramid Partitioning</td>
<td>4.35</td>
<td>1.99</td>
<td>0.78 0.11 0.06 0.05</td>
<td>0.70 0.63 0.47 0.81</td>
<td>89 56 63 13</td>
</tr>
<tr>
<td>Additive Pyramid Galerkin Weighting</td>
<td>1.19</td>
<td>0.27</td>
<td>0.61 0.22 0.11 0.06</td>
<td>0.14 0.44 0.36 0.49</td>
<td>100 100 100 93</td>
</tr>
</tbody>
</table>

TABLE 7.5
FOURIER ANALYSIS OF CMR FOR
\[ u = \sin(x) \sin(y) \]
\[ u_0 = u + (\pi-x)(\pi-y)xy \]
\[ u_0^1(x,y) = u_1(x,y) + \sin x \sin y \]

is used, while for the second

\[ u_0^2(x,y) = u_2(x,y) + (\pi-x)(\pi-y)xy \]

is employed. Both trial estimates were chosen because the error involved is not fluctuating rapidly. Care was taken, to avoid an error component that is a multiple of the solution itself, because Corollary (6.8) established that the multiplicative schemes completely remove the error from the trial solution.

The error vector before and after rebalancing is analysed by the fast Fourier inverse transform. This time the reported quantity \( FF(m_1,m_2) \), defined previously by (7.3.1), represents the fraction all Fourier coefficients in the range \((m_1,m_2)\) form of all the Fourier coefficients of the error, after application of the coarse mesh rebalance procedure.

It is useful in this study to report two additional parameters, the first is represented by \( FR(m_1,m_2) \). This is the fraction (of the original error) to which all the diminished Fourier components in the range \( m_1,m_2 \) are reduced.
\[ i.e. \ FR(m_1, m_2) = 1 - \frac{\sum \sum |A_{k_1 k_2}| - |\overline{A}_{k_1 k_2}|}{\sum \sum |A_{k_1 k_2}|} \]

for \[ \frac{|\overline{A}_{k_1 k_2}|}{|A_{k_1 k_2}|} \leq 1 \], where the unspecified summation is again over the range \((m_1, m_2)\); and where \(A_{k_1 k_2}\) and \(\overline{A}_{k_1 k_2}\) represent the Fourier coefficients before and after rebalancing respectively. (A small value of FR indicates considerable reduction in the error component).

The final quantity is the proportion of all terms within a range \((m_1, m_2)\) reduced by CMR, and this is reported as a percentage.

In the rebalancing used here, the coarse grid imposed on the 17x17 fine mesh satisfies the following three criteria for the four methods analysed:

(i) the order of the reduced system of linear equations is the same for each method,
(ii) the number of coarse mesh points for each axis is approximately the square root of the number of fine mesh points (5 and 4 for the disjunctive and pyramid partitioning methods respectively),
(iii) the coarse mesh grid points are positioned as symmetrically as possible.

Of the three methods involving non-Galerkin weighting; the multiplicative pyramid and disjunctive partitioning schemes show a significant reduction in the error of the estimate after CMR is applied. There appears to be a very significant reduction of error in
the low frequency groups as indicted by FR(0,2). For these two methods every individual Fourier component was reduced.

The high values of FF reported for the low frequency components may appear to be disappoointing at first sight. They are a little misleading on their own and this is why the second quantity FR was computed subsequently. The value of FF is high for low frequency terms, simply because the original error introduced in the approximate solution is made up of predominantly low frequency components itself. Even after the rebalance, the error still has a high relative proportion of the low frequency terms even though they are reduced absolutely in magnitude, (a fact reflected in FR). To some extent, a high value of FF may be interpreted as indicating that the rebalance process does not have a significant parasitic effect of exciting high frequency error components for these examples.

The reduction in the error is not restricted to the low end of the spectrum (as indicated by FR). The non-linear multiplicative nature of the schemes allows for this, whereas the same effect would not be anticipated from a simpler mapping operation between coarse and fine grids. This is verified by the two additive forms, where the value of FR for the high frequency terms is significantly higher.

The additive pyramid approach without Galerkin weighting seems of little value compared with the other three procedures. In fact many of the Fourier coefficients are found to have been increased from the
Significant improvement for the additive method is achieved if Galerkin weighting vectors are selected instead of disjunctive weights. Almost all the Fourier terms are reduced, and a marked reduction in the low frequency terms (relative to the higher frequencies) is observed for the two examples. Such vectors, however, require additional computation.

The location of the 4x4 coarse grid system achieved by the three criteria expressed here, is not as favourable to the additive form of rebalance, because the point \((\frac{\pi}{2}, \frac{\pi}{2})\) at which the error is a maximum is not a coarse mesh grid point. An alternative coarse mesh grid that exploited this knowledge overcomes the seemingly high maximum and average error reported for the two sample problems with additive rebalance, provided Galerkin weighting is used. The results of the newer coarse grid mesh for both problems and three of the CMR routines are presented in Table 7.6 The results indicate little change for the very satisfactory multiplicative pyramid form, even though additional coarse mesh lines are introduced to pass through \((\frac{\pi}{2}, \frac{\pi}{2})\). The additive form with disjunctive weighting actually deteriorated on this grid, while the Galerkin weighted version showed a dramatic improvement which now makes it comparable with the multiplicative pyramid form. The multiplicative forms would seem less sensitive to positioning of the coarse grid, for the class of problems considered. This undoubtedly is due to the non-additive nature of the rescaling
TABLE 7.6
FOURIER ANALYSIS OF \(-\nabla^2 u(x,y) = f(x,y)\)

FOR (1) \(u(x,y) = (x-x)(-y)xy + e\)
USING \(u_0(x,y) = u(x,y) + \sin(x)\sin(y),\)
AND (2) \(u(x,y) = \sin(x)\sin(y)\)
USING \(u_0(x,y) = u(x,y) + (x-x)(-y)xy,\)

WITH A COARSE MESH EXPECTED TO FAVOUR THE ADDITIVE SCHEMES
(- indicates no reduction obtained)

<table>
<thead>
<tr>
<th>Method of</th>
<th>(e_M)</th>
<th>(e_A)</th>
<th>(FF)</th>
<th>(FR)</th>
<th>(%) of terms reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>((0,2))</td>
<td>((3,4))</td>
<td>((5,6))</td>
<td>((7,7))</td>
<td>((0,2))</td>
</tr>
<tr>
<td>Problem 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multiplicative</td>
<td>0.08</td>
<td>0.03</td>
<td>0.77</td>
<td>0.12</td>
<td>0.08</td>
</tr>
<tr>
<td>Pyramid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive</td>
<td>0.80</td>
<td>0.30</td>
<td>0.77</td>
<td>0.12</td>
<td>0.08</td>
</tr>
<tr>
<td>Pyramid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive</td>
<td>0.09</td>
<td>0.02</td>
<td>0.40</td>
<td>0.38</td>
<td>0.13</td>
</tr>
<tr>
<td>Galerkin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multiplicative</td>
<td>0.08</td>
<td>0.03</td>
<td>0.77</td>
<td>0.12</td>
<td>0.08</td>
</tr>
<tr>
<td>Pyramid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive</td>
<td>4.94</td>
<td>1.98</td>
<td>0.77</td>
<td>0.11</td>
<td>0.10</td>
</tr>
<tr>
<td>Pyramid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Additive</td>
<td>0.09</td>
<td>0.03</td>
<td>0.40</td>
<td>0.38</td>
<td>0.13</td>
</tr>
<tr>
<td>Galerkin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
process. Despite their better performance, there is considerably more computational overhead in employing multiplicative schemes.

A comparison between the three iterative techniques and CMR was made for the low frequency error component of problem 1. SLOR (with optimal \( \omega \)) and MINI required 8 and 7 iterations respectively to achieve an accuracy comparable with CMR. ICCG did much better, producing a superior result after only 1 iteration.

To study the effect of CMR in reducing high frequency error, the following initial estimate was used on the two-dimensional problem

\[ u_0(x, y) = (\pi - x)(\pi - y)xy + e \]

where \( e = \sin kx \sin ky \) for increasing \( k \). Once \( k \) exceeded the order of the coarse grid, application of CMR not surprisingly was found actually to increase the error in the solution estimate. It would seem that a good computational strategy, where the error is known to be highly oscillatory, is to perform a few ordinary iterations to remove much of the high frequency spectrum before application of CMR.

The examples chosen so far provide a rather limited test of CMR methods, however, the results tend to favour the multiplicative approach. In Table 7.7 several additional model problems are described, (based on the equation \(-\nabla^2 u(x, y) = f(x, y)\)). Each is solved on a 17x17 fine grid (including boundaries) with a coarse mesh
### TABLE 7.7
SEVEN FURTHER TRIAL FUNCTIONS FOR \(-\nabla^2 u(x,y) = f(x,y)\)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Actual Solution (u)</th>
<th>Trial Solution (u_0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.5</td>
<td>(u + \sin x \sin y)</td>
</tr>
<tr>
<td>2</td>
<td>(\sin x \sin y)</td>
<td>(u + \sin 15x \sin 15y)</td>
</tr>
<tr>
<td>3</td>
<td>(\sin 15x \sin 15y)</td>
<td>(u + \sin x \sin y)</td>
</tr>
<tr>
<td>4</td>
<td>((\pi-x)(\pi-y)xy)</td>
<td>(u + \sin 15x \sin 15y)</td>
</tr>
<tr>
<td>5</td>
<td>((\pi-x)(\pi-y)xy)</td>
<td>(u + \sin 15x \sin 15y)</td>
</tr>
<tr>
<td>6</td>
<td>([(\pi-x)(\pi-y)xy]^2)</td>
<td>(\sqrt{u})</td>
</tr>
<tr>
<td>7</td>
<td>((\pi-x)(\pi-y)xy)</td>
<td>(u^2)</td>
</tr>
</tbody>
</table>
grid of 4x4 elements designed to ensure a node is located at the maximum solution value to assist the additive forms. The results of the four CMR methods are presented in Table 7.8, where instead of reporting absolute error magnitudes \((E_M \text{ and } E_A)\), the ratio of errors before and after CMR are given because of the high scale variation in error that occurs in the respective solutions.

The results show that the additive pyramid partitioning scheme with Galerkin weighting is superior on all but problem 2, which is an unrealistic candidate for CMR having a highly oscillatory error. In all other problems the additive Galerkin approach reduces both measurements of error. With the exception of problem 2, all the schemes show the percentage of low frequency error components reduced is never lower than the percentage of high frequency terms, and in many cases is considerably greater. In addition the extent of the reduction (as measured by FR) is significantly greater for the low frequency components.

Although the additive Galerkin scheme never increases errors, the same cannot be said of the others, particularly the multiplicative forms on problems 4 and 5. The identical behaviour of the respective additive method results on problems 6 and 7 could have been predicted from Chapter 6. The same applies to the respective multiplicative results for problems 2 and 3.
### TABLE 7.8
FOURIER ANALYSIS OF SIX TEST PROBLEMS

<table>
<thead>
<tr>
<th>Problem</th>
<th>EN Ratio</th>
<th>EN Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.51</td>
<td>0.46</td>
</tr>
<tr>
<td>2</td>
<td>0.98</td>
<td>0.97</td>
</tr>
<tr>
<td>3</td>
<td>0.98</td>
<td>0.97</td>
</tr>
<tr>
<td>4</td>
<td>12.0</td>
<td>33.0</td>
</tr>
<tr>
<td>5</td>
<td>4x10^5</td>
<td>5x10^5</td>
</tr>
<tr>
<td>6</td>
<td>0.70</td>
<td>0.73</td>
</tr>
<tr>
<td>7</td>
<td>0.92</td>
<td>0.60</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FF</th>
<th>FF</th>
<th>FF</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0-2)</td>
<td>(3-4)</td>
<td>(5-6)</td>
</tr>
<tr>
<td>0.55</td>
<td>0.57</td>
<td>0.65</td>
</tr>
<tr>
<td>0.73</td>
<td>0.89</td>
<td>0.75</td>
</tr>
<tr>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FR</th>
<th>FR</th>
<th>$%$ of terms reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0-2)</td>
<td>(3-4)</td>
<td>(5-6)</td>
</tr>
<tr>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

#### Disjunctive Partitioning

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.51</td>
<td>0.98</td>
<td>0.98</td>
<td>12.0</td>
<td>1x10^5</td>
<td>0.70</td>
<td>0.92</td>
</tr>
<tr>
<td>0.56</td>
<td>0.73</td>
<td>0.05</td>
<td>0.67</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

#### Multiplicative Pyramid Partitioning

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.58</td>
<td>0.98</td>
<td>0.98</td>
<td>1.5</td>
<td>1.4</td>
<td>0.66</td>
<td>0.07</td>
</tr>
<tr>
<td>0.95</td>
<td>0.62</td>
<td>0.61</td>
<td>0.44</td>
<td>0.04</td>
<td>0.41</td>
<td>0.10</td>
</tr>
</tbody>
</table>

#### Additive Pyramid Partitioning

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.80</td>
<td>3.1</td>
<td>0.80</td>
<td>1.0</td>
<td>0.79</td>
<td>0.52</td>
<td>0.60</td>
</tr>
<tr>
<td>0.03</td>
<td>0.81</td>
<td>0.66</td>
<td>0.56</td>
<td>0.56</td>
<td>0.56</td>
<td>0.00</td>
</tr>
</tbody>
</table>

#### Additive Pyramid Partitioning, Galerkin weighting

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.09</td>
<td>0.99</td>
<td>0.09</td>
<td>0.22</td>
<td>0.51</td>
<td>0.14</td>
<td>0.14</td>
</tr>
<tr>
<td>0.05</td>
<td>1.0</td>
<td>0.05</td>
<td>0.12</td>
<td>0.59</td>
<td>0.12</td>
<td>0.12</td>
</tr>
</tbody>
</table>

#### $\%$ of terms reduced

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
7.5 CONCLUSION

The global mode Fourier analysis provides a more mathematically robust tool to analyse the behaviour of iterative methods and CMR than the local mode approach of Brandt. The two methods, however, lead to somewhat similar conclusions.

The ICCG and MINI iterative schemes have been demonstrated to decrease rapidly the high frequency error components, while the behaviour of SLOR is related to the value used for \( \omega_{opt} \). The CMR schemes favour the removal of low frequency error and on these grounds promise to be of particular value when employed with MINI and ICCG.

By their nature the CMR techniques do not reduce errors exclusively in the low frequency range. Some reduction in high frequency error is evident in the results of Section 7.4. Consequently it is possible CMR may assist convergence of the relaxation approaches. The ultimate test will be undertaken in Chapter 9 where real reactor configurations are studied, and different CMR methods applied.
CHAPTER 8

A COMPARISON OF SLOR, MINI AND ICCG ON REAL REACTOR MODELS

A.A. Milne, *Us Two*

8.1 INTRODUCTION

Several new ideas were introduced in previous chapters and it was suggested they may provide a more efficient means of solving typical everyday problems in reactor physics. In this chapter, performances of the three iterative schemes for the spatial solution process are compared. Then MINI is considered as a candidate for accelerating energy group convergence. To avoid confusion in the comparison, study of various forms of region rebalance is delayed until Chapter 9. The value of energy rebalancing, however, will be obvious from some results. The evaluation of energy rebalancing is included here rather than Chapter 9, because the method is an alternative method to MINI for accelerating energy group convergence.

