WAPD-TM-395
AEC RESEARCH AND
DEVELOPMENT REPORT

THE UTILIZATION OF THE NEUTRON DIFFUSION PROGRAM PDQ-5

January 1965

CONTRACT AT-11-1-GEN-14

BETTIS ATOMIC POWER LABORATORY
WEST MIFFLIN, PENNSYLVANIA

Operated for the
U.S. ATOMIC ENERGY COMMISSION
by WESTINGHOUSE
ELECTRIC CORPORATION

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

WAPD-TM-395

UC-32: Mathematics and Computers
Special

THE UTILIZATION OF THE NEUTRON DIFFUSION PROGRAM PDQ-5

L. A. Hageman
C. J. Pfeifer

CONTRACT AT-11-1-GEN-14

January, 1965

For sale by the Clearinghouse for Federal Scientific and Technical Information, National Bureau of Standards, U. S. Department of Commerce, Springfield, Virginia.

NOTE

This document is an interim memorandum prepared primarily for internal reference and does not represent a final expression of the opinion of Westinghouse. When this memorandum is distributed externally, it is with the express understanding that Westinghouse makes no representation as to completeness, accuracy, or usability of information contained therein.

BETTIS ATOMIC POWER LABORATORY PITTSBURGH, PENNSYLVANIA

Operated for the U. S. Atomic Energy Commission by
Westinghouse Electric Corporation

SPECIAL EXTERNAL DISTRIBUTION

Argonne National Laboratory, Director of Applied Math Division	2
Brookhaven National Laboratory, J. Chernick	2
Case Institute of Technology, R. S. Varga	1
David Taylor Model Basin, H. Polachek	2
Harvard University, G. Birkhoff	1
Knolls Atomic Power Laboratory, R. Ehrlich	10
Los Alamos Scientific Laboratory, B. Carlson	2
New York University, R. Richtmyer	2
Oak Ridge National Laboratory, A. S. Householder	1
Philco Corporation, Computer Division, TUG Executive Secretary	10
Union Carbide Nuclear Company, K-25, V. E. Anderson	1
University of California Lawrence Radiation Laboratory, S. Fernbach	2
DTIE	3
Director, Dev. Div., PNRO	3

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employe or contractor of the Commission, or employe of such contractor, to the extent that such employe or contractor of the Commission, or employe of such contractor, prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, of his employment with such contractor.

TABLE OF CONTENTS

The PDQ-5 program provides a discrete numerical approximation to the two-dimensional, time-independent neutron diffusion problem. The purpose of this report is to give a general description of the computational methods employed by the program and to define and discuss the significance of the output numbers. Some numerical examples are given.

THE UTILIZATION OF THE NEUTRON DIFFUSION PROGRAM PDQ-5

L. A. Hageman
C. J. Pfeifer

I. INTRODUCTION

PDQ-5 is the latest in a series of programs which provide a discrete numerical approximation to the two-dimensional, time-independent, neutron diffusion problem. The present PDQ-5 program is quite different from the original PDQ-5 version as described in Ref. 10. The Chebyshev and inner-outer iteration strategies have been revised extensively. The convergence criterion and the quantities printed by the program during each outer iteration also have been changed. The purpose of this report is to give a general description of the computational methods employed by the program and to define and discuss the significance of the output numbers.

The material presented in this report is intended as an aid to the user of the PDQ-5 program. Thus, much of the mathematical detail and rigor will be omitted in order that a clear overall picture of what the program is trying to do may be presented. We hope that this report will help the user to use the program more efficiently and enable him to better analyze and evaluate the results obtained by the program.

In Chapter II, statements of the continuous and discrete problems are given. Chapter III is devoted to a general description of the method of solution of the discrete problem. Our aim in Chapter III is not to provide a detailed description of the numerical methods employed by the program but rather to describe what these numerical methods are trying to do. The reader should keep this goal in mind while reading Chapter III and not be overly concerned with each mathematical step. In Chapter IV, the present version of the PDQ-5 program is described. Chapter V is devoted to discussions of (1) the flux guess, (2) the first overtone mode eigenvalue, and (3) the convergence of the inner iterations. Numerical examples are given to illustrate certain points.

Most of what is said in Chapter V is based on experience and not on mathematical rigor. Thus, it is still up to the user to convince himself whether our reasons and our interpretation of results are valid for his particular problem.

The preparation of input and the final output edits are not discussed in this report. For a complete description of the PDQ-5 program, see Ref. 4.

II. THE CONTINUOUS AND DISCRETE PROBLEMS

The neutron diffusion approximation to the transport equation for a reactor model in a rectangular region R can be written as

$$(2.1)\qquad \left\{- \operatorname{div}[D_g(\underline{r})\operatorname{grad}\varphi_g(\underline{r})] + \Sigma_g(\underline{r})\varphi_g(\underline{r}) - \Sigma^r_{g-1}(\underline{r})\varphi_{g-1}(\underline{r}) = \frac{\chi_g}{\Lambda}\sum_{k=1}^{G} \nu\Sigma^f_k(\underline{r})\varphi_k(\underline{r}) \right\}_{g=1}^{g=G} ,$$

where

$\underline{r}$ = the position vector whose set of components denote the x-y or r-z coordinates,

g = the lethargy group index,

$\varphi_g(\underline{r})$ = the neutron flux in the g-th group,

$D_g(\underline{r})$ = the diffusion coefficient and $D_g(\underline{r}) > 0$,

$\Sigma^a_g(\underline{r})$ = the absorption macroscopic cross section,

$\Sigma^r_g(\underline{r})$ = the removal macroscopic cross section from group g to group g+1 with

$\Sigma^r_0(\underline{r}) = \Sigma^r_G(\underline{r}) \equiv 0$,

B^2_g = the geometric buckling,

$\Sigma_g(\underline{r}) = \Sigma^a_g(\underline{r}) + \Sigma^r_g(\underline{r}) + D_g(\underline{r})B^2_g$,

χ_g = the integral of the fission spectrum over the lethargy range represented by group g with $\sum_{g=1}^{G}\chi_g = 1.0$,

$\nu\Sigma^f_g(\underline{r})$ = the fission macroscopic cross section times the average number of neutrons released per fission,

Λ = the eigenvalue.

On the external boundary of R, we have the boundary condition that the fluxes are zero, $\varphi_g = 0$, or that the normal derivative is zero, $\frac{\partial\varphi_g}{\partial n} = 0$.[1]

We assume that the region R may be divided into a finite number of rectangular subregions R_i such that the group coefficients D_g, Σ_g, Σ_g^r, and Σ_g^f are constant and non-negative within each subregion R_i. Moreover, we assume that D_g and χ_1 are strictly positive and that $\Sigma_g^f > 0$ for some g and some subregion R_i. It is also assumed that $\varphi_g(\underline{r})$ and the normal component of $D_g(\underline{r})\text{grad}\varphi_g(\underline{r})$ are continuous across interfaces between subregions.

The time-independent diffusion problem stated above defines an eigenvalue problem and we seek to determine solutions φ_g of (2.1) corresponding to the largest (in modulus) eigenvalue Λ of (2.1).

Habetler and Martino [Ref. 9] have shown that the eigenvalue problem (2.1) has a unique positive dominant eigenvalue Λ_1 (i.e. $\Lambda_1 > |\Lambda_k|$ for all $k \neq 1$). Moreover, the solution $\Phi_1(\underline{r}) \equiv \{\varphi_1^1(\underline{r}), \varphi_2^1(\underline{r}), \ldots, \varphi_G^1(\underline{r})\}$ to (2.1) corresponding to Λ_1 can be taken to be positive everywhere. Thus, the continuous problem is well defined.

For complicated reactor designs, one can only hope to find an approximate solution to this problem by the use of numerical methods. The PDQ programs were written to solve this problem by numerical means.