In this chapter, the order in which results are presented reflects the introduction of the appropriate technique in POW3D. This is in keeping with the methodology chosen at the start of the project. The decision to test the techniques initially on two-dimensional
geometries was made because the implementation is orders of magnitude easier, and because all the non-zero matrix components can be retained in fast memory. This removed the necessity to develop a sophisticated data management scheme at the outset. Only with the ideas then firmly established (and in the case of MINI ultimately refined), was it worthwhile undertaking the final step of three-dimensional implementation.

8.2 A TWO-DIMENSIONAL MODEL PROBLEM

A two-dimensional model problem, first used in some numerical studies by Wachspress [1966] is investigated. This is a one group problem, without any neutron loss ($\sigma_r = 0$)

$$-\nabla \cdot \nabla \phi = 1,$$

with $\phi = 0$ on external boundaries and $\phi$ and $\nabla \phi$ continuous across internal boundaries, for the square reactor configuration of Figure 8.1.

In the first example, in order to estimate the average time per iteration, the reactor was solved for three different mesh refinements (of equal mesh spacing) with SLOR and MINI. All the times reported in this work (unless otherwise stated) are for an IBM 3031 running under the MVS operating system with 4 megabytes of real storage. The times given in Table 8.1 are expressed as central processing unit (CPU)
SOLUTION LINE AND DIRECTION

$\Delta y = 1 \text{cm}$

$\Delta x = 1 \text{cm}$

Problem Number

<table>
<thead>
<tr>
<th>Problem Number</th>
<th>$D_1$</th>
<th>$D_2$</th>
<th>$D_3$</th>
<th>$D_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5</td>
<td>25</td>
<td>125</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>10</td>
<td>100</td>
<td>1000</td>
</tr>
</tbody>
</table>

Model 2-D reactor configuration

Figure 8.1
TABLE 8.1
Timing comparison for SLOR, MINI and ICCG on 2-D model problem. Times are in microseconds per iteration pass per number of equations

<table>
<thead>
<tr>
<th>N</th>
<th>10^2</th>
<th>30^2</th>
<th>100^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR (FT)</td>
<td>192</td>
<td>169</td>
<td>162</td>
</tr>
<tr>
<td>MINI (FT)</td>
<td>240</td>
<td>213</td>
<td>211</td>
</tr>
<tr>
<td>ICCG (FT)</td>
<td>263</td>
<td>226</td>
<td>213</td>
</tr>
<tr>
<td>SLOR (ASM)</td>
<td>144</td>
<td>121</td>
<td>115</td>
</tr>
<tr>
<td>MINI (ASM)</td>
<td>180</td>
<td>159</td>
<td>156</td>
</tr>
</tbody>
</table>

TABLE 8.2
Number of iterations to converge 2-D model problem

<table>
<thead>
<tr>
<th>Problem</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>59</td>
<td>56</td>
<td>58</td>
</tr>
<tr>
<td>MINI</td>
<td>44</td>
<td>36</td>
<td>34</td>
</tr>
<tr>
<td>ICCG</td>
<td>25</td>
<td>25</td>
<td>27</td>
</tr>
</tbody>
</table>

TABLE 8.3
Influence of unequal mesh spacing on convergence

<table>
<thead>
<tr>
<th>Method</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>225</td>
</tr>
<tr>
<td>MINI</td>
<td>71</td>
</tr>
<tr>
<td>ICCG</td>
<td>31</td>
</tr>
</tbody>
</table>
### TABLE 8.4

**INFLUENCE OF MESH REFINEMENT ON CONVERGENCE**

<table>
<thead>
<tr>
<th>h cm</th>
<th>2</th>
<th>1</th>
<th>2/3</th>
<th>1/3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>27</td>
<td>56</td>
<td>76</td>
<td>167</td>
</tr>
<tr>
<td>MINI</td>
<td>17</td>
<td>34</td>
<td>62</td>
<td>216</td>
</tr>
<tr>
<td>ICCG</td>
<td>15</td>
<td>25</td>
<td>38</td>
<td>70</td>
</tr>
</tbody>
</table>

### TABLE 8.5

**NUMBER OF ITERATIONS TO CONVERGE 2-D 4-GROUP MOATA PROBLEM**

<table>
<thead>
<tr>
<th>Method Inner Layer</th>
<th>Method Energy Group</th>
<th>Energy Group</th>
<th>Total</th>
<th>Time min. FORTRAN</th>
<th>Time min. Assembler</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>SLOR</td>
<td>GS</td>
<td>144</td>
<td>91</td>
<td>78</td>
<td>118</td>
</tr>
<tr>
<td>MINI</td>
<td>GS</td>
<td>97</td>
<td>93</td>
<td>71</td>
<td>84</td>
</tr>
<tr>
<td>ICCG</td>
<td>GS</td>
<td>86</td>
<td>82</td>
<td>65</td>
<td>74</td>
</tr>
<tr>
<td>SLOR</td>
<td>MINI</td>
<td>172</td>
<td>100</td>
<td>86</td>
<td>124</td>
</tr>
<tr>
<td>MINI</td>
<td>MINI</td>
<td>100</td>
<td>93</td>
<td>69</td>
<td>82</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>86</td>
<td>84</td>
<td>65</td>
<td>72</td>
</tr>
</tbody>
</table>
microseconds per iteration pass per number of equations. At a later stage the ICCG routines were developed, and results are included now for comparison. Times are tabulated for the three methods coded in FORTRAN and run with the IBM FORTRAN H compiler, with optimisation level 2, and as well for SLOR and MINI coded in assembler language.

The number of iterations required to reach a converged solution with a relative error of less than $10^{-4}$ are reported, in Table 8.2, starting with a trial solution $\phi_{ij} = 1$. The analytic value of $\omega_{\text{opt}} = 1.6406$ (with a 10 x 10 equal spaced mesh) for SLOR was not supplied, but rather the code determined the estimate of $\omega_{\text{opt}} = 1.6404$. This was calculated after several initial iterations, with the estimate $\omega = 1.5$ used to commence the calculation. Once the improved estimate becomes available the code switches over to it.

The effect of unequal mesh spacing on convergence is examined for problem 2 in Table 8.3. The mesh spacing in cm for both the x and y directions is 0.5, 0.5, 0.5, 0.01, 0.49, 1.2, 1.0, 3.0, 1.0 repeated for the next 10 increments. The table entries are again the number of iterations required for convergence.

In Table 8.4 the number of iterations required to solve problem 2 with a varying uniform mesh ($\Delta x = \Delta y = h$) are displayed.
The results in Tables 8.2 and 8.3 show MINI is satisfactory except where the mesh is everywhere tight (i.e. for the h = 1/3 cm results in Table 8.4). MINI can stand a locally 'tight' mesh situation (Table 8.3), but when the 'tightness' extends everywhere, the implicit coupling to the neighbouring point (Figure 4.3) is not enough, because that point is so closely coupled to its own forward neighbouring point. MINI was developed to study three-dimensional problems and extreme mesh 'tightness' is unlikely in the problems being tackled because of limited capabilities of present day computers. In any case, real reactor calculations are the ultimate test, and a few such calculations are considered now.

8.3 REAL 2-D REACTOR CALCULATIONS

Four reactor calculations are considered for two-dimensional configurations, three of which are taken from the literature. Each problem is calculated by the three iterative schemes coded in optimised FORTRAN, and repeated with SLOR and MINI only, written in assembler language. The option of rebalance is enabled for all problems; group rebalance is performed (as indicated in Section 6.4) and throughout this chapter region rebalance is of the simple form disjunctive partitioning and region balance weighting (Section 6.4A).
8.3.1 Steady State Moata 2-D Thermal Reactor

Moata is a thermal reactor at the AAEC's Research Establishment. It is a 100 kW Argonaut-type reactor of highly enriched uranium clad in aluminium plates cooled and moderated by light water with a graphite reflector. It is represented by a two-dimensional (x,y) geometry for a quarter reactor with 22 x 16 mesh intervals. Four energy groups covering the energy range are sufficient for this steady state reactor calculation, and three of the four groups involve the upscatter of thermal neutrons. For Moata, however, unlike some other thermal reactors, the extent of upscattering is not particularly significant. The 'group layer' is treated by both Gauss-Seidel and MINI iterations. Unlike the previous model problem, this is a criticality (eigenvalue) calculation, and Chebyshev extrapolation is used (Section 3.3) for the (outer) eigenfunction determination. The data details for Moata are given by Pollard [1974]. The calculation is typical of a 'yesterday' problem, likely to be tackled by a two-dimensional code some ten or more years ago. The calculation contains no real difficulties. Table 8.5 shows the total number of 'inner layer' iterations and the machine time (in minutes) required to converge the solution to an 'outer layer' relative error of less than $10^{-4}$, starting from a trial solution of a homogeneous reactor of the same size.
8.9

The machine times are no longer averaged from several runs because IBM's MVS operating system is capable of more accurate accounting than the earlier MVT. The timing results reported here sometime demonstrate marked differences from those published previously, even after allowing for extrapolation from the IBM 360/65 to the IBM 3031. This is due to different timing mechanisms in the newer operating system, where all overheads are now charged against the user [IBM 1977]. The differences are noticeable, particularly for the three-dimensional problems of Section 8.5.

To support the results of Table 8.1, the average time spent per iteration pass is calculated for the Moata model and presented in Table 8.6. In POW3D, the SLOR and MINI routines are both implemented in double precision arithmetic, while ICCG with its lower sensitivity to rounding error is in single precision arithmetic, except for the LDL\(^T\) approximate decomposition.

The need for double precision arithmetic for all of MINI is probably unnecessary, so the FORTRAN version was altered to perform the $\gamma$ determination in single precision. This test showed a time saving of about 10 per cent. The relative performance of the methods is consistent for the two problems. In later experiments the timing reported for MINI is for the entire implementation in double precision arithmetic.
TABLE 8.6
AVERAGE TIME PER ITERATION PASS FOR 2-D MOATA

<table>
<thead>
<tr>
<th>Type of Iteration</th>
<th>Language</th>
<th>Average Time/iteration sec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>Assembler</td>
<td>$0.69 \times 10^{-3}$</td>
</tr>
<tr>
<td>SLOR</td>
<td>FORTRAN</td>
<td>$0.99 \times 10^{-3}$</td>
</tr>
<tr>
<td>MINI (double precision $\gamma'$s)</td>
<td>Assembler</td>
<td>$0.97 \times 10^{-3}$</td>
</tr>
<tr>
<td>MINI (double precision $\gamma'$s)</td>
<td>FORTRAN</td>
<td>$0.13 \times 10^{-2}$</td>
</tr>
<tr>
<td>MINI (single precision $\gamma'$s)</td>
<td>FORTRAN</td>
<td>$0.12 \times 10^{-2}$</td>
</tr>
<tr>
<td>ICCG</td>
<td>FORTRAN</td>
<td>$0.15 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

TABLE 8.7
NUMBER OF ITERATIONS TO CONVERGE 2-D 5-GROUP TRIGA PROBLEM
*a small variation in the total number of iterations was noticed for the FORTRAN version.

<table>
<thead>
<tr>
<th>Method</th>
<th>Method</th>
<th>Energy Groups</th>
<th>Time min.</th>
<th>Time min.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inner</td>
<td>Group</td>
<td>FORTRAN</td>
<td>Assembler</td>
</tr>
<tr>
<td></td>
<td>Layer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 2 3 4 5 Total</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>GS</td>
<td>112 102 80 130 212 636</td>
<td>2.61</td>
<td>2.01</td>
</tr>
<tr>
<td>MINI</td>
<td>GS</td>
<td>85 58 61 68 75 347</td>
<td>2.08*</td>
<td>1.72</td>
</tr>
<tr>
<td>ICCG</td>
<td>GS</td>
<td>75 34 36 53 93 297</td>
<td>1.83</td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>MINI</td>
<td>112 103 80 119 217 631</td>
<td>2.73</td>
<td>2.13</td>
</tr>
<tr>
<td>MINI</td>
<td>MINI</td>
<td>86 56 59 68 73 342</td>
<td>2.06*</td>
<td>1.75</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>76 34 36 53 121 325</td>
<td>1.97</td>
<td></td>
</tr>
</tbody>
</table>
A two-dimensional model of a TRIGA thermal reactor mockup, in (r,z) geometry and described by Froehlich [1969], is studied for all three schemes. The configuration contains a central hole, used for irradiation purposes. The void is surrounded by a stainless steel tube with iron plugs at the top and bottom. The fuel is 20 per cent enriched uranium in zirconium hydride canned in stainless steel and cooled by light water. The 'energy layer' is covered by 5 groups (2 of which involve upscatter), and a steady state half-reactor calculation is undertaken with a 32x30 mesh. (In Froehlich's study, 42 radial mesh intervals were used, whereas here (Table 8.7) the radial mesh consists of 8 intervals of 1.5 cm, 2 of 0.5 cm, 2 of 0.375 cm, 15 of 1.16 cm and 5 of 1.08 cm, corresponding to an overall radius of 36.55 cm.) In general, holes cause difficulties in diffusion theory calculations and this example is no exception. The problem is undertaken as an example of a difficult two-dimensional calculation presented to a reactor neutronics code. An outer layer error level of $10^{-3}$ was taken for all TRIGA runs instead of the usual $10^{-4}$. This was not because convergence failed at the usual level of $10^{-4}$ but because interaction of other portions of the iterative processes (for this particular eigenvalue problem) perturbed the process to an extreme degree and made a comparison of the inner iterative schemes difficult.
The main difficulty with the TRIGA problem is the flat flux solution across the void. Graphic inspection of the iteration process using 'movies' (Section 4.2), revealed that convergence of the flux is difficult because changes in one section are propagated slowly across the void. Region rebalance is almost essential for such problems, but even so difficulties exist (Section 9.2).

The results suggest that the three techniques are somewhat comparable for this problem, however, MINI and ICCG have a slight edge over SLOR.

8.3.3 HIFAR Reactor

No comparison would be complete without a calculation for the AAEC's principal reactor HIFAR. HIFAR is a DIDO class reactor, initially designed as a materials testing reactor, but whose principal functions now are to provide neutron beam sources for solid state physics research projects and to produce radiopharmaceuticals on a commercial basis. The model [Harrington 1980], involves a 59x59 grid and is covered by 5 energy groups, three of which involve upscatter. The results for the real and adjoint calculations, with all three iterative schemes, are presented in Tables 8.8 and 8.9 respectively. The good performance of ICCG is evident for both calculations, while the ability of MINI to handle the energy group iteration is apparent for the adjoint problem.
### TABLE 8.8
NUMBER OF ITERATIONS TO CONVERGE THE 2-D 5-GROUP HIFAR PROBLEM

<table>
<thead>
<tr>
<th>Method Inner Layer</th>
<th>Method Energy Group</th>
<th>Energy Groups</th>
<th>Time min.</th>
<th>Time min.</th>
</tr>
</thead>
<tbody>
<tr>
<td>G5</td>
<td>SLOR GS</td>
<td>625</td>
<td>197</td>
<td>126</td>
</tr>
<tr>
<td>MINI GS</td>
<td>547</td>
<td>208</td>
<td>199</td>
<td>144</td>
</tr>
<tr>
<td>ICCG GS</td>
<td>378</td>
<td>186</td>
<td>142</td>
<td>284</td>
</tr>
<tr>
<td>MINI MINI GS</td>
<td>205</td>
<td>143</td>
<td>145</td>
<td>136</td>
</tr>
<tr>
<td>ICCG MINI GS</td>
<td>379</td>
<td>102</td>
<td>94</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td></td>
<td>212</td>
<td>95</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>222</td>
<td>63</td>
<td>71</td>
</tr>
</tbody>
</table>

### TABLE 8.9
NUMBER OF ITERATIONS TO CONVERGE THE 2-D 5-GROUP HIFAR ADJOINT PROBLEM

<table>
<thead>
<tr>
<th>Method Inner Layers</th>
<th>Method Energy Groups</th>
<th>Energy Groups</th>
<th>Time min.</th>
<th>Time min.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR GS</td>
<td>197</td>
<td>126</td>
<td>126</td>
<td>2134</td>
</tr>
<tr>
<td>MINI GS</td>
<td>547</td>
<td>208</td>
<td>199</td>
<td>144</td>
</tr>
<tr>
<td>ICCG GS</td>
<td>378</td>
<td>186</td>
<td>142</td>
<td>284</td>
</tr>
<tr>
<td>MINI MINI GS</td>
<td>205</td>
<td>143</td>
<td>145</td>
<td>136</td>
</tr>
<tr>
<td>ICCG MINI GS</td>
<td>379</td>
<td>102</td>
<td>94</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td>212</td>
<td>95</td>
<td>100</td>
<td>87</td>
</tr>
<tr>
<td></td>
<td>222</td>
<td>63</td>
<td>71</td>
<td>53</td>
</tr>
</tbody>
</table>
8.3.4 Fast Reactor Benchmark Kinetics

A two-dimensional \((r,z)\) geometry model of a fast reactor used as a benchmark for kinetics studies was reported by Buffoni et al. [1975], who described the calculation as a 'super-prompt critical transient with feedback, 2 group neutron diffusion problem in a fast reactor'. A POW3D calculation is undertaken for the full core with a 32x44 mesh. A simple model of temperature feedback adjusts cross section data (and hence the elements of the matrix \(A\), equation 3.3.2). Ninety-six time steps are used to calculate the reactor behaviour for 0.1 seconds, and the results are given in Table 8.10. For this and subsequent runs, only times with the assembler version of SLOR and MINI are reported.

8.3.5 A Kinetics Study with Significant Upscatter

In Table 8.11 the influence of using a group MINI approach is shown for a two-dimensional, two group, kinetics calculation reported by Hageman and Yasinsky [1969] and calculated by Pollard [1977] amongst others. The spatial mesh for the quarter \((x,y)\) reactor model is 11x11. The time dependence arises from step insertion of \(\frac{1}{2}\%\) of reactivity, and fission emission is included in the 'scattering matrix'. This particular problem is tackled because convergence using the GS method for the group iteration without energy group rebalance is extremely slow [Pollard 1977]. The number of inner iterations reported in Table 8.11 are an average for the 20 kinetic time steps
### TABLE 8.10
NUMBER OF ITERATIONS FOR 2-D FAST REACTOR KINETICS STUDY WITH 96 TIME STEPS

<table>
<thead>
<tr>
<th>Energy Group</th>
<th>1</th>
<th>2</th>
<th>Total</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>3551</td>
<td>2466</td>
<td>6017</td>
<td>47.82</td>
</tr>
<tr>
<td>MINI</td>
<td>3245</td>
<td>2746</td>
<td>5991</td>
<td>55.56</td>
</tr>
<tr>
<td>ICCG</td>
<td>2621</td>
<td>1696</td>
<td>4317</td>
<td>55.86</td>
</tr>
</tbody>
</table>

### TABLE 8.11
AVERAGE NUMBER OF INNER ITERATIONS FOR A 2-D KINETICS CALCULATION

<table>
<thead>
<tr>
<th>Method</th>
<th>Group</th>
<th>Group</th>
<th>1</th>
<th>2</th>
<th>Total</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plane</td>
<td>Group</td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>GS</td>
<td>54</td>
<td>44</td>
<td></td>
<td>98</td>
<td>2.86</td>
</tr>
<tr>
<td>SLOR</td>
<td>MINI</td>
<td>27</td>
<td>19</td>
<td></td>
<td>46</td>
<td>1.54</td>
</tr>
<tr>
<td>MINI</td>
<td>GS</td>
<td>57</td>
<td>43</td>
<td></td>
<td>100</td>
<td>3.00</td>
</tr>
<tr>
<td>MINI</td>
<td>MINI</td>
<td>50</td>
<td>20</td>
<td></td>
<td>70</td>
<td>1.80</td>
</tr>
<tr>
<td>ICCG</td>
<td>GS</td>
<td>48</td>
<td>40</td>
<td></td>
<td>88</td>
<td>3.41</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>31</td>
<td>18</td>
<td></td>
<td>49</td>
<td>1.73</td>
</tr>
</tbody>
</table>
studied.

8.4 CONCLUSIONS FROM 2-D REACTOR STUDIES

In terms of a decreasing order in the number of iterations required to obtain a converged solution, the preceding results suggest the three iterative techniques may be ranked SLOR, MINI and ICCG for most two-dimensional problems. As a measure of their effective worth, however, iteration counts alone are insufficient to achieve an adequate conclusion. The number of floating point operations required per point per iteration are given in Table 8.12, for the three plane methods as implemented in POW3D. These are pertinent to the situation where the partial LDL\(^T\) decomposition and \(\omega\) estimation are already completed. They provide some measure of the penalty associated with the most efficient implementation of the methods. The heavy penalty suggested for ICCG is not reflected in the actual timing comparisons of Tables 8.1 and 8.6, because the use of single precision arithmetic is to its advantage. In actual timing of reactor problems, MINI incurs a larger penalty than the floating point operation count suggests. This is due largely to additional overheads associated with use and determination of \(\gamma\)'s.

In terms of actual computational (CPU) times, SLOR and MINI are comparable for the problems considered, while ICCG appears superior, provided the times are discounted to allow for the different computer languages used.
TABLE 8.12
NUMBER OF FLOATING POINT OPERATIONS PER ITERATION PER POINT FOR 2-DIMENSIONAL FORMS OF THE ITERATIVE SCHEMES AS IMPLEMENTED IN POW3D

<table>
<thead>
<tr>
<th>Type of Arithmetic</th>
<th>SLOR</th>
<th>MINI</th>
<th>ICCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>* /</td>
<td>10</td>
<td>12</td>
<td>20</td>
</tr>
<tr>
<td>+ -</td>
<td>6</td>
<td>7</td>
<td>17</td>
</tr>
</tbody>
</table>
Writing the iterative schemes in Assembler language is not a trivial task, and the implementation of ICCG is considerably more involved than that of the other two schemes. For optimum performance on two-dimensional problems this would appear a worthwhile undertaking for the future.

For two-dimensional problems, all the spatial data relevant to each energy calculation are held in memory. Consequently the I/O overheads for the three methods are not significantly different. (SLOR uses less memory because unlike MINI it doesn't require a plane of $S$'s, nor does it require memory of a LDL$^T$ partial decomposition.)

When considerable upscattering of neutrons occur the use of MINI to accelerate convergence of the energy groups gives a significant saving. Table 8.11 suggests almost a 50 per cent reduction in time, when MINI is used for this mode. The reduction applies when any of the three iterative schemes are used to solve the plane. The time savings for the HIFAR adjoint calculation (Table 8.9) also are considerable. For situations where upscatter is less important (e.g. Moata) there appears to be little or no computational penalty in having the option enabled. When MINI drives the groups, the diagonal terms of the coefficient matrix are perturbed, and this may affect the appropriateness of $\omega$ used for plane SLOR. The results here, however, indicate there is no need to re-determine $\omega$. The LDL$^T$ decomposition for ICCG is affected similarly. The approach chosen with ICCG, however, is to repeat the decomposition when the diagonal terms are
perturbed, because the overhead (compared with $\omega$ evaluation) is slight. It is not known whether it is really necessary.

From the historical perspective, MINI and SLOR were deemed comparable for two-dimensional studies. The decision to proceed with a three-dimensional formulation of MINI was made because:

(i) the reduced number of iterations observed with MINI was expected to pay dividends (despite the computational overhead) when one iterative scheme was used to drive another, and

(ii) MINI does not suffer (additional costs for re-estimation of $\omega$) when significant kinetic feedback occurs.

8.5 SOME 3D REACTOR STUDIES

Results for five three-dimensional $k$ eigenvalue test problems are reported: three are for the thermal reactor Moata, and the others are for fast reactors.

As indicated in Chapter 5, the implementation of two and three (spatial) dimensional algorithms to perform the inner iterative loops are different. This is particularly true of the ICCG algorithm. The organisation for a two-dimensional calculation follows the usual mathematical definition, but to achieve an efficient three-dimensional
version for the larger problem, the structure is altered as shown in Section 5.5. For this case it is not possible to reuse any of the simpler two-dimensional code. MINI and SLOR are block methods for the three-dimensional problems. It is only the inner block (equivalent to a two dimensional section) that is coded in assembler language.

With MINI, for any point \((i,j,k)\) in three space, two \(\gamma\) parameters are required as shown in Figure 5.1. The first, \(\gamma_1\) relates the current point to the forward point in the plane, while the second, \(\gamma_2\) relates the current point to the forward point out of the plane. In P0W3D a GS start \((\gamma = 0)\) is used for the first of each \((x,y)\) plane passes, for the first pass in the \(z\) direction and for the first upscatter) pass for each energy group. Thereafter, for all subsequent passes the variation that occurred in the fluxes for the previous one is recalled, so a true MINI start is made next time.

SLOR requires a number of \(\omega\) estimates for each energy group, \((N_z+1)\) in all, one for each block (plane) SLOR process, and one for the between planes SLOR iteration. It is only when all the \(\omega\)'s are determined that the region rebalance acceleration is applied, otherwise the iterative scheme for \(\omega\) determination is disturbed.

For the two block methods SLOR and MINI, the between plane inner iteration is converged to 0.1 times the accuracy required of the outer Chebyshev extrapolation scheme, and the in plane iterations to 0.07 of
this accuracy. If ICCG is a global driver it is converged to 0.1 times the outer accuracy. When ICCG is used to solve individual planes (being driven by either SLOR or MINI in the $z$ direction) the usual fraction 0.07 is employed. Various combinations of plane and $z$ drivers tested in the three-dimensional calculations are reported here.

8.5.1 Steady State 3-D Thermal Moata Reactor

The two-dimensional Moata model of Section 8.3 is now extended, and the buckling correction is replaced by a more detailed analysis of the third dimension. Details of the two-dimensional model [Pollard 1974] together with Figure 8.2 define the reactor adequately. Because of symmetry, only one eighth of the reactor is considered. A 23x17x17 spatial finite difference scheme with edge mesh points is used. Four energy groups are considered, leading to a system of approximately $2.5 \times 10^4$ unknowns. Three of the four energy groups involve upscatter.

In Table 8.13 the number of inner iterations and the machine times for all combination of methods are recorded. The calculation is repeated with refined mesh in the $y$-direction (half mesh spacing, i.e. 33 solution points orthogonal to the solution line used for the SLOR and MINI plane drivers).
<table>
<thead>
<tr>
<th>Method</th>
<th>(x,y) Plane Iterations</th>
<th>Total (x,y) Plane Iterations</th>
<th>Total z Iterations</th>
<th>Normal Mesh Spacing</th>
<th>1/2 Mesh Spacing in y-direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOR</td>
<td>4075</td>
<td>2800</td>
<td>2610</td>
<td>8334</td>
<td>100.25</td>
</tr>
<tr>
<td>MINI</td>
<td>2981</td>
<td>3699</td>
<td>2328</td>
<td>8214</td>
<td>49.53</td>
</tr>
<tr>
<td>SOR</td>
<td>3481</td>
<td>6805</td>
<td>2616</td>
<td>6182</td>
<td>49.53</td>
</tr>
<tr>
<td>MINI</td>
<td>2724</td>
<td>2913</td>
<td>1998</td>
<td>6198</td>
<td>49.53</td>
</tr>
<tr>
<td>SOR</td>
<td>3059</td>
<td>4031</td>
<td>2733</td>
<td>3170</td>
<td>49.53</td>
</tr>
<tr>
<td>MINI</td>
<td>2053</td>
<td>2032</td>
<td>1471</td>
<td>3093</td>
<td>49.53</td>
</tr>
<tr>
<td>ICCG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOR</td>
<td>7189</td>
<td>14831</td>
<td>19652</td>
<td>53768</td>
<td></td>
</tr>
<tr>
<td>MINI</td>
<td>4098</td>
<td>5151</td>
<td>4221</td>
<td>19652</td>
<td></td>
</tr>
<tr>
<td>SOR</td>
<td>6375</td>
<td>3568</td>
<td>3568</td>
<td>3170</td>
<td></td>
</tr>
<tr>
<td>MINI</td>
<td>3095</td>
<td>4031</td>
<td>2733</td>
<td>3170</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>3654</td>
<td>4466</td>
<td>3093</td>
<td>3170</td>
<td></td>
</tr>
<tr>
<td>SOR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MINI</td>
<td>3654</td>
<td>4466</td>
<td>3093</td>
<td>3170</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>2439</td>
<td>2073</td>
<td>1611</td>
<td>3170</td>
<td></td>
</tr>
</tbody>
</table>
Examination of the results for the 4 energy group Moata model reveals that whenever SLOR is driving the z direction, considerably more z passes are required than for either MINI or ICCG. This is reflected in the reported CPU time used. Considerably more z passes are required for the global form of ICCG, than for any other scheme where MINI drives the third-dimension. As expected, the arithmetic overhead in the plane is significantly more (than with ICCG) when block forms of MINI or SLOR are employed. It would appear that in switching from the all MINI block method to ICCG, computational effort expended in the plane is shifted to the z direction. Attempts to redress this unfortunate shift by using ICCG for the plane with a MINI (z) driver, lowers the number of z passes dramatically. The same benefit is not reaped with the ICCG SLOR combination. Excluding those combinations where ICCG is not involved, the CPU time required favours MINI over SLOR.

To compare ICCG and other combinations on the basis of CPU time, it is necessary to allow for the variation in performance due to different computer languages. Because ICCG is in FORTRAN and the SLOR and MINI plane drivers are written in assembler, it is best to adjust the assembler routines upwards. The average time for individual iterations (Table 8.6) leads to a correction factor of (0.013/0.0097) for a typical (x,y) plane of the Moata type. Of the 21.32 minutes required for the MINI-MINI solved Moata with normal mesh spacing, 10 minutes is spent in the inner MINI. Application of the correction factor leads to a time of 24.72 minutes, showing MINI-MINI and ICCG
are comparable. A similar correction for SLOR-SLOR shows SLOR is marginally slower.

For the half mesh spacing in the y direction, the SLOR-SLOR combination appears rather hopeless. This is due partly to the difficulties SLOR faced in obtaining estimates for \( \omega \), which arise even before group MINI is enabled resulting in an altered diagonal structure. Corrections to the half mesh timing results for MINI shows it again is comparable with ICCG. Most of the other comparisons from the four group Moata problem carry across to the half mesh spacing calculation.

A second variation on Moata is considered. In this the number of energy groups is increased, seven of which now involve upscatter. This leads to a linear system of approximately \( 5 \times 10^4 \) unknowns. Because so many groups are involved with the upscatter of thermal neutrons, GS and MINI are expected to differ significantly in their ability to handle the energy group 'layer'. It is anticipated the process of energy group rebalance also may assist convergence because so many thermal groups are involved.