To obtain the discrete numerical analogue to the continuous problem, the coupled differential equations given by (2.1) are approximated by a coupled system of linear algebraic equations obtained by a finite difference technique. Basically, such a technique consists of imposing a mesh of horizontal and vertical lines on the rectangular region R and then for each mesh point replacing

[1] Rotational symmetry boundary conditions and internal zero derivative conditions may also be imposed. See Ref. 4.

the differential equation by a certain finite difference expression involving the φ_g only at the mesh points of R. If N is the number of mesh points, then for each neutron group one obtains a system of N linear equations. In the PDQ-5 program, the finite difference approximation for (2.1) at a general mesh point P (See Fig. 2.1) can be expressed as[1]

$$(2.2)\quad - a_N\varphi_{g,N} - a_S\varphi_{g,S} - a_E\varphi_{g,E} - a_W\varphi_{g,W} + a_P\varphi_{g,P} - r\varphi_{g-1,P} = \frac{\chi_g}{\lambda}\sum_{k=1}^{G} f_k\varphi_{k,P} ,$$

where for x-y geometry

$$a_N = \frac{D_{g,R_1}h_W + D_{g,R_4}h_E}{2h_N} , \quad a_S = \frac{D_{g,R_2}h_W + D_{g,R_3}h_E}{2h_S}$$

$$a_E = \frac{D_{g,R_4}h_N + D_{g,R_3}h_S}{2h_E} , \quad a_W = \frac{D_{g,R_1}h_N + D_{g,R_2}h_S}{2h_W}$$

$$e = \frac{\Sigma_{g,R_1}h_Nh_W + \Sigma_{g,R_2}h_Sh_W + \Sigma_{g,R_3}h_Eh_S + \Sigma_{g,R_4}h_Eh_N}{4} ,$$

$$r = \frac{\Sigma^r_{g-1,R_1}h_Nh_W + \Sigma^r_{g-1,R_2}h_Sh_W + \Sigma^r_{g-1,R_3}h_Eh_S + \Sigma^r_{g-1,R_4}h_Eh_N}{4} ,$$

$$f_k = \frac{\nu\Sigma^f_{k,R_1}h_Nh_W + \nu\Sigma^f_{k,R_2}h_Sh_W + \nu\Sigma^f_{k,R_3}h_Eh_S + \nu\Sigma^f_{k,R_4}h_Eh_N}{4} ,$$

$$a_p = a_N + a_S + a_E + a_W + e .$$

[1] For a derivation of the difference equations for x-y geometry, see Ref. 14, and for r-z geometry, see Ref. 10.

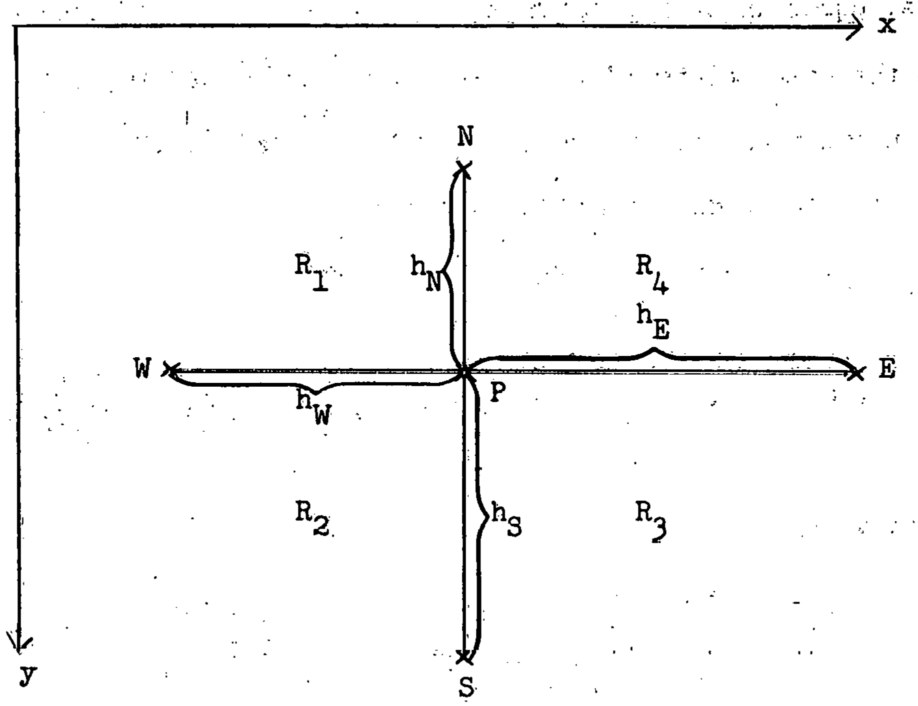


FIGURE 2.1

In matrix notation, the discrete analogue to (2.1) may be written as

$$\left\{ A_g \underline{\varphi}_g - R_{g-1} \underline{\varphi}_{g-1} = \frac{\chi_g}{\lambda} \sum_{k=1}^{G} F_k \underline{\varphi}_k \right\}_{g=1}^{g=G} \tag{2.3}$$

Here $\underline{\varphi}_g$ is a vector whose components are the approximations for $\varphi_g(\underline{r})$ at the N prescribed mesh points and λ is the approximation for Λ. If the finite difference approximation for $\varphi_g(\underline{r})$ at the n-th mesh point is defined as $\varphi_{g,n}$, then

$$\underline{\varphi}_g \equiv \{\varphi_{g,1}, \varphi_{g,2}, \ldots, \varphi_{g,N}\} .$$

A_g, R_{g-1}, and the F_k's are NxN matrices. The matrix A_g corresponds to the discrete analogue of the diffusion and total absorption terms, R_{g-1} corresponds to the removal term from group g-1 to group g, and F_k corresponds to the k-th group fission term.

If we let $\underline{\Phi} \equiv \{\underline{\varphi}_1, \underline{\varphi}_2, \ldots, \underline{\varphi}_G\}$,

$$(2.4) \quad M \equiv \begin{pmatrix} A_1 & & & \bigcirc \\ -R_1 & A_2 & & \\ & \ddots & \ddots & \\ \bigcirc & & -R_{G-1} & A_G \end{pmatrix}, \text{ and } F \equiv \begin{pmatrix} \chi_1 F_1 & \chi_1 F_2 & \cdots & \chi_1 F_G \\ \chi_2 F_1 & \chi_2 F_2 & \cdots & \chi_2 F_G \\ \vdots & & & \vdots \\ \chi_G F_1 & \chi_G F_2 & \cdots & \chi_G F_G \end{pmatrix},$$

then (2.3) may be simply written as

$$(2.5) \qquad \lambda \underline{\Phi} = M^{-1} F \underline{\Phi} .$$

The discrete problem then is to determine the eigenvector corresponding to the largest (in modulus) eigenvalue of (2.5).

Birkhoff and Varga [Ref. 1] have shown that the discrete problem (2.5) has a unique positive dominant eigenvalue λ_1. Moreover, the eigenvector $\underline{\Phi}_1$ corresponding to this fundamental eigenvalue has all positive components. Further, any positive eigenvector of $M^{-1}F$ is simply a scalar multiple of $\underline{\Phi}_1$. Thus, like the continuous problem, the discrete problem is well defined.

The PDQ-5 program obtains by iterative means only an approximate solution to the discrete problem which in turn is only an approximation of the continuous problem. Thus, the solution provided by the program is twice removed from the solution of the continuous problem.[1] We shall use the term discretization error to denote the error introduced in passing from the continuous to the discrete problem and the term iteration error to denote the error introduced in the iterative solution of the discrete problem. The magnitudes of the discretization and iteration errors are a function of how the user specifies his problem.

In order to run a problem on the PDQ-5 program, the user must specify

(1) the continuous problem

(2) the finite difference mesh to be imposed, and

(3) the accuracy desired in the solution of the discrete problem.

Items (1) and (2) together define the discrete problem and determine the discretization error. Item (3) determines the iteration error.

The assumption that the discretization error approaches zero as the separation between mesh points approaches zero is implicit in any finite difference technique. However, the authors know of no rigorous mathematical verification of this assumption for the most general neutron diffusion problem. Of more practical importance is the estimation of the discretization error for a particular discretized problem. Again, this is a very difficult problem for which a satisfactory answer is not known. If a user is concerned about the magnitude of the discretization error, he can solve the problem again using a finer mesh. If the difference between the two solutions is small, then he may feel justified in assuming that the discretization error is small. This procedure is somewhat dangerous but for the present seems to be the most practical.

It should be noted that for a fixed number of mesh points, there is usually a "best" way of placing the mesh points. In general, more mesh points

[1] The continuous problem as defined here is normally only an approximation of the actual physical model. Thus, in reality, the program solution is at least three times removed from the solution of the true physical problem.

should be placed in those areas of the reactor where the flux is changing most radically. For example, that portion of the reflector where the thermal flux peaking occurs should have a higher mesh point density than the rest of the reflector.

For a fixed number of groups the running time of the PDQ-5 program, roughly speaking, is directly proportional to the number of mesh points. Thus, the imposed mesh net affects the program running time as well as the accuracy of the solution. An optimum mesh net might be defined as the mesh net with the smallest number of mesh points[1] for which the discretization error is within the accuracy desired.