Results are presented in Table 8.14 for GS and MINI group iterations, combined with energy group rebalance option both enabled and disabled. (Discussion of the effects of region rebalancing is delayed until Chapter 9).
TABLE 8.14
RESULTS OF THE 3-D 8-GROUP MOATA MODEL
*Convergence never obtained in time available; extrapolation used to obtain these figures

<table>
<thead>
<tr>
<th>Spatial Iterative Scheme (x,y) (z)</th>
<th>Energy Iterative Scheme</th>
<th>Group Balance</th>
<th>x,y Plane Iterations x100</th>
<th>z Plane Iterations</th>
<th>Time (min.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR SLOR</td>
<td>GS</td>
<td>OFF</td>
<td>2200*</td>
<td>5500*</td>
<td>280*</td>
</tr>
<tr>
<td>SLOR SLOR</td>
<td>MINI</td>
<td>OFF</td>
<td>598</td>
<td>1508</td>
<td>93.40</td>
</tr>
<tr>
<td>SLOR SLOR</td>
<td>GS</td>
<td>ON</td>
<td>1647</td>
<td>4298</td>
<td>206.48</td>
</tr>
<tr>
<td>SLOR SLOR</td>
<td>MINI</td>
<td>ON</td>
<td>402</td>
<td>1079</td>
<td>73.73</td>
</tr>
<tr>
<td>MINI MINI</td>
<td>GS</td>
<td>OFF</td>
<td>1550*</td>
<td>3800*</td>
<td>280*</td>
</tr>
<tr>
<td>MINI MINI</td>
<td>MINI</td>
<td>OFF</td>
<td>318</td>
<td>898</td>
<td>82.01</td>
</tr>
<tr>
<td>MINI MINI</td>
<td>GS</td>
<td>ON</td>
<td>1077</td>
<td>2993</td>
<td>225.43</td>
</tr>
<tr>
<td>MINI MINI</td>
<td>MINI</td>
<td>ON</td>
<td>281</td>
<td>803</td>
<td>76.87</td>
</tr>
<tr>
<td>ICCG</td>
<td>GS</td>
<td>OFF</td>
<td></td>
<td>5410*</td>
<td>310*</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>OFF</td>
<td></td>
<td>1364*</td>
<td>108.25</td>
</tr>
<tr>
<td>ICCG</td>
<td>GS</td>
<td>ON</td>
<td></td>
<td>3635</td>
<td>231.05</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>ON</td>
<td></td>
<td>946</td>
<td>87.62</td>
</tr>
<tr>
<td>ICCG SLOR</td>
<td>MINI</td>
<td>ON</td>
<td>323</td>
<td>1463</td>
<td>125.53</td>
</tr>
<tr>
<td>ICCG MINI</td>
<td>MINI</td>
<td>ON</td>
<td>183</td>
<td>805</td>
<td>92.72</td>
</tr>
</tbody>
</table>

TABLE 8.15
RESULTS OF THE 3-D (REFINED y-MESH) 8-GROUP MOATA MODEL
*Convergence not obtained in time available; extrapolation used to obtain these figures

<table>
<thead>
<tr>
<th>Spatial Iterative Scheme (x,y) (z)</th>
<th>Energy Iterative Scheme</th>
<th>Group Balance</th>
<th>x,y Plane Iterations x100</th>
<th>z Plane Iterations</th>
<th>Time (min.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR SLOR</td>
<td>MINI</td>
<td>ON</td>
<td>3000*</td>
<td>8000*</td>
<td>680.0*</td>
</tr>
<tr>
<td>MINI MINI</td>
<td>MINI</td>
<td>ON</td>
<td>304</td>
<td>833</td>
<td>154.4</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>ON</td>
<td>192</td>
<td>1048</td>
<td>167.9</td>
</tr>
<tr>
<td>ICCG MINI</td>
<td>MINI</td>
<td>ON</td>
<td>335</td>
<td>812</td>
<td>171.6</td>
</tr>
<tr>
<td>ICCG SLOR</td>
<td>MINI</td>
<td>ON</td>
<td>335</td>
<td>1484</td>
<td>233.4</td>
</tr>
</tbody>
</table>
It is relevant at this point, however, to compare the use of MINI as a means of accelerating energy convergence with the concept of energy rebalance. The particular energy rebalance scheme employed in this instance was outlined in Section 6.6, namely the reactor is 'collapsed' to a series of G points in energy by a disjunctive partitioning of energy-space. The system of equations is then solved by a direct technique and the group fluxes are rescaled by a multiplicative constant. Like MINI, this has the effect of strengthening the coupling between energy groups.

In some instances convergence was not obtained within prescribed time limits, and extrapolation is applied to estimate the results. Single precision arithmetic for ICCG proved inadequate to complete the calculation to the required level of precision for the most energetic group of neutrons. The convergence criteria was relaxed for all iterative schemes on this problem to enable a fair comparison. This was done, rather than convert ICCG to double precision, because the larger working arrays would alter the data structures in POW3D, requiring a redesign of the code. Even if POW3D structures were redesigned, the storage requirements of primary memory would be so great as to cause excessive page swapping in a virtual machine, and probably be beyond the capabilities of many fixed memory computers.

The results indicate energy group rebalancing makes a considerable difference when the GS scheme is used for energy iterations, but has less effect when MINI is enabled. The overhead
8.28

for the group rebalance is very small and the default option in POW3D is to have it enabled at all times. With MINI handling the energy groups, the improvement in performance is truly outstanding. For thermal reactors upscatter can be significant, as it is here, where five upscatter passes are performed for each outer iteration.

The eight group Moata is also refined in the y direction and results for some methods are given in Table 8.15. SLOR appears to be an inferior performer (on CPU time) as the reactor model becomes more refined, while ICCG and MINI-MINI are roughly comparable when allowance for language differences is made. It is observed that ICCG always requires more z passes than a MINI driver in the same direction.

Upscattering is unimportant in fast reactor studies, where the model normally has only one thermal neutron group. A typical fast reactor is studied next.

8.5.2 Fast 3D reactor problem

A fast benchmark, sodium cooled breeder reactor (LMFBR) studied by Buckel et al. [1977] is calculated here. Two (x,y,z) geometry versions are considered:
BI of 20x20x19 mesh points, and

B2 of 39x39x37, a half spacing of BI mesh points.

The data are available for four energy groups. The number of inner iterations and machine time for various combinations of the three iterative schemes are recorded in Table 8.16. For the simpler BI geometry there are few differences in CPU time required, but for the more involved B2 geometry SLOR again performs poorly. MINI-MINI and ICCG are somewhat comparable on CPU time, but again MINI driving the z direction iteration process requires fewer passes.

8.6 INFLUENCE OF MINI ON SLOR PARAMETERS

MINI causes the diagonal component of the matrix to be lowered. When MINI is applied at a higher level, e.g. as a means of accelerating group convergence, and/or as the iterative scheme in the z direction, then should SLOR be used to converge the innermost loop, the previous estimate for $\omega_{\text{opt}}$ may be no longer appropriate. The altered diagonal term equation (3.3.1), for energy group $g$ with MINI for the groups, and also in the z direction is

$$ g_{\text{a}ijk}^5 = g_{\text{a}ijk}^5 + g_{\text{a}ijk}^6 \gamma_{\text{z}ijk} - \sum_{g' > g} \sigma_{gg'} \gamma_{\text{g}g'}, $$

where $\gamma_{\text{z}ijk}$ and $\gamma_{\text{g}g'}$, are the $\gamma$ values at the point (ijk) used for z and energy connections respectively. For the first iteration, all $\gamma$'s are
TABLE 8.16
NUMBER OF INNER ITERATIONS TO CONVERGE LMFBR PROBLEM

*Convergence not achieved in time available;
extrapolation used to obtain the figures.

<table>
<thead>
<tr>
<th>Method</th>
<th>Group 1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Total iterations</th>
<th>Total z iterations</th>
<th>Time (min.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(x,y) plane iterations</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(z)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B1 - Coarse Mesh Spacing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>SLOR</td>
<td>3516</td>
<td>3248</td>
<td>2165</td>
<td>2140</td>
<td>11069</td>
<td>226</td>
</tr>
<tr>
<td>SLOR</td>
<td>MINI</td>
<td>2867</td>
<td>2653</td>
<td>2150</td>
<td>2175</td>
<td>9845</td>
<td>194</td>
</tr>
<tr>
<td>MINI</td>
<td>SLOR</td>
<td>5751</td>
<td>3715</td>
<td>2155</td>
<td>1732</td>
<td>13353</td>
<td>333</td>
</tr>
<tr>
<td>MINI</td>
<td>MINI</td>
<td>2676</td>
<td>2466</td>
<td>1735</td>
<td>1744</td>
<td>8621</td>
<td>213</td>
</tr>
<tr>
<td>ICCG</td>
<td>SLOR</td>
<td>2486</td>
<td>2180</td>
<td>1489</td>
<td>1622</td>
<td>7777</td>
<td>267</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>1770</td>
<td>1798</td>
<td>1174</td>
<td>1202</td>
<td>5944</td>
<td>215</td>
</tr>
<tr>
<td>ICCG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>207</td>
<td>20.65</td>
</tr>
<tr>
<td>B2 - Fine Mesh Spacing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>SLOR</td>
<td>10141</td>
<td>37167</td>
<td>4475</td>
<td>5907</td>
<td>57690</td>
<td>744</td>
</tr>
<tr>
<td>SLOR</td>
<td>MINI</td>
<td>9433</td>
<td>7861</td>
<td>6328</td>
<td>6833</td>
<td>30455</td>
<td>366</td>
</tr>
<tr>
<td>MINI</td>
<td>SLOR</td>
<td>9287*</td>
<td>33948*</td>
<td>4877*</td>
<td>4934*</td>
<td>52946*</td>
<td>687*</td>
</tr>
<tr>
<td>MINI</td>
<td>MINI</td>
<td>8439</td>
<td>7035</td>
<td>4273</td>
<td>4793</td>
<td>24540</td>
<td>306</td>
</tr>
<tr>
<td>ICCG</td>
<td>SLOR</td>
<td>9160*</td>
<td>10970*</td>
<td>4960*</td>
<td>8112*</td>
<td>33202</td>
<td>634*</td>
</tr>
<tr>
<td>ICCG</td>
<td>MINI</td>
<td>5009</td>
<td>4464</td>
<td>2811</td>
<td>2969</td>
<td>15253</td>
<td>282</td>
</tr>
<tr>
<td>ICCG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>454</td>
<td>228.1</td>
</tr>
</tbody>
</table>
zero, because a GS iteration is used to commence the pass. Hence the initially determined \((x,y)\) SLOR optimal iteration parameter is adequate for the first plane solution, and corresponds to that obtained should SLOR or GS have driven the outer processes.

To gain some idea of the influence of MINI on SLOR for the \((x,y)\) plane, the eight group Moata calculation (Section 8.5.1) is recalculated. In Table 8.17 averages (over all planes) of the group 6 SLOR parameters \(\omega_6\), are given. These are redetermined every three outer iterations (and to a higher accuracy than normally prescribed). The times recorded are from runs undertaken on the AAEC's previous central computer (an IBM360/65). (The times seemed excessive to warrant repeating the calculations on the IBM 3031 just for comparison.)

In each case, group and region rebalance are enabled. There are four entries in the table:

1. SLOR for the z direction and no group MINI, to show the variability introduced in the determination of \(\omega\), from different starting estimates of the solution,

2. MINI in the z direction only,

3. MINI for both the z direction and energy groups,
TABLE 8.17

$\omega_6$ RE-EVALUATED AT DIFFERENT OUTERS, n

*Convergence not achieved in time available; extrapolation used to obtain these figures.

<table>
<thead>
<tr>
<th>Outer (z) Group</th>
<th>n=0</th>
<th>3</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>15</th>
<th>Time* (min.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR GS</td>
<td></td>
<td>1.215</td>
<td>1.214</td>
<td>1.215</td>
<td>1.214</td>
<td>1.213</td>
<td>1.212</td>
</tr>
<tr>
<td>MINI GS</td>
<td></td>
<td>1.215</td>
<td>1.261</td>
<td>1.256</td>
<td>1.252</td>
<td>1.250</td>
<td>1.248</td>
</tr>
<tr>
<td>MINI MINI</td>
<td></td>
<td>1.215</td>
<td>1.274</td>
<td>1.271</td>
<td>1.271</td>
<td>1.266</td>
<td>1.258</td>
</tr>
<tr>
<td>MINI$^3$</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

TABLE 8.18

NUMBER OF I/O TRANSFERS AND FLOATING POINT OPERATIONS PER $z$
ITERATION FOR THE 3-DIMENSIONAL FORMS OF THE ITERATIVE SCHEMES
AS IMPLEMENTED IN POW3D

$N_x, N_y, N_z$ are the numbers of grid points in each
direction, while I is the number of iterations necessary
to converge the 2-dimensional sub-systems.

<table>
<thead>
<tr>
<th>Type of Operation</th>
<th>SLOR</th>
<th>MINI</th>
<th>ICCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix Block Transfers</td>
<td>$N_z$</td>
<td>$N_z$</td>
<td>$3N_z$</td>
</tr>
<tr>
<td>Vector Block Transfers</td>
<td>$3N_z$</td>
<td>$7N_z$</td>
<td>$20N_z$</td>
</tr>
<tr>
<td>*/ Floating Point Arithmetic</td>
<td>$(3 + 10I)N_xN_y$</td>
<td>$(9 + 12I)N_xN_y$</td>
<td>$37N_xN_y$</td>
</tr>
<tr>
<td>+ - Floating Point Arithmetic</td>
<td>$(5 + 6I)N_xN_y$</td>
<td>$(5 + 7I)N_xN_y$</td>
<td>$32N_xN_y$</td>
</tr>
</tbody>
</table>
(4) MINI for all options as a timing comparison.

To make the comparison fair, the other times are discounted for the \( \omega \) re-evaluation.

In Figure 8.3 the variation of the spectral radius \( \rho(\omega) \) is shown for an 'average' group 6 \((x,y)\) plane SLOR iteration scheme at outer iteration \((n=3)\) for different values of the SLOR extrapolating parameter about the critical points. The three different \((z)\) and group iterative methods, SLOR-GS, MINI-GS, MINI-MINI are compared. From the figure it can be seen that the use of initially determined \( \omega \)'s (equivalent to SLOR, GS) results in a rather large decreased effectiveness of the \((x,y)\) SLOR iteration scheme. Even so, the use of MINI reduces the number of \( z \) and group passes to such an extent that the overall effect is a substantial time saving. Repeated re-determination of the \( \omega \)'s is possible, but hardly seems warranted, because of the additional computational time that would be required. (Frequently as many as 30 iterations are \( \epsilon \) expended before an estimate of \( \omega \) emerges.) The results in Table 8.17 reinforce those from Table 8.11 as to the vast superiority of group MINI.

The affect of MINI on the diagonal term in the coefficient matrix has implications for ICCG as well. In POW3D, because of the structure of the seven point difference scheme, re-computation of the \( LDL^T \) approximate decomposition (unlike \( \omega \) re-estimation) involves little overhead (Theorem 5.1), and is performed each time MINI affects the
Spectral radius of $(x,y)$ SLOR iteration scheme at outer $n=3$

Figure 8.3
8.7 OTHER FACTORS DETERMINING THE CHOICE OF ITERATIVE METHOD

It is not sufficient to measure the cost of an algorithm in terms of raw CPU time alone, or number of iterations required to obtain a converged solution. Various aspects of computer architecture should be taken into account before a particular algorithm is chosen for a given task on an specifically configured computer. With the advent of large scale multiprogramming environments and virtual memory, the choice of algorithm may vary with the time of day, the accounting philosophy (which may or may not charge for I/O activity and secondary storage utilisation in addition to CPU activity) and the priority with which the return of results is required.

To give some idea of the overhead involved for the three-dimensional implementation of SLOR-SLOR, MINI-MINI, and ICCG, the number of floating point operations for an individual plane per z iteration are presented in Table 8.18. It is difficult to compare directly the block forms (SLOR and MINI) with ICCG as the number of plane iterations (I) needs to be known. Provided MINI takes fewer than 2.3 iterations on average and SLOR fewer than 3.4, they appear superior to ICCG as far as floating point multiplication and division operations are concerned. This goal is not as unrealistic as it may first appear. For the eight group ordinary mesh Moata, the average number of MINI iterations per plane is 2.05. The number of plane
iterations is higher at the early outer stages of the eigenvalue problem. Consequently it may be more efficient to use ICCG in the plane where such difficulty is encountered, and to switch to the simpler methods when the intermediate solutions show less variation.

The number of I/O transfer operations for the three methods also are shown in Table 8.18. The matrix block transfers concern the coefficient matrix, and each transfer requires the seven matrix entries for every point in the \((x,y)\) plane. The vector transfers each involve \(N_x N_y\) components corresponding to flux, source, or temporary work areas associated with each plane. It is immediately obvious for each \(z\) pass, that ICCG is penalised heavily compared to both other methods, and SLOR involves the least overhead of the three.

To indicate the cost of the I/O overheads, the timing for the fast three-dimensional reactor is given in Table 8.19. The runs were undertaken in a virtual storage environment where the region requested for the VIRTUL structure of POW3D was equal to the available real storage on the IBM 3031. It was impossible nevertheless to request exclusive use of the machine, so elapsed times have been discounted to allow for the presence of other possible users. The discounting rule for measuring real elapsed time is
Real Elapsed time = Measured Elapsed time x CPU time in frequently invoked routines without I/O .

Elapsed time in same routines

It is assumed the frequently invoked routines give a reliable representation of the distribution of resources amongst multiple users. Fortuitously other users had little impact on the test calculations so discounting was seldom great.

The CPU time spent in arranging I/O transfers for the three methods indicates a considerable penalty for ICCG. The relative cost is not so high as predicted from Table 8.18, because other routines in POW3D also require considerable I/O overhead as well. The I/O elapsed time (column three) shows the time that elapses while POW3D is involved in purely I/O activity. (It is not a measure of total elapsed time a code user waits before his results are returned. Assuming no other computer users, the total elapsed time is obtained by adding the individual times in columns one and three.) The elapsed time shows ICCG as a very poor performer indeed. While the SLOR-SLOR combination picks up some advantage in I/O elapsed time, MINI-MINI emerges the winner when elapsed time and CPU time are combined for the fine mesh problem.

8.8 CONCLUSION

The results suggest the three iterative schemes SLOR, MINI and ICCG are all possible contenders for solving the spatial aspects of the neutron diffusion problem. The characteristics of each scheme are
TABLE 8.19
TIMING CONSIDERATIONS FOR THE LMFBR STUDY
*Convergence was not achieved in the time available; extrapolation was used to obtain these figures

<table>
<thead>
<tr>
<th>Method</th>
<th>Total CPU Time (min)</th>
<th>I/O CPU Time (min)</th>
<th>I/O Elapsed Time</th>
<th>Number I/O Calls x 1000</th>
<th>Total Plane Iterations</th>
<th>Total z Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>B1 - Coarse Mesh Spacing</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>17.8</td>
<td>1.8</td>
<td>16.8</td>
<td>36</td>
<td>11069</td>
<td>226</td>
</tr>
<tr>
<td>SLOR</td>
<td>18.3</td>
<td>2.3</td>
<td>22.4</td>
<td>44</td>
<td>9845</td>
<td>194</td>
</tr>
<tr>
<td>MINI</td>
<td>24.9</td>
<td>2.7</td>
<td>26.5</td>
<td>56</td>
<td>13353</td>
<td>333</td>
</tr>
<tr>
<td>MINI</td>
<td>21.2</td>
<td>3.0</td>
<td>31.3</td>
<td>57</td>
<td>8621</td>
<td>213</td>
</tr>
<tr>
<td>ICCG</td>
<td>24.0</td>
<td>2.1</td>
<td>11.2</td>
<td>44</td>
<td>7777</td>
<td>267</td>
</tr>
<tr>
<td>ICCG</td>
<td>24.0</td>
<td>1.7</td>
<td>12.7</td>
<td>57</td>
<td>5944</td>
<td>215</td>
</tr>
<tr>
<td>ICCG</td>
<td>20.7</td>
<td>5.0</td>
<td>26.5</td>
<td>102</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td><strong>B2 - Fine Mesh Spacing</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>279.4</td>
<td>9.9</td>
<td>73.6</td>
<td>153</td>
<td>57690</td>
<td>744</td>
</tr>
<tr>
<td>SLOR</td>
<td>204.3</td>
<td>12.0</td>
<td>144.7</td>
<td>155</td>
<td>30455</td>
<td>366</td>
</tr>
<tr>
<td>MINI</td>
<td>310</td>
<td>12.7</td>
<td>152.0</td>
<td>194</td>
<td>52946</td>
<td>687</td>
</tr>
<tr>
<td>MINI</td>
<td>189.8</td>
<td>10.8</td>
<td>111.6</td>
<td>149</td>
<td>24540</td>
<td>306</td>
</tr>
<tr>
<td>ICCG</td>
<td>353</td>
<td>8.9</td>
<td>98.9</td>
<td>442</td>
<td>33220</td>
<td>634</td>
</tr>
<tr>
<td>ICCG</td>
<td>212.7</td>
<td>10.5</td>
<td>118.3</td>
<td>57</td>
<td>15253</td>
<td>282</td>
</tr>
<tr>
<td>ICCG</td>
<td>228.1</td>
<td>25.3</td>
<td>296.2</td>
<td>401</td>
<td>454</td>
<td></td>
</tr>
</tbody>
</table>
considerably different and the most appropriate choice probably depends on the problem at hand and the priority with which a solution is required.

ICCG is generally the fastest for two-dimensional problems (CPU time) and for small three-dimensional problems, although the savings are not necessarily significant. Without an assembler written version of ICCG the other schemes seem superior. As the complexity of the problem increases, MINI-MINI and ICCG appear comparable in CPU time, but I/O penalties in data organisation work to favour MINI. The amount of disc storage required for the larger problems is significant, the requirements of ICCG being greater than either SLOR or MINI. The requirements for all methods are sufficiently large to present real difficulties. The working space data sets are best allocated on different units for efficiency of disc head movements, and ICCG again suffers most because it uses more data sets. Single precision arithmetic is adequate for ICCG on most problems although it presented difficulties for one problem tested here. Recourse to double precision intermediate working space in memory and on disc files would probably rule out the method on very large problems for most computing facilities available in this country.

The I/O overhead per iteration is a minimum for SLOR-SLOR, but the SLOR driver in the z direction is the most inefficient scheme in terms of number of iterations, and this counterbalances SLOR-SLOR's advantage.
As an energy group driver, MINI is significantly more efficient than GS when considerable upscatter occurs. Even without significant upscatter a MINI groups driver involves relatively little additional expense and is worth retaining as the default option in the code for all thermal reactor problems.
CHAPTER 9

RESULTS OF COARSE MESH REBALANCING

'Le superflu, chose très nécessaire.'

Voltaire , Le Mondain.

9.1 COMPARISON OF ARITHMETIC OVERHEAD

An evaluation of coarse mesh rebalancing is reported for a number of realistic reactor configurations, and an effort is made to determine the respective worth of the four schemes described in Chapter 6.

The number of floating point arithmetic operations required to establish the coarse mesh system of equations, and rebalance the flux are presented in Table 9.1. They apply to a two-dimensional NxN fine grid which is divided evenly by a KxK coarse grid system. The discussion ignores the cost of the coarse mesh solution process, whether an iterative or direct technique is involved, because the solution constitutes only a very small proportion of the time spent in the CMR process. For the pyramid functions, the components given by (6.4.2), either may be calculated every time interpolation at the point (x,y) is required or they may be pre-computed and stored. Pre-computation requires four components for each grid point and it may present problems in data organisation, because of the size of reactor model usually studied in two-dimensions, and may well prove impracticable for three-dimensional cases. The number of arithmetic
<table>
<thead>
<tr>
<th>Method</th>
<th>Compute Coarse Mesh Matrix</th>
<th>Compute Right Hand Side</th>
<th>Rebalance flux Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>*</td>
<td>+</td>
<td>*</td>
</tr>
<tr>
<td>Disjunctive Partitioning</td>
<td>$5N^2 + 4NK$</td>
<td>$5N^2 + 4NK$</td>
<td>$0$</td>
</tr>
<tr>
<td>Region Balance</td>
<td>$32N^2$</td>
<td>$28N^2$</td>
<td>$0$</td>
</tr>
<tr>
<td>Multiplicative (a)</td>
<td>$24N^2$</td>
<td>$20N^2$</td>
<td>$5N^2$</td>
</tr>
<tr>
<td>Pyramid Region (b)</td>
<td>$20N^2$</td>
<td>$20N^2$</td>
<td>$5N^2$</td>
</tr>
<tr>
<td>Balance</td>
<td>$12N^2$</td>
<td>$12N^2$</td>
<td>$4N^2$</td>
</tr>
<tr>
<td>Additive Pyramid (a)</td>
<td>$32N^2$</td>
<td>$32N^2$</td>
<td>$14N^2$</td>
</tr>
<tr>
<td>Region Balance (b)</td>
<td>$16N^2$</td>
<td>$16N^2$</td>
<td>$6N^2$</td>
</tr>
<tr>
<td>Galerkin (b)</td>
<td>$16N^2$</td>
<td>$16N^2$</td>
<td>$6N^2$</td>
</tr>
</tbody>
</table>
operations with (a) and without (b) rebalance are shown in Table 9.1.

The additive form of pyramid partitioning leads to a system of linear equations (6.4.5) with a strong computational advantage for the classical reactor physics eigenvalue problem, namely, the matrix $A^A$ of (6.4.6) need be computed only once, because the estimated flux vector $\Phi_0$ has been transferred to the right hand side. (For kinetics and other problems, however, it may be necessary to periodically recompute $A^A$ due to changes in the material cross sections.) Consequently, for the eigenvalue problem with additive forms of rebalance, only the overhead in computing the right hand side, and in rebalancing the flux, need be considered.

In Table 9.2 a comparison is given of the total number of floating point operations required. It assumed that all common calculations for repeated use of CMR in the eigenvalue calculation have been carried out independently and are not counted. (Note that for the additive methods, however, a single computation of the reduced matrix terms is inappropriate when MINI is used to accelerate energy convergence).

The best case results are costed next on the basis of the equivalent number of fine mesh iterations that could be performed instead. The results given in Tables 9.3 are derived from Tables 8.12 and 9.2 respectively.
### TABLE 9.2
NUMBER OF FLOATING POINT OPERATIONS FOR THE 'BEST CASE'
TWO-DIMENSIONAL IMPLEMENTATION OF CMR ON A KxK COARSE
GRID IMPOSED OVER A NxN FINE GRID

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of Floating Point Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Disjunctive partitioning region balance</td>
<td>$6N^2$ (approx)</td>
</tr>
<tr>
<td>Multiplicative pyramid partitioning, region balance</td>
<td>$32N^2$</td>
</tr>
<tr>
<td>Additive pyramid partitioning, region balance</td>
<td>$9N^2$</td>
</tr>
<tr>
<td>Additive pyramid partitioning, Galerkin weighting</td>
<td>$10N^2$</td>
</tr>
</tbody>
</table>

### TABLE 9.3
APPROXIMATE COST OF APPLYING CMR IN TERMS OF THE
NUMBER OF FINE MESH ITERATIONS FOR THE THREE ITERATIVE SCHEMES

<table>
<thead>
<tr>
<th>Method</th>
<th>SLOR</th>
<th>MINI</th>
<th>ICCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disjunctive</td>
<td>$3/4$</td>
<td>$2/3$</td>
<td>$1/3$</td>
</tr>
<tr>
<td>Multiplicative pyramid</td>
<td>$3-1/2$</td>
<td>$3$</td>
<td>$1-1/2$</td>
</tr>
<tr>
<td>Additive pyramid</td>
<td>$1-1/5$</td>
<td>$1$</td>
<td>$1/2$</td>
</tr>
<tr>
<td>Additive pyramid Galerkin weighting</td>
<td>$1-1/4$</td>
<td>$1$</td>
<td>$1/2$</td>
</tr>
</tbody>
</table>
As expected the overhead in using the disjunctive form of rebalance is less than all other schemes, requiring significantly less computational work than one fine mesh iteration. The two additive schemes require much the same effort as each other for the best case form, while the multiplicative pyramid form is significantly more expensive than all other methods. As the iterative scheme becomes computationally more involved, the cost of performing CMR drops in terms of the number of fine mesh iterations equivalent to the rebalance operation. When the reduced matrix requires continual recomputation for the additive scheme, the number of arithmetic operations is increased greatly. The cost of the 'best case' additive Galerkin scheme is approximately that of the multiplicative pyramid form, but should the pyramid functions require continual evaluation the cost is almost twice as high (if the left and right hand sides are evaluated separately). It remains to be seen whether CMR is worthwhile in practical situations.