In summary, the discretization error exists but, as yet, no satisfactory practical method exists for appraising it. Thus, the user often must resort to an intuitive appraisal based on experience and trial and error. For more general discussions on discretization errors, the interested reader is referred to Refs. 5, 8, and 16.

The <u>iteration error</u> is a much easier quantity to appraise. For the iteration scheme used in the PDQ-5 program, the iteration error approaches zero as the number of iterations approaches infinity. Also, practical numerical methods exist for estimating the iteration error after a finite number of iterations has been performed. After each iteration the program obtains an estimate for the iteration error and makes this information available in the output.

If $\{\lambda(\ell),\boldsymbol{\phi}(\ell)\}$ denote the program's approximation to the solution $\{\lambda_1,\boldsymbol{\phi}_1\}$ of the discrete problem (2.5), then the user is primarily interested in the relative errors

[1] Since the interfaces between different material regions must lie on mesh lines, a great number of mesh lines is often needed just to describe accurately the different materials present in the reactor of interest.

(2.6) $$R\lambda E \equiv \left| \frac{\lambda(\ell) - \lambda_1}{\lambda_1} \right|$$

and

(2.7) $$RPE \equiv \max_{j} \left| \frac{[\underline{\Phi}(\ell)]_j - [\underline{\Phi}_1]_j}{[\underline{\Phi}_1]_j} \right| ,$$

where $[\underline{\Phi}(\ell)]_j$ and $[\underline{\Phi}_1]_j$ denote the j-th component of the $\underline{\Phi}(\ell)$ and $\underline{\Phi}_1$ vectors.

The iteration error, as given in this chapter, encompasses both the eigenvalue (2.6) and eigenvector (2.7) errors. However, in the remaining chapters we shall, for the most part, neglect the eigenvalue error and concentrate on the eigenvector error.

Basically, the reason for this is that the eigenvector is more crucial and more evasive than the eigenvalue. Intuitively, this may be seen by considering a general matrix eigenvalue problem $G\underline{x} = \mu\underline{x}$. Given the eigenvalue μ, it is still a difficult task to determine $\underline{x}$. Whereas, given the eigenvector $\underline{x}$, it is easy to calculate μ. Thus, most eigenvalue problems are in reality eigenvector problems.

In the next two chapters we shall describe how the program measures the iteration error and discuss the significance of the information available in the output.

III. THE SOLUTION OF THE DISCRETE PROBLEM

One may iteratively solve the eigenvalue problem (2.5) using the well known power method. Given the arbitrary positive initial vector $\underline{\Phi}(0)$ and eigenvalue $\lambda(0)$, the power method generates successive estimates for the fundamental eigenvector $\underline{\Phi}_1$ and eigenvalue λ_1 by the process

$$(3.1) \qquad \begin{cases} \underline{S}(\ell) = \dfrac{M^{-1}F}{\lambda(\ell-1)}\,\underline{\Phi}(\ell-1) \\ \underline{\Phi}(\ell) = \underline{S}(\ell) \\ \lambda(\ell) = \lambda(\ell-1)\,\dfrac{\underline{e}^T F\underline{\Phi}(\ell)}{\underline{e}^T F\underline{\Phi}(\ell-1)} \end{cases} ,$$

where $\underline{e}$ is the summation vector; i.e. $\underline{e}$ is a vector all of whose components are unity. By $\underline{x}^T$ is meant the transpose of the column vector $\underline{x}$. We remark that the quantity $\underline{e}^T F\underline{\Phi}$ is just the sum of the components of the fission source vector $\underline{\psi}$, which is defined by $\underline{\psi} \equiv \sum_{g=1}^{G} F_g \underline{\varphi}_g$.

In the solution of the multi-group neutron diffusion problem, these iterations (3.1) are called <u>outer iterations</u> and ℓ is called the outer iteration index number.

Since the largest (in modulus) eigenvalue of $M^{-1}F$ is simple and real, the power method is guaranteed to converge, i.e., for an arbitrary positive guess vector $\underline{\Phi}(0)$,

$$\lim_{\ell\to\infty} \lambda(\ell) = \lambda_1 \quad \text{and} \quad \lim_{\ell\to\infty} \underline{\Phi}(\ell) = \underline{\Phi}_1 \quad .$$

The power method is so-called because it involves repeated multiplication by the matrix $M^{-1}F$. For the process (3.1) gives $\Phi(1) = \frac{1}{\lambda(0)} M^{-1}F\Phi(0)$, $\Phi(2) = \frac{1}{\lambda(1)} M^{-1}F\Phi(1) = \frac{1}{\lambda(1)\cdot\lambda(0)} (M^{-1}F)^2\Phi(0)$ and in general for outer iteration ℓ

$$\Phi(\ell) = \frac{(M^{-1}F)^{\ell}}{\lambda(\ell-1)\lambda(\ell-2)\cdots\lambda(0)}\Phi(0) \quad .$$

For the rest of this paper we shall assume that

$$(3.2)\left\{\begin{array}{ll} \text{(a)} & \text{the eigenvalues } \{\lambda_i\}_1^{GN} \text{ of } M^{-1}F \text{ are real and non-negative and} \\ & \text{are ordered such that } \lambda_1 > \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_{GN} \geq 0 \text{ and that} \\ \text{(b)} & \text{the eigenvectors } \{\Phi_i\}_1^{GN} \text{ of } M^{-1}F \text{ form a basis for the associated} \\ & \text{vector space, i.e., for any vector } \underline{x} \text{ of order GN, there exist} \\ & \text{constants } c_i \text{ such that } \underline{x} = \sum_{i=1}^{GN} c_i\Phi_i. \text{ We take } \Phi_i \text{ to be the eigenvector} \\ & \text{associated with } \lambda_i\text{, i.e., } \lambda_i\Phi_i = M^{-1}F\Phi_i \end{array}\right.$$

Here we have taken N to be the number of mesh points in the finite difference mesh and G to be the number of groups. Except for the fact that $\lambda_1 > |\lambda_2|$, there exists no rigorous basis for these assumptions for the general problem. However, numerical experience indicates that the above assumptions are valid for most "physically reasonable" problems. For the sake of simplicity, in what follows we shall also assume that $\lambda_2 > \lambda_3$.

In order to see how quickly $\Phi(\ell)$ approaches Φ_1 in (3.1), let us expand $\Phi(\ell^*)$ in terms of the eigenvectors of $M^{-1}F$

$$(3.3) \qquad \Phi(\ell^*) = \Phi_1 + \sum_{i=2}^{GN} c_i\Phi_i \quad ,$$

where the c_i are scalars[1]. The iteration error vector, $\underline{E}(\ell^*)$, at outer iteration ℓ^* is defined by

$$(3.4) \qquad \underline{E}(\ell^*) \equiv \boldsymbol{\Phi}(\ell^*) - \boldsymbol{\Phi}_1 = \sum_{i=2}^{GN} c_i \boldsymbol{\Phi}_i \quad .$$

For outer iteration (ℓ^*+1), we have

$$(3.5) \qquad \boldsymbol{\Phi}(\ell^*+1) = \frac{M^{-1}F}{\lambda(\ell^*)} \boldsymbol{\Phi}(\ell^*) = \frac{\lambda_1}{\lambda(\ell^*)} \boldsymbol{\Phi}_1 + \sum_{i=2}^{GN} c_i \left(\frac{\lambda_i}{\lambda(\ell^*)}\right) \boldsymbol{\Phi}_i \quad .$$

If we now assume that ℓ^* is large enough so that the eigenvalue estimates $\lambda(\ell^*+r)$, $r \geq 0$, are sufficiently[2] close to λ_1, then for outer iteration (ℓ^*+r) we can write

$$(3.6) \qquad \boldsymbol{\Phi}(\ell^*+r) = \left(\frac{M^{-1}F}{\lambda_1}\right)^r \boldsymbol{\Phi}(\ell^*) = \boldsymbol{\Phi}_1 + \sum_{i=2}^{GN} c_i \left(\frac{\lambda_i}{\lambda_1}\right)^r \boldsymbol{\Phi}_i$$

and

$$(3.7) \qquad \underline{E}(\ell^*+r) = \sum_{i=2}^{GN} c_i \left(\frac{\lambda_i}{\lambda_1}\right)^r \boldsymbol{\Phi}_i \quad .$$

Since $\frac{\lambda_i}{\lambda_1} < 1$ for $i \geq 2$, we see that $\boldsymbol{\Phi}(\ell^*+r)$ approaches $\boldsymbol{\Phi}_1$ as r approaches infinity. Moreover, the rate at which $\boldsymbol{\Phi}(\ell^*+r)$ approaches $\boldsymbol{\Phi}_1$ or equivalently the rate at which the error vector $\underline{E}$ approaches the null vector depends on how well separated the fundamental eigenvalue λ_1 is from the other eigenvalues of $M^{-1}F$. If the dominance ratio $\bar{\sigma}$ of the matrix $M^{-1}F$ is defined by

[1] For a discussion on the significance of the c_i scalars, see the first few pages of Chapter V.