9.2 SOME PRACTICAL TWO-DIMENSIONAL CALCULATIONS

For this purpose a number of two-dimensional reactor models have been tested, and the results are reported in Tables 9.4, 9.5, 9.6, 9.7, 9.8 and 9.9. The details for most of the reactor models are referred to in Chapter 8. Two additional calculations are included however, the first involves a 'search' type calculation on the standard Moata reactor, while the second is a typical calculation associated with a modern research reactor of the swimming-pool design
**TABLE 9.4**  
**COARSE MESH REBALANCE RESULTS FOR 4 GROUP MOATA REACTOR**

<table>
<thead>
<tr>
<th>Group Rebalance</th>
<th>Type of Region Rebalance</th>
<th>Iteration Method</th>
<th>Iteration Count Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SLOR</td>
<td>1</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>MINI</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>96</td>
</tr>
<tr>
<td>OFF</td>
<td>Disjunctive, Region</td>
<td>SLOR</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td>Balance</td>
<td>MINI</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>94</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SLOR</td>
<td>154</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>88</td>
</tr>
<tr>
<td>ON</td>
<td>Disjunctive, Region</td>
<td>SLOR</td>
<td>144</td>
</tr>
<tr>
<td></td>
<td>Balance</td>
<td>MINI</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>86</td>
</tr>
<tr>
<td>ON</td>
<td>Multiplicative Pyramid,</td>
<td>SLOR</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>Region</td>
<td>MINI</td>
<td>76</td>
</tr>
<tr>
<td></td>
<td>Balance</td>
<td>ICCG</td>
<td>45</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Pyramid,</td>
<td>SLOR</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>Region</td>
<td>MINI</td>
<td>125</td>
</tr>
<tr>
<td></td>
<td>Balance</td>
<td>ICCG</td>
<td>90</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Pyramid,</td>
<td>SLOR</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td>Galerkin</td>
<td>MINI</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td>Weighting</td>
<td>ICCG</td>
<td>88</td>
</tr>
<tr>
<td>Group Rebalance</td>
<td>Type of Region Rebalance</td>
<td>Iteration Method</td>
<td>Outer Iterations</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------</td>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>SLOR MINI ICCG</td>
<td>14</td>
</tr>
<tr>
<td>OFF</td>
<td>Disjunctive, Region Balance</td>
<td>SLOR MINI ICCG</td>
<td>25 24 14</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SLOR MINI ICCG</td>
<td>16</td>
</tr>
<tr>
<td>ON</td>
<td>Disjunctive, Region Balance</td>
<td>SLOR MINI ICCG</td>
<td>17 21 14</td>
</tr>
<tr>
<td>ON</td>
<td>Multiplicative Pyramid, Region Balance</td>
<td>SLOR MINI ICCG</td>
<td>14 20 13</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Pyramid, Region Balance</td>
<td>SLOR MINI ICCG</td>
<td>11 16 12</td>
</tr>
<tr>
<td></td>
<td>Additive Pyramid, Galerkin Weighting</td>
<td>SLOR MINI ICCG</td>
<td>12 12 12</td>
</tr>
<tr>
<td>Group Rebalance</td>
<td>Type of Region Rebalance</td>
<td>Iteration Method</td>
<td>Outer Iterations</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------</td>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>SLOR</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>DISJUNCTIVE, REGION BALANCE</td>
<td>SLOR</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>16</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SLOR</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>DISJUNCTIVE, REGION BALANCE</td>
<td>SLOR</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>MULTIPlicative, REGION BALANCE</td>
<td>SLOR</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>Additive, REGION BALANCE</td>
<td>SLOR</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>Additive, WEIGHTING</td>
<td>SLOR</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>19</td>
</tr>
</tbody>
</table>
TABLE 9.7
COARSE MESH REBALANCING RESULTS FOR ADJOINT HIFAR REACTOR

*Difficulty in convergence for the outer was achieved with this problem so the convergence criteria was relaxed

<table>
<thead>
<tr>
<th>Group Rebalance</th>
<th>Type of Region Balance</th>
<th>Iteration Method</th>
<th>Outer Iterations</th>
<th>Iteration Count Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>SLOR</td>
<td>29</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td>Disjunctive, Region Balance</td>
<td>SLOR</td>
<td>&gt;99</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SLOR</td>
<td>17</td>
<td>126</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>ON</td>
<td>Disjunctive, Region Balance</td>
<td>SLOR</td>
<td>14</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>ON</td>
<td>Multiplicative Pyramid, Region Balance</td>
<td>SLOR</td>
<td>12</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Pyramid, Region Balance</td>
<td>SLOR</td>
<td>13</td>
<td>198</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Pyramid, Galerkin Weighting</td>
<td>SLOR</td>
<td>12</td>
<td>166</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td></td>
<td>12</td>
</tr>
</tbody>
</table>

*Difficulty in convergence for the outer was achieved with this problem so the convergence criteria was relaxed.
<table>
<thead>
<tr>
<th>Group Rebalance</th>
<th>Type of Region Rebalance</th>
<th>Iteration Method</th>
<th>Outer Iterations</th>
<th>Iteration Count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SLOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>MINI</td>
<td>33</td>
<td>432 444 390 825</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>34</td>
<td>337 426 390 642</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>33</td>
<td>288 348 324 474</td>
</tr>
<tr>
<td>OFF</td>
<td>Disjunctive, Region</td>
<td>SLOR</td>
<td>35</td>
<td>458 463 339 685</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>34</td>
<td>337 409 348 573</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>32</td>
<td>283 337 268 442</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SLOR</td>
<td>34</td>
<td>428 486 392 720</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>35</td>
<td>337 488 396 633</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>35</td>
<td>436 456 302 596</td>
</tr>
<tr>
<td>ON</td>
<td>Disjunctive, Region</td>
<td>SLOR</td>
<td>35</td>
<td>462 546 365 638</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>35</td>
<td>364 460 371 607</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>35</td>
<td>311 379 303 482</td>
</tr>
<tr>
<td>ON</td>
<td>Multiplicative Pyramid, Region</td>
<td>SLOR</td>
<td>35</td>
<td>436 456 302 596</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>34</td>
<td>305 333 246 431</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>34</td>
<td>259 310 252 392</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Pyramid, Region</td>
<td>SLOR</td>
<td>34</td>
<td>471 550 405 686</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>34</td>
<td>409 396 318 482</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>34</td>
<td>323 322 260 421</td>
</tr>
<tr>
<td>ON</td>
<td>Additive Galerkin Weighting</td>
<td>SLOR</td>
<td>34</td>
<td>431 492 366 644</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>34</td>
<td>357 306 280 430</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>34</td>
<td>279 271 243 399</td>
</tr>
</tbody>
</table>
### TABLE 9.9
COARSE MESH REBALANCING FOR A POOL REACTOR

<table>
<thead>
<tr>
<th>Group Rebalance</th>
<th>Type of Region Rebalance</th>
<th>Iteration Method</th>
<th>Outer Iterations</th>
<th>Iteration Count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SLOR</td>
<td>13</td>
<td>736 540 674 351 460 2761</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>18</td>
<td>484 287 304 230 481 1786</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>12</td>
<td>284 181 212 106 277 1060</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>SLOR</td>
<td>12</td>
<td>763 590 725 275 462 2815</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>14</td>
<td>310 201 220 156 174 1161</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>12</td>
<td>248 102 113 76 141 680</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SLOR</td>
<td>12</td>
<td>687 506 344 277 811 2625</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>20</td>
<td>445 284 316 230 511 1786</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>12</td>
<td>284 181 202 106 277 1050</td>
</tr>
<tr>
<td>ON</td>
<td>Disjunctive, Region</td>
<td>SLOR</td>
<td>12</td>
<td>699 551 391 216 766 2623</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MINI</td>
<td>14</td>
<td>255 181 212 157 272 1077</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>11</td>
<td>226 98 117 70 137 648</td>
</tr>
<tr>
<td>ON</td>
<td>Multiplicative, Pyramid,</td>
<td>SLOR</td>
<td>11</td>
<td>623 478 326 212 770 2409</td>
</tr>
<tr>
<td></td>
<td>Region</td>
<td>MINI</td>
<td>11</td>
<td>216 180 185 136 181 898</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>11</td>
<td>222 114 136 79 144 695</td>
</tr>
<tr>
<td>ON</td>
<td>Additive, Pyramid,</td>
<td>SLOR</td>
<td>11</td>
<td>909 800 825 526 893 3953</td>
</tr>
<tr>
<td></td>
<td>Region</td>
<td>MINI</td>
<td>11</td>
<td>983 611 684 312 261 2851</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>11</td>
<td>321 248 424 229 397 1619</td>
</tr>
<tr>
<td>ON</td>
<td>Additive, Galerkin,</td>
<td>SLOR</td>
<td>12</td>
<td>848 717 699 456 853 3573</td>
</tr>
<tr>
<td></td>
<td>Weighting</td>
<td>MINI</td>
<td>11</td>
<td>820 576 558 282 256 2492</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ICCG</td>
<td>11</td>
<td>294 232 322 206 369 1423</td>
</tr>
</tbody>
</table>
[Robertson, unpublished]. Gaps in the tables indicate that convergence was not obtained within a reasonable time. Where the number of outer iterations showed a variation, they are reported as well. The variation in the number of outer iterations tends to confuse the comparison of the different iterative and coarse mesh systems. The interaction of the five levels of solution (outers, energy, energy rebalance, inners, and inner rebalance) affect the convergence process. The differences often become noticeable as a converged result is approached. It is common for the number of outer iterations required to vary as a result. The comparisons reported here ignore any variation in the number of outer iterations, only the total number of plane inner iterations are considered.

A careful examination of the results reveals that with coarse mesh rebalancing, for the spatial and energy aspects, there is generally a reduction in the number of fine mesh iterations required to obtain a converged solution for all but the case of additive pyramid rebalancing. The latter method actually has harmful effects in many studies (e.g. the HIFAR model) and produces little benefit for the other cases. With Galerkin weighting the additive scheme usually shows an improvement, but this is not the case for all the models discussed here. It certainly fails to work on real reactor designs as well as it does with the 'model' problems of Chapter 7. The results tend to suggest that the four schemes may be ranked
(i) multiplicative pyramid,
(ii) disjunctive,
(iii) additive Galerkin,
(iv) additive
in their ability to reduce the number of inner iterations otherwise used.

Due to the void, the TRIGA reactor is a difficult design to analyse with diffusion theory, and it is not surprising that all the spatial rebalance schemes are worthwhile, if not essential, for this reactor calculation. Further discussion of the results will purposefully exclude this reactor model, and consideration is given only to those where the advantage of CMR is more marginal.

To give some indication of the relative improvement with the three superior coarse mesh schemes, the number of inner iterations, required with CMR in use, is expressed as a fraction of those without CMR for both space and energy rebalancing in Table 9.10. The results indicate that if an improvement results from a particular CMR scheme on any reactor problem the benefit generally appears to apply to all three iteration schemes. The pool reactor is a notable exception, however, with the disjunctive scheme. There is no clear pattern suggesting greater benefit for any particular scheme.

The net saving, expressed in terms of the number of iterations saved, after CMR was applied, is tabulated for the trial problems in Table 9.11. (Negative entries denote a cost through application of CMR). The results suggest that the potentially large savings with the
### TABLE 9.10
FRACTIONAL REDUCTION IN NUMBER OF FINE MESH ITERATIONS WITH CMR

<table>
<thead>
<tr>
<th>Method</th>
<th>Moata</th>
<th>HIFAR</th>
<th>HIFAR</th>
<th>Moata</th>
<th>Pool</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adjoint</td>
<td>Search</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISJUNCTIVE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>0.75</td>
<td>1.14</td>
<td>0.58</td>
<td>0.96</td>
<td>0.95</td>
</tr>
<tr>
<td>MINI</td>
<td>0.86</td>
<td>0.79</td>
<td>0.96</td>
<td>1.00</td>
<td>0.60</td>
</tr>
<tr>
<td>ICCG</td>
<td>0.84</td>
<td>0.78</td>
<td>0.54</td>
<td>1.02</td>
<td>0.61</td>
</tr>
<tr>
<td>MULTIPLICATIVE PYRAMID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>0.65</td>
<td>0.94</td>
<td>0.64</td>
<td>0.85</td>
<td>0.87</td>
</tr>
<tr>
<td>MINI</td>
<td>0.59</td>
<td>0.74</td>
<td>0.73</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>0.52</td>
<td>0.79</td>
<td>0.77</td>
<td>0.85</td>
<td>0.66</td>
</tr>
<tr>
<td>ADDITIVE GALERKIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>0.77</td>
<td>1.52</td>
<td>1.06</td>
<td>0.92</td>
<td>1.29</td>
</tr>
<tr>
<td>MINI</td>
<td>0.76</td>
<td>1.31</td>
<td>0.76</td>
<td>1.39</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>0.75</td>
<td>1.39</td>
<td>1.18</td>
<td>0.83</td>
<td>1.34</td>
</tr>
<tr>
<td>Method</td>
<td>Moata</td>
<td>HIFAR</td>
<td>HIFAR</td>
<td>Moata</td>
<td>Pool</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>Adjoint</td>
<td>Search</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DISJUNCTIVE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>88</td>
<td>-365</td>
<td>303</td>
<td>-103</td>
<td>70</td>
</tr>
<tr>
<td>MINI</td>
<td>7</td>
<td>301</td>
<td>-170</td>
<td>640</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>52</td>
<td>207</td>
<td>250</td>
<td>-123</td>
<td>382</td>
</tr>
<tr>
<td><strong>MULTIPlicative PYRAMID</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>-66</td>
<td>-379</td>
<td>-26</td>
<td>-557</td>
<td>38</td>
</tr>
<tr>
<td>MINI</td>
<td>-64</td>
<td>76</td>
<td>-255</td>
<td>618</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>60</td>
<td>34</td>
<td>-10</td>
<td>-147</td>
<td>230</td>
</tr>
<tr>
<td><strong>ADDITIVE GALERKIN</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>39</td>
<td>-1147</td>
<td>-225</td>
<td>-148</td>
<td>-920</td>
</tr>
<tr>
<td>MINI</td>
<td>21</td>
<td>-740</td>
<td>-23</td>
<td>-796</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td>51</td>
<td>-522</td>
<td>-164</td>
<td>120</td>
<td>-408</td>
</tr>
</tbody>
</table>
Multiplicative pyramid form are essentially lost when overheads are considered, even with optimal implementation of the technique. For lesser implementations there is never a gain. The additive pyramid form with Galerkin weighting shows a gain only for the Moata configuration. The disjunctive form has the least overhead and appears the most practicable of the three to have as the default option in POW3D.

9.3 SOME THREE-DIMENSIONAL PRACTICAL CALCULATIONS

The pyramid forms of partitioning were first implemented for two-dimensional problems. It was decided to test them in this environment and then progress to three-dimensional studies if their performance indicated further application might be worthwhile. This was the same cautious approach adopted with the development of MINI. Because the pyramid forms did not appear to live up to the high expectations held for them when tested on two-dimensional models, only the disjunctive form of rebalance was implemented in the third dimension. The extension of the disjunctive partitioning process to the extra dimension is a trivial mathematical concept, its implementation in a computer code is not quite so easy because of the data organisation required to handle large spatial geometries (Chapter 5). Nevertheless, the disjunctive scheme is far simpler to program than the other forms. The performances of MINI-solved region rebalance with some of the three-dimensional reactor configurations studied in Chapter 8 are given in Tables 9.12, 9.13, 9.14 and 9.15.
**TABLE 9.12**

PERFORMANCE OF MINI SOLVED REGION REBALANCE FOR NORMAL 3D MOATA. A 4 x 4 x 4 COARSE GRID IS IMPOSED ON A 23 x 17 x 17 FINE MESH WITH FOUR ENERGY GROUPS

<table>
<thead>
<tr>
<th>Method of Fine Mesh</th>
<th>Number of (x,y) Plane</th>
<th>Number of (x,y) Plane</th>
<th>Number of (z) Plane</th>
<th>Number of (z) Plane</th>
<th>Average No. of (x,y)</th>
<th>Average No. of (z)</th>
<th>Reduction Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution Without Rebalance</td>
<td>20278</td>
<td>17814</td>
<td>484</td>
<td>443</td>
<td>13.1</td>
<td>5.3</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td>MINI</td>
<td>11994</td>
<td>10130</td>
<td>314</td>
<td>277</td>
<td>7.8</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td>ICCG</td>
<td>493</td>
<td>348</td>
<td></td>
<td></td>
<td>5.4</td>
<td></td>
</tr>
</tbody>
</table>
TABLE 9.13
PERFORMANCE OF MINI SOLVED REGION REBALANCE FOR LMFBR OF CHAPTER 8

B1: A 5 x 5 x 5 coarse grid is imposed on a 20 x 20 x 19 fine mesh with four energy groups

B2: A 6 x 6 x 6 coarse grid is imposed on a 39 x 39 x 37 fine mesh with four energy groups

<table>
<thead>
<tr>
<th>Method of Fine Mesh Solution</th>
<th>Number of (x,y) Plane Iterations Without Rebalance</th>
<th>Number of (x,y) Plane Iterations With Rebalance</th>
<th>Number of (z) Plane Iterations Without Rebalance</th>
<th>Number of (z) Plane Iterations With Rebalance</th>
<th>Average No. of (x,y) Iterations Per Outer</th>
<th>Average No. of (z) Iterations Per Outer</th>
<th>Time Reduction Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>15903</td>
<td>11069</td>
<td>346</td>
<td>226</td>
<td>9.1</td>
<td>3.8</td>
<td>0.78</td>
</tr>
<tr>
<td>MINI</td>
<td>14374</td>
<td>8621</td>
<td>340</td>
<td>213</td>
<td>8.6</td>
<td>3.9</td>
<td>0.71</td>
</tr>
<tr>
<td>ICCG</td>
<td></td>
<td></td>
<td>320</td>
<td>207</td>
<td></td>
<td></td>
<td>0.77</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method of Fine Mesh Solution</th>
<th>Number of (x,y) Plane Iterations Without Rebalance</th>
<th>Number of (x,y) Plane Iterations With Rebalance</th>
<th>Number of (z) Plane Iterations Without Rebalance</th>
<th>Number of (z) Plane Iterations With Rebalance</th>
<th>Average No. of (x,y) Iterations Per Outer</th>
<th>Average No. of (z) Iterations Per Outer</th>
<th>Time Reduction Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOR</td>
<td>45552</td>
<td>57690</td>
<td>580</td>
<td>744</td>
<td>11.8</td>
<td>5.8</td>
<td>1.27*</td>
</tr>
<tr>
<td>MINI</td>
<td>47378</td>
<td>24540</td>
<td>557</td>
<td>306</td>
<td>13.8</td>
<td>6.3</td>
<td>0.63</td>
</tr>
<tr>
<td>ICCG</td>
<td></td>
<td></td>
<td>588</td>
<td>454</td>
<td></td>
<td></td>
<td>0.90**</td>
</tr>
</tbody>
</table>

* The influence of the outer process affects final stages of convergence. Provided the 19th outer is taken as indicating a converged situation the time reduction is worthwhile.

** The influence of the outer process again affects convergence. ICCG with rebalance takes two additional outer iterations.
TABLE 9.14
PERFORMANCE OF MINI SOLVED REGION REBALANCE FOR A
REFINED (y) MESH 3D MOATA

A 4 x 5 x 4 coarse grid is imposed on a 23 x 33 x 17 fine
mesh with four energy groups

<table>
<thead>
<tr>
<th>Method of</th>
<th>Number of (x,y) Plane</th>
<th>Number of (x,y) Plane</th>
<th>Number of (z) Plane</th>
<th>Number of (z) Plane</th>
<th>No. of (x,y)</th>
<th>No of (z)</th>
<th>Reduction Factor</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine Mesh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solution</td>
<td>Iterations Without Rebalance</td>
<td>Iterations With Rebalance</td>
<td>Iterations Without Rebalance</td>
<td>Iterations With Rebalance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLORE</td>
<td>58785</td>
<td>53768</td>
<td>596</td>
<td>561</td>
<td>38.0</td>
<td>6.6</td>
<td>0.97</td>
<td></td>
</tr>
<tr>
<td>MINI</td>
<td>16196</td>
<td>11818</td>
<td>361</td>
<td>296</td>
<td>10.4</td>
<td>4.0</td>
<td>0.87</td>
<td></td>
</tr>
<tr>
<td>ICCG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6.9</td>
</tr>
</tbody>
</table>


TABLE 9.15

PERFORMANCE OF MINI SOLVED REGION REBALANCE FOR AN
ORDINARY SPATIAL MESH 3D MOATA WITH EIGHT ENERGY GROUPS

A 4 x 4 x 4 coarse grid is imposed on a 23 x 17 x 17 fine grid

<table>
<thead>
<tr>
<th>Method of</th>
<th>Number of</th>
<th>Number of</th>
<th>Number of</th>
<th>Number of</th>
<th>Average</th>
<th>Average</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine Mesh (x,y) Plane</td>
<td>Iterations</td>
<td>(x,y) Plane</td>
<td>(z) Plane</td>
<td>(z) Plane</td>
<td>No. of (x,y)</td>
<td>No of (z)</td>
<td>Reduction</td>
</tr>
<tr>
<td>Solution Without Rebalance</td>
<td>Iterations With Rebalance</td>
<td>Without Rebalance</td>
<td>Iterations With Rebalance</td>
<td>Iterations Per Outer</td>
<td>Iterations Per Outer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLOR</td>
<td>48000</td>
<td>40200</td>
<td>1286</td>
<td>1079</td>
<td>7.1</td>
<td>3.3</td>
<td>0.99*</td>
</tr>
<tr>
<td>MINI</td>
<td>30000</td>
<td>28100</td>
<td>843</td>
<td>803</td>
<td>4.9</td>
<td>2.3</td>
<td>1.01</td>
</tr>
<tr>
<td>ICCG</td>
<td>1055</td>
<td>946</td>
<td></td>
<td></td>
<td>2.9</td>
<td></td>
<td>0.99</td>
</tr>
</tbody>
</table>

* SLOR experiences problems due to problems with \( \omega \) estimation.

The performance is sensitive to the starting solution.
The fine mesh iterations are undertaken with the three iterative schemes.

Rather than being forced to converge fully, the reduced region rebalance system of equations is solved to an accuracy proportional to the current error limit required of the outer iteration. This is the same strategy as used for the fine mesh equations. Even though the rebalance matrices are comparatively small a block MINI process is used. This enables repeated use of the code developed for the fine mesh problem. (The fractions used for convergence are 0.007 for the \((x,y)\) plane inners and 0.01 for the between plane inners.)

An indication of the time savings arising from application of CMR is given in Tables (9.12-9.15). The code developed for CMR with disjunctive partitioning is optimal and the comparison consequently is valid. The time reduction factor (TRF) quoted is defined as

\[
TRF = \frac{\text{total CPU time with rebalance}}{\text{total CPU time without rebalance}}.
\]

There is a secondary saving through elapsed time, due to the reduced number of iterations required. This is not reported, but is really only applicable for the three-dimensional studies because it is a reduction in the \((z)\) iterative process that reduces the I/O overheads.
Should the pyramids be implemented in the three-dimensional form and no effective CPU savings occur (as for the two-dimensional situation), it is still possible for elapsed time savings to be significant. This would be more probable as the sophistication of the iterative approach increases. Such is the case for ICCG, because significantly more I/O activity is involved in the fine mesh iteration than in the rebalance process.

9.4 CONCLUSION

(i) The practical results support the initial supposition that a CMR procedure will reduce the number of iterations required for three of the rebalance techniques considered.

(ii) Despite certain apprehension concerning the applicability of CMR methods to the SLOR process (Chapter 7) it appears worthwhile having for practical situations.

(iii) The CMR process is applicable to the newer iterative methods (MINI and ICCG) in reducing the number of inner iterations, the computational effort required for the solution, and the elapsed time while I/O activity occurs.

(iv) In overall benefit only the disjunctive form seems worthwhile using, because of its significantly lower overhead. When it fails to reduce significantly the number of iterations, the overhead penalty is not too severe. Consequently it is a worthwhile default option in a working code.
(v) Even when the spatial rebalance procedure is not highly significant (e.g. 8 group Moata), the energy rebalance option can be important. This is related to the upscatter involved with thermal reactors. As seen in Chapter 8, however, the alternative approach with group MINI reduces the necessity for energy rebalance.

(vi) The disjunctive form is far easier to implement, requires less auxiliary working storage and produces a coarse mesh system suitable for two of the iterative methods.

(vii) The properties of the reduced matrix and the preservation of the original band structure are adequate for the disjunctive scheme to be used with a recursive multigrid system.

(viii) The band structure of the reduced matrix differs from the original when pyramid functions are used. It is more difficult to incorporate such functions into an effective multigrid algorithm.
Better is the end of a thing than its beginning; ...

Ecclesiastes 7:8.

In Chapters 2 and 3 some fundamental computational problems confronting reactor physicists and designers were described. The need to produce fast accurate determinations of neutron flux requires suitable numerical tools. The central mathematical hurdle in this determination is the solution of very large systems of linear equations.

The MINI implicit method (Chapter 4) provides one means of solution that is particularly adequate for the larger three-dimensional models that are now being considered. It appears to have certain advantages over ICCG and SLOR in reducing the computational overheads for the third spatial direction. Because symmetry is not required by MINI it has immediate application in accelerating energy group convergence. While the theoretical justification for MINI is somewhat limited, its performance on practical reactor problems (Chapter 8) indicates its success.
A secondary form of acceleration (CMR) was applied to assist convergence of the linear iterative process. Several such schemes were discussed and properties of the reduced matrices were investigated. In certain instances MINI is applicable for solving these reduced systems (disjunctive partitioning - region balance weighting).

Reasons for the possible success of such rebalancing as discussed by Brandt [ibid.] and others were considered (Chapters 6 and 7). A somewhat theoretical investigation in Chapter 7 revealed that such schemes could be used when MINI and ICCG are employed at the inner level of a nuclear code. Practical studies (Chapter 9) revealed that MINI and ICCG are indeed suitable iterative schemes to combine with several forms of coarse mesh rebalancing.

The fundamental reason for commencing these studies was to consider problems associated with and to develop a working nuclear code. The mathematical details selected for discussion in this thesis constitute only a portion of the nuclear code POW3D. The code was intended as a workhorse code for reactor physics studies at the AAEC's Research Establishment and at present it is being phased into this role. The code has a degree of generality and its application outside neutron diffusion studies has been demonstrated already.
APPENDIX A

TEMPORAL INTEGRATION OF THE TIME DEPENDENT NEUTRON DIFFUSION EQUATION

A sketch is given of the numerical methods used in POW3D to handle the temporal integration of the multigroup diffusion equation (3.1.1) and the precursor concentration (3.1.2). The method is essentially that due to Pollard [1973], designed for the two-dimensional code POW [Pollard 1974], the forerunner of POW3D. The method is based on direct time integration along the lines of Stacey [1969] and aims to include the precursor concentration in a natural way.

Assume that the flux and precursor concentration solutions up to the time $t_{p-1}$ are known, and the solution at time $t_p = (t_{p-1} + \Delta t)$ is sought. Equation (3.1.2) can be written as

$$\frac{\lambda}{\Delta t} [e^{\lambda_{d} t} C_{d}(r,t)] = \beta_{d} e^{\lambda_{d} t} \sum_{J} \phi_{g}(r,t) \phi_{g}(r,t) d = 1, 2, \ldots, D, \quad (A.1)$$

which on integration yields

$$C_{d}(r; p) = C_{d}(r; p-1) e^{-\lambda_{d} \Delta t} +$$

$$\beta_{d} \int_{t_{p-1}}^{t_{p}} e^{-\lambda_{d} (t_{p} - t)} \sum_{J} \phi_{g}(r,t) \phi_{g}(r,t) dt, \quad (A.2)$$

with the notation showing $C_{d}(r,t)$ is discretised now. Across each
time step the flux is assumed to vary linearly,

$$\phi_g(r,t) = \left(\frac{t-t_p}{\delta t}\right) \phi_g(r,t_p-1) + \left(\frac{t-t_p-1}{\delta t}\right) \phi_g(r,t_p) ,$$  \hspace{2cm} (A.3)

and each cross section is assumed to be constant,

\[ \text{i.e. } \sigma(r,t) = \sigma(r_{\tilde{t}_p}) , \]  \hspace{2cm} (A.4)

where

$$\sigma(r_{\tilde{t}_p}) = \int_{t_{p-1}}^{t_p} \sigma(r,t) dt/\delta t .$$  \hspace{2cm} (A.5)

Equation (A.5) is necessary, because within each step, the cross section may otherwise have had a discontinuity (in function value or slope). Normally, however,

$$\tilde{t}_p = \frac{1}{2}(t_{p-1} + t_p) .$$

By making the change in variable

$$t = t_{p-1} + \gamma t$$

equation (A.2) becomes
The functions

\[ F_n(x) = \int_0^1 e^{x(1-\gamma)} \gamma^{n-1} d\gamma, \]

\[ F'_n(x) = \int_0^1 e^{x(1-\gamma)} \gamma^{n-1} (1-\gamma) d\gamma \]

are introduced, and (A.7) becomes

\[ C_d(\xi;\rho) = C_d(\xi;\rho-1) e^{-\lambda_d \delta t} + \beta_d \delta t \sum_{k} \frac{1}{\sigma_f} \sigma_f(\xi;\rho) \]

\[ \int [(-\lambda_d \delta t) \Phi_g(\xi;\rho-1) + F_2(-\lambda_d \delta t) \Phi_g(\xi;\rho)] d\gamma, \quad d=1,2,\ldots,D. \quad (A.8) \]

The multigroup diffusion equation can be integrated in a similar fashion. The terms are basically of the form

\[ I = \int_{t_{p-1}}^{t_p} \sigma g(\xi,t) \Phi_g(\xi,t) dt, \]

and with the assumptions (A.3) and (A.4) the integral becomes
To deal with the integrated precursor terms from (A.1), the precursor equation is integrated directly to give

$$
\int_{t_{p-1}}^{t_{p}} \lambda_d C_d(r,t)dt = \beta_d \frac{\delta t}{2} \sum_k \sigma_{fg}(r,t_p)[\phi_g(r;p-1) + \phi_g(r;p)]
$$

$$
+ C_d(r;p-1) - C_d(r;p) . \quad (A.9)
$$

Substitution of (A.8) for $C_d(r;p)$ in (A.9) and replacement of the integrated precursor term from (A.1) with (A.9) leads to the required time integrated diffusion equation

$$
-\nabla \cdot D_{n,g}(r,t_p)\nabla \phi_g(r;p) + [\sigma_{rg}(r,t_p) + 2/(v_g \delta t)] \phi_g(r;p)
$$

$$
- \sum_g \sigma_{gg'}(r,t_p) \phi_{g'}(r;p) - \chi(2)_g(\delta t) \sum_k \sigma_{fg'}(r,t_p) \phi_{g'}(r;p)
$$

$$
= \nabla \cdot D_{n,g}(r,t_p)\nabla \phi_g(r;p-1) - [\sigma_{rg}(r,t_p) - 2/(v_g \delta t)] \phi_g(r;p-1)
$$

$$
+ \sum_g \sigma_{gg'}(r,t_p) \phi_{g'}(r;p-1) + \chi(1)_g(\delta t) \sum_k \sigma_{fg'}(r,t_p) \phi_{g'}(r;p-1)
$$

$$
+ 2 \sum_d \lambda_d \sigma_d G_0(\lambda_d \delta t) C_d(r;p-1) + S_g(r,t_p) , \quad (A.10)
$$

where the fission spectrum functions are given by
\[ X(1)g(\delta t) = X_{pg}(1-\beta) + \sum_d \chi_d g_d G_1(\lambda_d \delta t), \]
\[ X(2)g(\delta t) = X_{pg}(1-\beta) + \sum_d \chi_d g_d G_2(\lambda_d \delta t), \]

\[ G_0(x) = F_1(-x), \]

\[ G_1(x) = 1 - 2F'_1(-x), \]

and

\[ G_2(x) = 1 - 2F_2(-x). \]

The integral functions \( F_n(x) \) and \( F'_n(x) \) are evaluated by Padé approximations [Pollard 1973].
SPATIAL DISCRETISATION OF THE NEUTRON DIFFUSION EQUATION

B.1 INTRODUCTION

The derivation of the finite difference representation for the three spatial dimensional diffusion equation is given for a particular energy group and follows the basic ideas of Wachspress [1966]. The derivation is for an \((x,y,z)\) rectangular form divided by a grid system that consists of planes parallel to the appropriate axes. It is normal for the reactor design to be oriented so that the structural variation is minimal for the \((z)\) direction. In the system implemented, any internal material boundary must be associated with the resulting grid, such that the grid points do not fall on the boundary but are internal to the material. Spatial integration of the diffusion equation \((A.10)\) is carried out over integration boxes surrounding each grid point. A typical box is shown in Figure B.1, and this box itself is decomposed into eight smaller boxes to permit integration over varying materials. Approximations are then sought for the resulting integrals. The required flux solution \(\phi_g(r,t)\) is calculated at the intersection of the grid lines (the edge flux method) rather than the computationally simpler (centre flux) method.
using grid centres. The edge flux method results in a more accurate approximation for the neutron leakage about a point.