[2] Numerical experience indicates that the eigenvalue estimates $\lambda(\ell)$ do tend to converge faster than the eigenvector estimates $\boldsymbol{\Phi}(\ell)$.

(3.8) $$\bar{\sigma} \equiv \max_{i\neq 1} \frac{|\lambda_i|}{\lambda_1} = \frac{\lambda_2}{\lambda_1} \quad ,$$

then the most slowly decaying contribution to the error vector, the $\underline{\Phi}_2$ contribution, is multiplied by a factor equal to $\bar{\sigma}$ each outer iteration. In other words, for the iterative process (3.1), the rate at which $\underline{\Phi}(\ell)$ converges to $\underline{\Phi}_1$ generally is governed by the dominance ratio $\bar{\sigma}$ of the matrix $M^{-1}F$.

We now want to consider what practical criterion may be used to terminate the iterative process. If $\underline{e}_j$ is a vector of order GN whose j-th component is unity and all other components zero, then the relative point error RPE(ℓ) for outer iteration ℓ is defined to be

(3.9) $$\text{RPE}(\ell) \equiv \max_j \left| \frac{\underline{e}_j^T \underline{E}(\ell)}{\underline{e}_j^T \underline{\Phi}_1} \right| = \max_j \left| \frac{\underline{e}_j^T (\underline{\Phi}(\ell) - \underline{\Phi}_1)}{\underline{e}_j^T \underline{\Phi}_1} \right| \quad .$$

Certainly, the relative point error is a good measure as to how well $\underline{\Phi}(\ell)$ approximates $\underline{\Phi}_1$ but how can one determine RPE(ℓ) without knowing the desired answer $\underline{\Phi}_1$?

In order to obtain a computable approximation for RPE(ℓ), we begin with a definition. Let

(3.10) $$\text{EPS}(\ell) \equiv \frac{\bar{\lambda}(\ell) - \underline{\lambda}(\ell)}{2} \quad ,$$

where

(3.11) $$\bar{\lambda}(\ell) \equiv \max_j \frac{\underline{e}_j^T \underline{S}(\ell)}{\underline{e}_j^T \underline{\Phi}(\ell-1)} \quad \text{and} \quad \underline{\lambda}(\ell) \equiv \min_j \frac{\underline{e}_j^T \underline{S}(\ell)}{\underline{e}_j^T \underline{\Phi}(\ell-1)} \quad :$$

Since we have assumed that $\lambda_2 > \lambda_i$ for $i \geq 3$, we may take r large enough so that Eq. (3.6) may be written as

$$\underline{\Phi}(\ell^{*}+r) = \underline{S}(\ell^{*}+r) \approx \underline{\Phi}_1 + c_2(\bar{\sigma})^r \underline{\Phi}_2 \tag{3.12}$$

and hence since $\underline{e}_j^T \underline{\Phi}_1 > 0$ for all j

$$\frac{\underline{e}_j^T \underline{S}(\ell^{*}+r+1)}{\underline{e}_j^T \underline{\Phi}(\ell^{*}+r)} \approx 1 - \frac{(1-\bar{\sigma})(\bar{\sigma})^r a_j}{1 + (\bar{\sigma})^r a_j} ,$$

where $a_j = \dfrac{c_2 \underline{e}_j^T \underline{\Phi}_2}{\underline{e}_j^T \underline{\Phi}_1}$. From the definitions of $\bar{\lambda}$ and $\underline{\lambda}$ we can write EPS(ℓ^*+r+1) as

$$\text{EPS}(\ell^{*}+r+1) \approx \frac{(1-\bar{\sigma})(\bar{\sigma})^r}{2} \left[\max_j \frac{a_j}{1 + (\bar{\sigma})^r a_j} - \min_j \frac{a_j}{1 + (\bar{\sigma})^r a_j} \right] . \tag{3.13}$$

Again using (3.12), the relative point error at outer iteration (ℓ^*+r) can be expressed as

$$\text{RPE}(\ell^{*}+r) \approx \max_j \left| \frac{(\bar{\sigma})^r c_2 \underline{e}_j^T \underline{\Phi}_2}{\underline{e}_j^T \underline{\Phi}_1} \right| = (\bar{\sigma})^r \max_j |a_j| . \tag{3.14}$$

Thus, <u>for ℓ sufficiently large</u> we have, after some manipulation, from (3.13) and (3.14) that

$$\frac{\text{EPS}(\ell+1)}{1 - \bar{\sigma} + \text{EPS}(\ell+1)} \lesssim \text{RPE}(\ell) \lesssim \frac{2\text{EPS}(\ell+1)}{1 - \bar{\sigma} - 2\text{EPS}(\ell+1)} . \tag{3.15}$$

Thus, one could terminate the iterative procedure by using EPS(ℓ+1), modified in some way by a function of $\bar{\sigma}$, to measure the relative point error. We note that EPS for outer iteration (ℓ+1) is a measure of the relative point error for outer iteration ℓ.

Another possible measure of the error vector is what we call the <u>relative sum error</u>, RSE(ℓ), which is defined by

$$(3.16) \qquad \mathrm{RSE}(\ell) = \left\{ \frac{(\underline{E}(\ell))^T \underline{E}(\ell)}{\underline{\Phi}_1^T \underline{\Phi}_1} \right\}^{1/2}$$

We shall use

$$(3.17) \qquad \mathrm{BOUND}(\ell) \equiv \left\{ \frac{[\underline{S}(\ell) - \underline{\Phi}(\ell-1)]^T [\underline{S}(\ell) - \underline{\Phi}(\ell-1)]}{[\underline{S}(\ell)^T \underline{\Phi}(\ell-1)]} \right\}^{1/2}$$

as a computable approximation for RSE(ℓ). Similar to (3.15), we have for ℓ sufficiently large

$$(3.18) \qquad \frac{\mathrm{BOUND}(\ell+1)}{1 - \overline{\sigma} + \mathrm{BOUND}(\ell+1)} \lesssim \mathrm{RSE}(\ell) \lesssim \frac{\mathrm{BOUND}(\ell+1)}{1 - \overline{\sigma} - \mathrm{BOUND}(\ell+1)}$$

We note that the relative sum error is an aggregate measure of the error vector $\underline{E}(\ell)$ while the relative point error is a pointwise measure of $\underline{E}(\ell)$. The PDQ-5 program uses the relative point error approximation, EPS, to terminate the outer iterations. The relative sum error approximation, BOUND, is available from the output merely as additional information.

The inequality (3.15) is based on the assumption that ℓ is large enough so that

1. the eigenvalue estimates $\lambda(\ell)$ are sufficiently close to λ_1 and
2. the eigenvector expansion of the error vector consists of one predominate eigenvector, i.e., only the most slowly decaying contribution to the error vector has not been damped out sufficiently.

The two conditions given above are necessary in order to make (3.15) mathematically rigorous. It is felt, however, that the bounds for the relative point error given by (3.15) are practical under much less stringent conditions. In using (3.15) or (3.18), it is important that one have a good estimate for $\overline{\sigma}$. (This is especially true when $\bar{\sigma}$ is close to unity.) The program provides estimates for $\bar{\sigma}$. More will be said about this later.

As seen from Eq. (3.7), the power method is slowly convergent for those problems for which the dominance ratio is close to unity. In the next section we shall describe the Chebyshev polynomial iterative method which is used in the PDQ-5 program to accelerate the convergence of the basic power method.

A. CHEBYSHEV POLYNOMIALS

Suppose that ℓ^* outer iterations have been done and that $\lambda(\ell^*)$ is a good estimate for λ_1. Then from Eq. (3.6), doing r additional power iterations gives

$$(3.19) \qquad \underline{\Phi}(\ell^*+r) = \left(\frac{M^{-1}F}{\lambda_1}\right)^r \underline{\Phi}(\ell^*) = \underline{\Phi}_1 + \sum_{i=2}^{GN} c_i \left(\frac{\lambda_i}{\lambda_1}\right)^r \underline{\Phi}_i \quad .$$

Thus, these r power iterations result in the multiplication of the most slowly decaying contribution to the error vector by a factor of $(\bar{\sigma})^r$. We note that these r power iterations correspond to applying the matrix operator $\left(\frac{M^{-1}F}{\lambda_1}\right)^r$ to the vector $\underline{\Phi}(\ell^*)$. Now if a r-th degree matrix polynomial[1], $P_r\left(\frac{M^{-1}F}{\lambda_1}\right)$, were used to operate on $\underline{\Phi}(\ell^*)$ we could express $\underline{\Phi}(\ell^*+r)$ as

$$(3.20) \qquad \underline{\Phi}(\ell^*+r) = P_r\left(\frac{M^{-1}F}{\lambda_1}\right)\underline{\Phi}(\ell^*) = P_r(1)\underline{\Phi}_1 + \sum_{i=2}^{GN} c_i P_r\left(\frac{\lambda_i}{\lambda_1}\right)\underline{\Phi}_i \quad .$$

Hence, if we could choose the polynomial P_r such that $P_r(1) = 1$ and $\sum_{i=2}^{GN} c_i P_r\left(\frac{\lambda_i}{\lambda_1}\right)\underline{\Phi}_i = \underline{0}$, then we would have $\underline{\Phi}(\ell^*+r) = \underline{\Phi}_1$. Even if such a polynomial existed, it would be a function of the c_i, $\underline{\Phi}_i$, and λ_i, which generally are not

[1] If $P_r(x) = \sum_{k=0}^{r} b_k x^k$ is a polynomial of degree r in x, then the matrix polynomial $P_r(B)$ in the matrix B is defined as $P_r(B) = \sum_{k=0}^{r} b_k (B)^k$.

known for all i. Therefore, such a special polynomial is out of the question. However, for i $\geq$ 2 we have that $0 \leq \frac{\lambda_i}{\lambda_1} \leq \bar{\sigma}$, where the dominance ratio $\bar{\sigma}$ can be estimated by numerical means by the program. Thus, we can try to choose $P_r(x)$ such that $P_r(1) = 1$ and such that the maximum of $|P_r(x)|$ is minimized over the range $0 \leq x \leq \bar{\sigma}$. Such a polynomial exists [Ref. 7] and is called the Chebyshev polynomial. We remark that the coefficients of the Chebyshev polynomial are functions of the program's estimate for $\bar{\sigma}$. Henceforth, we shall denote this estimate by σ_o and shall take $P_{t,\sigma_o}(x)$ to be the Chebyshev polynomial of degree t in which σ_o is used as the estimate for $\bar{\sigma}$.

The Chebyshev polynomial method gives a marked improvement over the power method in speed of convergence. For example, if $\bar{\sigma} = .9$ and $r = 4$, then from (3.19) we see that for the power method the most slowly decaying contribution to the error vector is multiplied by a factor of $(.9)^4$ or .656; whereas, for the Chebyshev polynomial method, the most slowly decaying contribution is multiplied by a factor of $P_{4,.9}(.9)$ or .145. If .8 were used as the estimate for $\bar{\sigma}$, then $P_{4,.8}(.9) = .34$. Thus, the efficiency of the Chebyshev method of iteration depends on the "goodness" of the estimate for $\bar{\sigma}$.

The graph of $P_4\left(\frac{\lambda}{\lambda_1}\right)$ with $\sigma_o = .9$ is given in Fig. 3.1.

To give a practical illustration of the effectiveness of the Chebyshev polynomial method, we cite a problem which was solved by the PDQ-5 program two ways. The regular PDQ-5 program, which uses the Chebyshev method of iteration, required 28 outer iterations to converge the problem; whereas, doing only power iterations, the program required 110 outer iterations. The dominance ratio for this problem was about .975.

GRAPH OF $P_4\left(\frac{\lambda}{\lambda_1}\right)$ WITH $\bar{\sigma} = 0.9$

AND GRAPH OF $\left(\frac{\lambda}{\lambda_1}\right)^4$

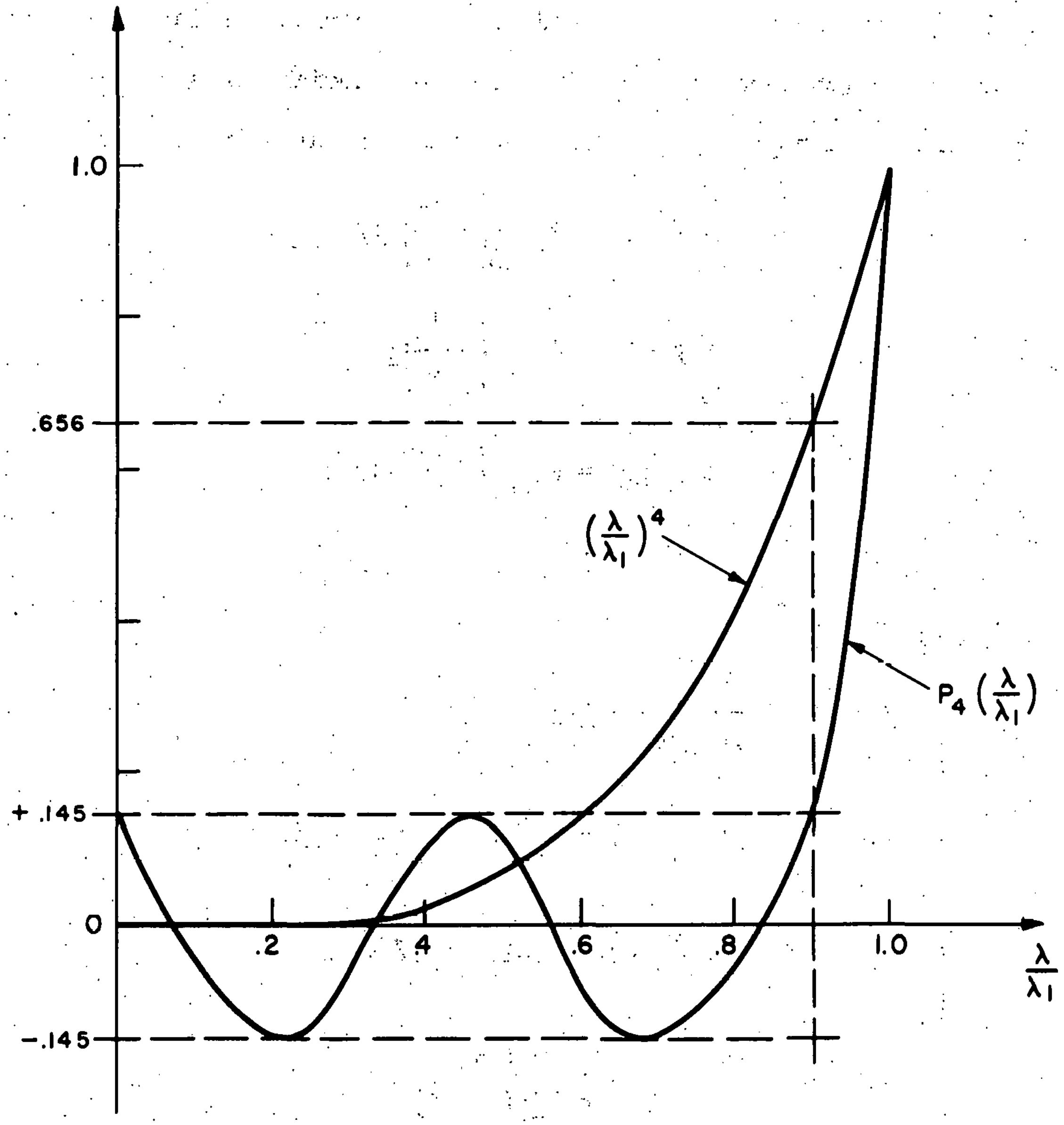


FIGURE 3.1

The Chebyshev polynomials satisfy a three-term recurrence relation of the form

$$(3.21) \qquad P_{t+1,\sigma_o}(x) = (q_1 x - q_2)P_{t,\sigma_o}(x) - q_3 P_{t-1,\sigma_o}(x) \ ,$$

where the q's are functions of σ_o and t. The recurrence relation (3.21) enables us to successively "generate" the Chebyshev polynomials in a straightforward way[1]. Starting with $\underline{\Phi}(\ell^*)$, the PDQ-5 program generates in succession

$$\underline{\Phi}(\ell^*+1) = P_{1,\sigma_o}\left(\frac{M^{-1}F}{\lambda_1}\right)\underline{\Phi}(\ell^*)$$

$$\underline{\Phi}(\ell^*+2) = P_{2,\sigma_o}\left(\frac{M^{-1}F}{\lambda_1}\right)\underline{\Phi}(\ell^*)$$

$$\underline{\Phi}(\ell^*+3) = P_{3,\sigma_o}\left(\frac{M^{-1}F}{\lambda_1}\right)\underline{\Phi}(\ell^*)$$

$$\vdots \qquad \vdots \qquad \qquad \vdots$$

using the procedure

$$(3.22) \quad \left\{ \begin{aligned} &\underline{S}(\ell^*+t) - \frac{M^{-1}F}{\lambda(\ell^*+t-1)}\,\underline{\Phi}(\ell^*+t-1) \\ &\underline{\Phi}(\ell^*+t) = \underline{\Phi}(\ell^*+t-1) + \alpha_{\ell^*+t}[\underline{S}(\ell^*+t) - \underline{\Phi}(\ell^*+t-1] \\ &\qquad\qquad + \beta_{\ell^*+t}[\underline{\Phi}(\ell^*+t-1) - \underline{\Phi}(\ell^*+t-2)] \\ &\lambda(\ell^*+t) = \lambda(\ell^*+t-1)\,\frac{\underline{e}^T F\underline{\Phi}(\ell^*+t)}{\underline{e}^T F\underline{\Phi}(\ell^*+t-1)} \end{aligned} \right.$$