To identify the materials making up the region of integration in Figure B.1, the sub-boxes are numbered as indicated

1(OSAPO$^{+}$S$^{+}$A$^{+}$P$^{+}$), 2(ORDSO$^{+}$R$^{+}$D$^{+}$S$^{+}$), 3(OQCRQ$^{+}$Q$^{+}$C$^{+}$R$^{+}$),
4(OPBQO$^{+}$P$^{+}$B$^{+}$Q$^{+}$), 5(OSAPO$^{-}$S$^{-}$A$^{-}$P$^{-}$), 6(ORDSO$^{-}$R$^{-}$D$^{-}$S$^{-}$),
7(OQCRQ$^{-}$Q$^{-}$C$^{-}$R$^{-}$), 8(OPBQO$^{-}$P$^{-}$B$^{-}$Q$^{-}$),

and the grid dimensions are as shown.

The non-leakage terms such as fission emission ($\mathcal{V}\sigma_f$) are considered first, and then the neutron leakage term ($-\nabla \cdot D_n \nabla$). The following notation is used

$\lambda=1,2,3,4,5,6,7,8$ as an index for each sub-box,

$m_\lambda$ identifies the material for box $\lambda$,

$\sigma(m)$ is the cross section of material $m$ filling a sub-box,

$h_j$ is the grid width as indicated in Figure B.1; $j=1,2,\ldots,6$,

$h = \max(h_1,h_2,\ldots,h_6),

v_\lambda$ is the volume of the $\lambda$th sub-box,

$\phi_{ijk}$ is the flux value at the point $(i,j,k)$.

**B.2 NON-LEAKAGE REACTIONS**

Approximations to integrals of the form
are sought, where for convenience the notation is simplified from that of (A.10). \( \phi(r,t) \) may be expanded by a Taylors series about the grid point \((i,j,k)\)

\[
\phi(r) = \phi_{ijk} + O(h),
\]

and when the first term approximation is used, the integral becomes

\[
I = \phi_{ijk} \int_V \sigma(r) \, dr + O(h^4),
\]

since \( v = h^3 \)

\[
I = \phi_{ijk} \sum_{\lambda} \sigma(m_\lambda) v_\lambda + O(h^4).
\]

The discretised approximation is then

\[
\int_V \sigma(r) \phi(r) \, dr \approx \phi_{ijk} \sum_{\lambda=1}^{8} \sigma(m_\lambda) v_\lambda.
\]

The centre flux method would yield a computationally more efficient approximation of the same order of accuracy. The edge flux system is used because of the more accurate approximation it provides to the leakage term.
B.3 LEAKAGE REACTION

The leakage integral

\[ I = - \int \nabla \cdot \mathbf{D}_n(r) \nabla \phi(r) \, dr \]

is transformed by Green's theorem to

\[ I = - \int_{\text{box surface}} \mathbf{n} \cdot \mathbf{D}_n(r) \phi(r) \, ds \quad \text{(B.3.1)} \]

and suitable approximations are sought which are required to satisfy the internal boundary conditions (3.1.5 and 3.1.6). The discrete approximation for (B.3.1) is given by

\[ I = \sum_{\ell=1}^{8} I_{\ell} \quad \text{(B.3.2)} \]

where the \( I_{\ell} \) denote integral approximations over the eight sub-boxes. Attention is restricted first to boxes not touching the external boundaries. The summation in B.3.2 is broken down into summation over 24 box surfaces

\[ I = \sum_{j=1}^{24} I_{j} \quad \text{,} \]

where the 24 surfaces are ordered APP\(^+\)A\(^+\), DRR\(^+\)D\(^+\), SAA\(^+\)S\(^-\), QBB\(^+\)Q\(^+\), DSS\(^+\)D\(^+\), CQQ\(^+\)C\(^+\), RCC\(^+\)R\(^+\), PBB\(^+\)P\(^+\), S\(^+\)A\(^+\)P\(^+\)O\(^+\), S\(^-\)A\(^-\)P\(^-\)O\(^-\), D\(^+\)S\(^+\)O\(^+\)R\(^+\),
Consider a typical surface $APP^+A^+$ (designated for simplicity 1), the leakage component $D_x(m_1)\frac{\partial \phi}{\partial x}$ may be approximated

$$D_x(m_1)\frac{\partial \phi}{\partial x} = D_x(m_1)\frac{\partial \phi}{\partial x}|_0 + \frac{h_1}{2} \frac{\partial}{\partial x} D_x(m_1)\frac{\partial \phi}{\partial x}|_0$$

$$+ \frac{\partial}{\partial y} D_x(m_1)\frac{\partial \phi}{\partial x}|_0 + \frac{\partial}{\partial z} D_x(m_1)\frac{\partial \phi}{\partial x}|_0 + 0(h^2).$$

Integrating this over surface (1) yields

$$I_1 = -\iint D_x(m_1)\frac{\partial \phi}{\partial x} \, dydz$$

$$= \left[ D_x(m_1)\frac{\partial \phi}{\partial x}|_0 + \frac{h_1}{2} \frac{\partial}{\partial x} D_x(m_1)\frac{\partial \phi}{\partial x}|_0 \right] \frac{h_2 h_3}{4}$$

$$+ \frac{h_2}{16} \frac{h_3}{\partial y} D_x(m_1)\frac{\partial \phi}{\partial x}|_0 + \frac{h_2}{16} \frac{h_3}{\partial z} D_x(m_1)\frac{\partial \phi}{\partial x}|_0 ] + 0(h^4) \quad (B.3.3)$$

and by substituting the expansion

$$D_x(m_1)[\phi_{ijk} - \phi_{i+1jk}] = -[h_1 D_x(m_1)\frac{\partial \phi}{\partial x}|_0 + \frac{h_1^2}{2} \frac{\partial}{\partial x} D_x(m_1)\frac{\partial \phi}{\partial x}|_0 ] + 0(h^3)$$

in (B.3.3), the result
\[ I_1 = \frac{h_2 h_3 D_x(m_1)}{4 h_1} [\phi_{ijk} - \phi_{i+1jk}] - \frac{h_2^2 h_3}{16} \frac{\partial}{\partial y} D_x(m_1) \frac{\partial \phi}{\partial x} \bigg|_0 \]

\[- \frac{h_2^2 h_3}{16} \frac{\partial}{\partial z} D_x(m_1) \frac{\partial \phi}{\partial x} \bigg|_0 + O(h^4) \]

is obtained. On the opposite face RDDR^+ (2) a similar result (B.3.5) emerges

\[ I_2 = \frac{h_2 h_3 D_x(m_2)}{4 h_4} [\phi_{ijk} - \phi_{i-1jk}] + \frac{h_2 h_3}{16} \frac{\partial}{\partial y} D_x(m_2) \frac{\partial \phi}{\partial x} \bigg|_0 \]

\[ + \frac{h_2^2 h_3}{16} \frac{\partial}{\partial z} D_x(m_2) \frac{\partial \phi}{\partial x} \bigg|_0 + O(h^4) \]

Combining (B.3.4) and (B.3.5), while taking care to satisfy the internal boundary conditions (3.1.5 and 3.1.6), produces

\[ I_1 + I_2 = \frac{h_2 h_3}{4} \left\{ \frac{D_x(m_1)}{h_1} [\phi_{ijk} - \phi_{i+1jk}] + \frac{D_x(m_2)}{h_4} [\phi_{ijk} - \phi_{i-1jk}] \right\} + O(h^4) \]

In a similar way the following integral approximations may be obtained,

\[ I_3 + I_4 = \frac{h_1 h_3}{4} \left\{ \frac{D_y(m_1)}{h_2} [\phi_{ijk} - \phi_{ij+1k}] + \frac{D_y(m_4)}{h_5} [\phi_{ijk} - \phi_{ij-1k}] \right\} + O(h^4) \]
Consequently, the \( (h^4) \) approximation for the leakage is
\[ -\int_V \nabla \cdot D_n(r) \nabla \phi(r) \, dr \]

\[ = a_{ijk}^1 \phi_{ijk+1} + a_{ijk}^2 \phi_{ij+1k} + a_{ijk}^3 \phi_{i-1jk} + a_{ijk}^4 \phi_{ij-1k} + a_{ijk}^5 \phi_{i+1kj} + a_{ijk}^6 \phi_{ijk-1}, \]

where

\[ a_{ijk}^1 = \left[ \frac{-h_1 h_2}{4} D_z(m_1) + \frac{h_2 h_4}{4} D_z(m_2) + \frac{h_4 h_5}{4} D_z(m_3) + \frac{h_1 h_6}{4} D_z(m_4) \right] / h_3 , \]

\[ a_{ijk}^2 = \left[ \frac{-h_1 h_3}{4} D_y(m_1) + \frac{h_3 h_4}{4} D_y(m_2) + \frac{h_1 h_6}{4} D_y(m_5) + \frac{h_4 h_6}{4} D_y(m_6) \right] / h_2 , \]

\[ a_{ijk}^3 = \left[ \frac{-h_2 h_3}{4} D_x(m_2) + \frac{h_3 h_5}{4} D_x(m_3) + \frac{h_2 h_6}{4} D_x(m_6) + \frac{h_5 h_6}{4} D_x(m_7) \right] / h_4 , \]

\[ a_{ijk}^4 = \left[ \frac{-h_1 h_3}{4} D_y(m_4) + \frac{h_3 h_4}{4} D_y(m_5) + \frac{h_1 h_6}{4} D_y(m_7) + \frac{h_4 h_6}{4} D_y(m_8) \right] / h_5 , \]

\[ a_{ijk}^5 = \left[ \frac{-h_2 h_3}{4} D_x(m_1) + \frac{h_3 h_5}{4} D_x(m_4) + \frac{h_2 h_6}{4} D_x(m_5) + \frac{h_5 h_6}{4} D_x(m_8) \right] / h_1 , \]

\[ a_{ijk}^6 = \left[ \frac{-h_1 h_2}{4} D_z(m_6) + \frac{h_2 h_4}{4} D_z(m_5) + \frac{h_4 h_5}{4} D_z(m_7) + \frac{h_1 h_6}{4} D_z(m_8) \right] / h_6 , \]

\[ a_{ijk}^7 = \sum_{k=1}^{6} a_{ijk}^k. \]  \hspace{1cm} (B.3.6)

It is necessary to study the behaviour of the approximation at boundaries. Consider the face \( S^+S^-Q^-Q^+ \) is an external boundary, then
\[ D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 = \text{SS}^{-0-0} \]

\[ D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 + y \frac{\partial}{\partial y} D(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 + z \frac{\partial}{\partial z} D(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 + 0(h^2) \]

and upon integration over the external face

\[-\int \int D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 \ dydz = -\left[ \frac{h_2h_6}{4} D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 + \frac{h_2h_6}{16} D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_0 \right] + 0(h^4). \quad (B.3.7)\]

On the opposite face (not on the boundary) the integrated term obtained in a manner similar to (B.3.5) is

\[ \int \int D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_{D-R-RD} \ dydz = \frac{h_2h_6}{4h_4} D_x(m_6)[\phi_{ijk} - \phi_{i-1jk}] \]

\[ + \frac{h_2h_6}{16} D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_{0} + \frac{h_2h_6}{16} D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_{0} + 0(h^4). \quad (B.3.8)\]

Substitution of the approximation

\[ D_x(m_6)[\phi_{ijk} - \phi_{i-1jk}] = [h_4 D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_{0} - \frac{h_2^2}{2} D_x(m_6) \frac{\partial \phi}{\partial x} \bigg|_{0}] + 0(h^3) \]

for the first term on the right in (B.3.7) and combining the result
with (B.3.8) leads to

\[- \int \int D_x (m_6) \frac{\partial \phi}{\partial x} \, dydz + \int \int D_x (m_6) \frac{\partial \phi}{\partial x} \, dydz\]

\[= \frac{h_2 h_6}{4h_4} D_x (m_6) \left[ \phi_{i,j,k} - \phi_{i-1,j,k} \right] - \frac{h_2 h_6}{4} D_x (m_6) \frac{\partial \phi}{\partial x} |_0 + O(h^4) \cdot (B.3.9)\]

The second term in (B.3.9) can be replaced directly through use of either boundary condition (3.1.3) or (3.1.4). Consequently the leakage approximations given by (B.3.6), and modified according to (B.3.10 and B.3.11), are suitable at all grid points. The modifications are

\[a_{ijk}^2 = 0 \quad \text{for } \lambda=3, i=1\]
\[\lambda=5, i=N_x\]
\[\lambda=4, j=1\]
\[\lambda=2, j=n_y\]
\[\lambda=6, k=1\]
\[\lambda=1, k=N_z,\]

for both reflective and extrapolated boundaries;

\[a_{ijk}^7 = - \sum_{\lambda=1}^6 a_{ijk}^2 + \frac{1}{12} \left[ h_2 h_3 + h_3 h_5 + h_2 h_6 + h_5 h_6 \right] \left[ \delta_{i1} / d_i + \delta_{iN_x} / d_{N_x} \right] \]
\[+ \frac{1}{12} \left[ h_1 h_3 + h_4 h_3 + h_1 h_6 + h_4 h_6 \right] \left[ \delta_{j1} / d_j + \delta_{jN_y} / d_{N_y} \right] \]
\[ + \frac{1}{T_2} [h_1 h_2 + h_4 h_5 + h_1 h_5] \left[ \frac{\delta_{k_1}}{d_k} + \frac{\delta_{k_N}}{d_{N_z}} \right], \]

\text{(B.3.11)}

where \( \delta_{nm} = 1 \) for \( n=m \)
\[ = 0 \quad \text{for} \quad n \neq m \]

and \( d'_i \) are the boundary extrapolation distances in transport mean free paths, with \( d'_i = 0.71 \) for a free boundary and \( d'_i = \infty \) for a reflective boundary.
REFERENCE

Ho una casa nell'Honan
con il suo laghetto blu,
tutto cinto di bambu.
E sto qui a dissiparmi la mia vita,
a stillarmi il cervel sui libri sacri...

Puccini, Turandot.


Pollard, J.P. [1974] - AUS module POW - a general purpose 0,1, and 2D, multigroup neutron diffusion code including feedback-free kinetics. AAEC/E269.