[1] See Ref. 10, pp 26-28

for $t = 1,2,3,\cdots$. α_{ℓ^*+t} and β_{ℓ^*+t} are functions of σ_o and t and $\beta_{\ell^*+1} = 0$. (The λ calculation is included in (3.22) to take into account the fact that $\lambda(\ell^*)$ is not exactly equal to λ_1.)

Note that the Chebyshev polynomial procedure (3.22) is similar to that of the power method (3.1); in fact, with $\alpha = 1$ and $\beta = 0$ the iterative procedure (3.22) is the same as (3.1).

As indicated previously the efficiency of the Chebyshev polynomial method of iteration depends on a knowledge of $\bar{\sigma}$. Generally, of course, $\bar{\sigma}$ is not known a priori. In the PDQ-5 program, before starting the Chebyshev method of iteration, four or five power iterations (3.1) are performed in order to obtain an initial estimate for $\bar{\sigma}$. (These initial power iterations also provide a reasonable estimate for λ_1 for use in the Chebyshev iterations.) Further, low degree Chebyshev polynomials are then repeatedly applied so that the estimates for $\bar{\sigma}$ may be continuously updated. After a good estimate for $\bar{\sigma}$ is obtained a high degree polynomial is applied, if needed.

Estimates for $\bar{\sigma}$ may be obtained by observing the decay rate of the error vector. The PDQ-5 program uses the quantity

$$(3.23) \qquad ER(\ell+1) = \left\{ \frac{[\underline{S}(\ell+1) - \underline{\Phi}(\ell)]^T[\underline{S}(\ell+1) - \underline{\Phi}(\ell)]}{[\underline{S}(\ell) - \underline{\Phi}(\ell-1)]^T[\underline{S}(\ell) - \underline{\Phi}(\ell-1)]} \right\}^{1/2}$$

to measure the decay of the error vector each outer iteration.

For the power method of iteration, it is easy to see from Eq. (3.12) that[1]

[1] Of course, we are assuming that the c_2 in the expansion (3.3) of $\underline{\Phi}(\ell^*)$ is not zero. If c_2 were zero but $c_3 \neq 0$, then $ER(\ell)$ would converge to $\left|\frac{\lambda_3}{\lambda_1}\right|$.

(3.24) $$\lim_{\ell \to \infty} \mathrm{ER}(\ell) = \bar{\sigma} \quad .$$

Thus, for power iterations we may use $\mathrm{ER}(\ell)$ directly to estimate $\bar{\sigma}$.

For the Chebyshev method of iteration (3.22), the estimation of $\bar{\sigma}$ becomes more complicated. From Eq. (3.20), the vector $[\underline{S}(\ell^{*}+t+1) - \boldsymbol{\Phi}(\ell^{*}+t)]$ may be written as

(3.25) $$\underline{S}(\ell^{*}+t+1) - \boldsymbol{\Phi}(\ell^{*}+t) = \sum_{i=2}^{GN} \left(\frac{\lambda_i}{\lambda_1} - 1\right) c_i P_{t,\sigma_o}\left(\frac{\lambda_i}{\lambda_1}\right) \boldsymbol{\Phi}_i \quad .$$

If we now assume that $\sum_{i=3}^{GN} \left(\frac{\lambda_i}{\lambda_1} - 1\right) c_i P_{t,\sigma_o}\left(\frac{\lambda_i}{\lambda_1}\right) \boldsymbol{\Phi}_i$ is small[1] relative to $(\bar{\sigma} - 1) c_2 P_{t,\sigma_o}(\bar{\sigma}) \boldsymbol{\Phi}_2$, then we may write (3.25) as

$$\underline{S}(\ell^{*}+t+1) - \boldsymbol{\Phi}(\ell^{*}+t) \approx (\bar{\sigma} - 1) c_2 P_{t,\sigma_o}(\bar{\sigma}) \boldsymbol{\Phi}_2 \quad .$$

Therefore, for $t > 1$, we have

(3.26) $$\mathrm{ER}(\ell^{*}+t+1) \approx \left| \frac{P_{t,\sigma_o}(\bar{\sigma})}{P_{t-1,\sigma_o}(\bar{\sigma})} \right| ,$$

where $P_{o,\sigma_o}(\bar{\sigma}) = 1$.

Now with $(\mathrm{ER})_{t+1} \equiv [\mathrm{ER}(\ell^{*}+t+1)] \cdot [\mathrm{ER}(\ell^{*}+t)] \cdots [\mathrm{ER}(\ell^{*}+2)]$, it follows from (3.26) that

(3.27) $$(\mathrm{ER})_{t+1} \approx \left| P_{t,\sigma_o}(\bar{\sigma}) \right| \quad .$$

[1] More will be said about this assumption in Chapter V.

One may then obtain a new estimate by solving (3.27) for $\bar{\sigma}$. The PDQ-5 program uses the largest positive solution [Ref. 10 pp 55] to (3.27) as the new estimate for $\bar{\sigma}$.

We remark that the Chebyshev method of iteration (3.22) does not change the meaning of EPS and BOUND as given previously.

In order to carry out either the power (3.1) or Chebyshev (3.22) method of iteration, the matrix equation

$$M\underline{S}(\ell) = \frac{F}{\lambda(\ell-1)}\underline{\Phi}(\ell-1) \tag{3.28}$$

must be solved for $\underline{S}(\ell)$. If $\underline{S}(\ell)$ is written in group component form as $\underline{S}(\ell) \equiv \{\underline{s}_1(\ell), \underline{s}_2(\ell), \cdots, \underline{s}_G(\ell)\}$, then from the definition (2.4) of the matrix M it follows that Eq. (3.28) may be solved for $\underline{S}(\ell)$ by solving successively the system of group equations

$$\left\{A_g\underline{s}_g(\ell) = R_{g-1}\underline{s}_{g-1}(\ell) + \frac{\chi_g}{\lambda(\ell-1)}\sum_{k=1}^{G} F_k\underline{\varphi}_k(\ell-1)\right\}_{g=1}^{g=G} . \tag{3.29}$$

Thus, the vector $\underline{S}(\ell)$ can be determined if we can solve matrix equations of the form

$$A_g\underline{s}_g(\ell) = \underline{b}_g(\ell) \tag{3.30}$$

where A_g is a non-singular matrix and $\underline{b}_g(\ell)$ is a known column vector.

Thus far, we have assumed that direct inversions of the A_g are possible. This is true in most one-dimensional programs such as WANDA [Ref. 11]. However, for most two dimensional problems, the direct inversion of A_g is not feasible.

Thus, the solutions $\underline{s}_g(\ell)$ to the group equations (3.29) must be approximated by some iterative process. The iterations used to obtain these approximations are called <u>inner iterations</u>.

B. INNER ITERATION EFFECT

The iteration method used in the PDQ-5 program for the inner iterations will not be described in detail in this report. Instead we seek only to discuss those points which may be of use to the user. In this section we will discuss the effect of the inner iterations on the outer iterations and the eigenvalue problem (2.5). In Chapter V we will discuss (and give examples of) what the user possibly may do to make the inner iterations more efficient.

In the PDQ-5 program a fixed number of inner iterations, m_g, are performed in group g every outer iteration. (This number is determined by the program.) Let $\tilde{\underline{s}}_g(\ell)$ be the approximation for $\underline{s}_g(\ell)$ which is obtained by doing these m_g iterations. Then when inner iterations are performed, instead of solving the group equations (3.29), we are actually obtaining [Ref. 10, pp 75] the solution to the pseudo system of group equations

$$(3.31)\qquad \left\{A_g(I - E_g)^{-1}\tilde{\underline{s}}_g(\ell) = A_g(I - E_g)^{-1}E_g\underline{\varphi}_g(\ell-1) + R_{g-1}\tilde{\underline{s}}_{g-1}(\ell) + \frac{\chi_g}{\lambda(\ell-1)}\sum_{k=1}^{G} F_k\underline{\varphi}_k(\ell-1)\right\}_{g-1}^{g=G} ,$$

where E_g is the error matrix associated with the iteration method used for the inner iterations. As m_g approaches infinity, E_g approaches the null matrix.

Since we are now solving a different system of group equations, it seems likely that we are also solving a different eigenvalue problem. Indeed, when inner iterations are used, we are actually solving the problem

$$(3.32) \qquad \gamma\underline{\Phi} = \left[I - \tilde{M}^{-1}M\left(I - \frac{M^{-1}F}{\lambda}\right)\right]\underline{\Phi} \quad .$$

Here $\tilde{M}$ is a function of the iteration error matrices E_g and $\tilde{M}$ approaches M as the number of inner iterations in each group approaches infinity. We remark that (3.32) is a non-linear problem.

The particular solution we seek is a scalar $\tilde{\lambda}$ and a vector $\tilde{\underline{\Phi}}$ such that

(a) $\tilde{\lambda}$ and $\tilde{\underline{\Phi}}$ satisfy Eq. (3.32) with $\gamma = 1.0$

(b) $\tilde{\underline{\Phi}}$ is a vector whose components are all positive.

From (3.32), condition (a) requires that $\tilde{\lambda}$ be an eigenvalue and $\tilde{\underline{\Phi}}$ an eigenvector of $M^{-1}F$. Since $M^{-1}F$ can have only one linearly independent positive eigenvector, we see that condition (b) requires $\tilde{\lambda}$ to be the largest eigenvalue of $M^{-1}F$ and $\tilde{\underline{\Phi}}$ its corresponding eigenvector. Thus, if $\tilde{\lambda}$ and $\tilde{\underline{\Phi}}$ satisfy conditions (a) and (b), then $\tilde{\lambda} = \lambda_1$ and $\tilde{\underline{\Phi}} = \underline{\Phi}_1$. Therefore, our discrete problem, even though inner iterations are performed, is still well defined.

One effect then of the inner iterations is that the eigenvalue problem we set out to solve is changed. The particular answer to seek, though, is a solution to both problems. Thus if we solve our problem properly, i.e. satisfy conditions (a) and (b) above, inner iterations do not cause us to get the wrong answer.

Instead of (3.1) then, the PDQ-5 program is actually doing

$$(3.33) \qquad \left\{ \begin{aligned} \underline{S}(\ell) &= \left[I - \tilde{M}^{-1}M\left(I - \frac{M^{-1}F}{\lambda(\ell-1)}\right)\right]\underline{\Phi}(\ell-1) \\ \gamma(\ell) &= \frac{[\underline{S}(\ell)]^T\underline{S}(\ell)}{[\underline{S}(\ell)]^T\underline{\Phi}(\ell-1)} \\ \underline{\Phi}(\ell) &= \underline{S}(\ell) \\ \lambda(\ell) &= \lambda(\ell-1)\,\frac{\underline{e}^T F\underline{\Phi}(\ell)}{\underline{e}^T F\underline{\Phi}(\ell-1)} \end{aligned} \right.$$

The $\gamma(\ell)$ calculation in (3.33) is the estimate for the γ in (3.32). This number should converge to unity.

We know a priori that the power method (3.1) performed without inner iterations is a convergent process. The process (3.33) is also convergent provided a sufficient number of inner iterations are performed. It is felt that the PDQ-5 program does more than a sufficient number of inner iterations so that we may assume the process (3.33) to be convergent.

If we now assume that the eigenvalue estimates $\lambda(\ell)$ are sufficiently close to λ_1, then the iterative process (3.33) is simply the power method applied to the matrix $\left[I - \tilde{M}^{-1}M\left(I - \frac{M^{-1}F}{\lambda_1}\right)\right]$. Thus, for the iterative process (3.33), the rate at which the eigenvector estimates $\underline{\Phi}(\ell)$ converge to $\underline{\Phi}_1$ depends primarily on the dominance ratio, $\hat{\sigma}$, of the matrix $\left[I - \tilde{M}^{-1}M\left(I - \frac{M^{-1}F}{\lambda_1}\right)\right]$. We note that the dominance ratio, $\hat{\sigma}$, of the problem when inners are performed is generally different than the dominance ratio, $\bar{\sigma}$, of the problem when no inners are required[1]. Only as the number of inner iterations approaches infinity does $\hat{\sigma}$ approach $\bar{\sigma}$.

Similarly, instead of (3.22), the PDQ-5 program carries out the Chebyshev method of iteration by

$$(3.34)\begin{cases} \underline{S}(\ell^{*}+t) = \left[I - \tilde{M}^{-1}M\left(I - \dfrac{M^{-1}F}{\lambda(\ell^{*}+t-1)}\right)\right]\underline{\Phi}(\ell^{*}+t-1) \\[2ex] \gamma(\ell^{*}+t) = \dfrac{[\underline{S}(\ell^{*}+t)]^{T}\underline{S}(\ell^{*}+t)}{[\underline{S}(\ell^{*}+t)]^{T}\underline{\Phi}(\ell^{*}+t-1)} \\[2ex] \underline{\Phi}(\ell^{*}+t) = \underline{\Phi}(\ell^{*}+t-1) + \alpha_{\ell^{*}+t}[\underline{S}(\ell^{*}+t) - \underline{\Phi}(\ell^{*}+t-1)] + \beta_{\ell^{*}+t}[\underline{\Phi}(\ell^{*}t-1) - \underline{\Phi}(\ell^{*}+t-1)] \\[2ex] \lambda(\ell^{*}+t) = \lambda(\ell^{*}+t-1)\ \dfrac{\underline{e}^{T}F\underline{\Phi}(\ell^{*}+t)}{\underline{e}^{T}F\underline{\Phi}(\ell^{*}+t-1)} \end{cases}$$

[1] Numerical examples illustrating this will be given in Chapter V. For the PDQ-5 program, numerical experience indicates that $\hat{\sigma}$ often is in the range $\bar{\sigma} - .2(1 - \bar{\sigma}) < \hat{\sigma} < \bar{\sigma} + .2(1 - \bar{\sigma})$.

By doing inner iterations, we have changed the eigenvalue matrix and hence we might expect the associated eigenvalue range to change also. Indeed, when inner iterations are performed, the eigenvalue matrix may have negative and even complex eigenvalues so that assumption (a) in (3.2) is no longer valid. This change of eigenvalue range is taken into account[1] in the PDQ-5 program and generally causes no trouble.

The comments given previously concerning the iterative methods (3.1) and (3.22) generally remain valid for their "inner iteration" counterparts (3.33) and (3.34). One needs only to replace $\bar{\sigma}$ with $\hat{\sigma}$. That is, instead of (3.15), (3.18), (3.24) and (3.27), we have

$$(3.35) \qquad \frac{\mathrm{EPS}(\ell+1)}{1 - \hat{\sigma} + \mathrm{EPS}(\ell+1)} \lesssim \mathrm{RPE}(\ell) \lesssim \frac{2\mathrm{EPS}(\ell+1)}{1 - \hat{\sigma} - 2\mathrm{EPS}(\ell+1)} \quad ,$$

$$(3.36) \qquad \frac{\mathrm{BOUND}(\ell+1)}{1 - \hat{\sigma} + \mathrm{BOUND}(\ell+1)} \lesssim \mathrm{RSE}(\ell) \lesssim \frac{\mathrm{BOUND}(\ell+1)}{1 - \hat{\sigma} - \mathrm{BOUND}(\ell+1} \quad ,$$

$$(3.37) \qquad \lim_{\ell \to \infty} \mathrm{ER}(\ell) = \hat{\sigma} \quad ,$$

$$(3.38) \qquad (\mathrm{ER})_{t+1} \approx \left| P_{t,\sigma_0}(\hat{\sigma}) \right| \quad .$$

Summarizing, the user should be aware that inner iterations

1. are being done,
2. cause the dominance ratio, or equivalently the first overtone eigenvalue, to be altered,
3. consume computer time and hence should be made as efficient as possible.[2]

[1] This topic is discussed in more detail in a separate report on the use of Chebyshev polynomials in the numerical solution of the neutron diffusion problem.

[2] As mentioned previously, the efficiency of the inner iterations will be discussed in Chapter V.

In the next chapter we shall define and discuss the information available in the output of the PDQ-5 program.

IV. THE PDQ-5 PROGRAM

The PDQ-5 program solves the few-group, time independent neutron diffusion equation (2.1) in either x-y or r-z geometry and was written for the Philco-2000 digital computer in the FORTRAN language. The method of solution used by the program is essentially that as described in the previous chapter.

The PDQ-5 program described in this report is actually the third version of PDQ-5. The first version [Refs. 2, 10] differs quite radically from the second and third versions. Most of what is said in this chapter is applicable also to the second version but not the first version.

A. BACKGROUND

PDQ-5 is the latest in a series of PDQ programs which solve the few group diffusion equations. The main differences between the various programs are

(a) the computer for which the program was written;

(b) the number of allowable lethargy groups;

(c) the number of allowable mesh points;

(d) the inner iteration method; and

(e) the application of Chebyshev polynomials.

For the various programs, these differences are listed in Table 4.1.

Because of the magnitude of the neutron diffusion program, data flow and storage are a significant problem. For all PDQ programs listed in Table 4.1, magnetic tape is used for auxiliary storage and data is transferred to fast memory as needed. Essentially all data is transferred from tape to fast memory at least once every outer iteration.

PROGRAM	COMPUTER	MAX. NUMBER OF GROUPS	MAX. NUMBER OF MESH PTS.	INNER ITERATION METHOD	CHEBYSHEV POLY APPLIED TO
PDQ-2	IBM-704	4	6,500	POINT S.O.R.	FISSION SOURCE
PDQ-3	IBM-704	4	7,500	1-LINE S.O.R.	FISSION SOURCE
PDQ-4	PHILCO-2000 (Model 211-10)	4	20,000	2-LINE S.O.R.	FISSION SOURCE
PDQ-5 (Version 1)	PHILCO-2000 (Model 211-2)	5	250,000/Groups	CYCLICALLY REDUCED 3-LINE S.O.R.	FISSION SOURCE
PDQ-5 (Version 2)	PHILCO-2000 (Model 212-2)	5	250,000/Groups	1-LINE S.O.R.	GROUP FLUXES
PDQ-5 (Version 3)	PHILCO-2000 (Model 212-2)	5	250,000/Groups	1-LINE CYCLIC CHEBYSHEV	GROUP FLUXES

TABLE 4.1

The IBM-704 is an unbuffered (input/output) computer and the limits on the number of allowable mesh points for the PDQ-2 and PDQ-3 programs were chosen such that the inner iterations for each group could be performed in fast memory after all the required data for that group was transferred from magnetic tape to fast memory.

The Philco-2000 is a buffered (input/output) computer with the multiple channel feature. The different models denoted in Table 4.1 correspond to increasingly faster memory or arithmetic units. The memory plus arithmetic unit for model 212-2 is roughly 4 times faster than that for model 211-10 which in turn is about 4 times faster than the memory plus arithmetic unit for the IBM-704.

For the PDQ-4 and PDQ-5 programs, the data required to perform the inner iterations for each group could not be stored in fast memory at one time since the allowable number of mesh points was increased beyond the fast memory capabilities. Thus, the performance of inner iterations in a group would require the repeated sweeping of the necessary data from tape. If only one iteration were performed per tape sweep, the speed of the memory and arithmetic unit relative to tape speed for the 211-2 and 212-2 models of the Philco computer is such that the program would run effectively at the slower tape speed despite efficient buffering of the tape data. The device employed in PDQ-5 to circumvent this problem is the use of concurrent iterations, i.e., more than one iteration is done per tape sweep. (See Ref. 13) The present PDQ-5 program tries, if possible, to do all the necessary iterations in one tape sweep. (See Ref. 3) The one-line method is used in PDQ-5 rather than a two or three line method since the one-line method allows more iterations to be performed per tape sweep.

PDQ-5 is the first program in the PDQ series in which the Chebyshev polynomials are applied to the group fluxes. This application of Chebyshev polynomials was made practical by the multiple input/output channel feature of the Philco computer and made necessary by the recent design interest in physically large reactors with their high dominance ratios.

Thus, the characteristics of the computer at hand exert, or should exert, a strong influence on the numerical procedures used. In essense, the numerical techniques are chosen to minimize the computer time *required to solve the problems of interest with a particular computer*. Usually it is the inner iteration method which must be tailored to the computer at hand.

B. OUTER ITERATIONS

Except for two minor changes caused by program efficiency, the outer iterations are carried out as described in the previous chapter.

First, three initial power iterations are carried out by the process (3.33), where $\boldsymbol{\phi}(0)$ and $\lambda(0)$ are input quantities supplied by the user. Then, provided certain conditions are satisfied[1], the Chebyshev polynomial procedure (3.34) is started on outer iteration 4. Chebyshev polynomials of *at least degree 3* are then repeatedly generated. The decision whether to terminate the generation of the present Chebyshev polynomial and start the generation of a new polynomial using an improved estimate for $\hat{\sigma}$ is made by comparing the actual decay rate of the error vector with the theoretical decay rate. The theoretical decay rate is the decay rate one would obtain if σ_0 were equal to $\hat{\sigma}$. ER(ℓ^*+t+1) is used to measure the decay rate actually being obtained. The ratio of the actual decay rate to the theoretical decay rate is printed in the output and will be given later.

[1] If these conditions are not satisfied, power iterations are done until they are satisfied.

1. OUTER ITERATION CONVERGENCE CRITERIA

A problem is considered converged at the end of outer iteration ℓ if all of the following criteria are satisfied:

$$\widetilde{EPS}(\ell) \le \varepsilon_1^2 \tag{4.1}$$

$$\widetilde{EPS}(\ell) \le [1 - \mathrm{SIGMA}(\ell)]\varepsilon_1 \tag{4.2}$$

$$|\gamma(\ell) - 1.0| \le \varepsilon_1 \quad , \tag{4.3}$$

where ε_1 is an input quantity. $\gamma(\ell)$ is defined in (3.33) and (3.34). SIGMA(ℓ) is the most recent estimate for $\hat{\sigma}$ and is obtained either from ER(ℓ) directly or from Eq. (3.38). If $\underline{e}_j$ is a vector of order N whose j-th component is unity and all other components zero, then $\widetilde{EPS}(\ell)$ is defined by

$$\widetilde{EPS}(\ell) \equiv \frac{\bar{\lambda}(\ell) - \underline{\lambda}(\ell)}{2} \tag{4.4}$$

where

$$\bar{\lambda}(\ell) \equiv \max_{g} \max_{j} \frac{\underline{e}_j^T \underline{s}_g(\ell)}{\underline{e}_j^T \underline{\varphi}_g(\ell-1)} \quad \text{and} \quad \underline{\lambda}(\ell) \equiv \min_{g} \min_{j} \frac{\underline{e}_j^T \underline{s}_g(\ell)}{\underline{e}_j^T \underline{\varphi}_g(\ell-1)} \tag{4.5}$$

and where the subscript j for each g varies only over the set of indices for which $\underline{e}_j^T F_g \underline{\varphi}_g(\ell-1) \neq 0$. In (4.5), $\underline{s}_g(\ell)$ and $\underline{\varphi}_g(\ell)$ are the g-th group components of the vectors $\underline{S}(\ell)$ and $\underline{\Phi}(\ell)$.

$\widetilde{EPS}(\ell)$ differs from EPS(ℓ) as defined by (3.10) only in that the determination of $\bar{\lambda}$ and $\underline{\lambda}$ for $\widetilde{EPS}$ excludes those flux points which are in nonfissionable material while the determination of $\bar{\lambda}$ and $\underline{\lambda}$ for EPS does not. In most problems where both $\widetilde{EPS}$ and EPS were calculated, it was found that these quantities differed to any extent only in the early iterations. Thus, for most practical purposes, $\widetilde{\text{EPS}}(\ell)$ may be taken to be equivalent to EPS(ℓ).

Normally, it is condition (4.1) which determines the termination of the problem. Only when [1 - SIGMA(ℓ)] becomes smaller than ε_1 does condition (4.2) take over. If $\hat{\partial}$ is close to unity, we see from the inequality (3.35) that EPS(ℓ) being small does not necessarily mean that RPE(ℓ) is small. Condition (4.2) was included to prevent this "pseudo convergence". Of course, for condition (4.2) to be of any help, SIGMA(ℓ) must be a good estimate for $\hat{\partial}$. Condition (4.3) was included for a very special type of problem and usually should not be of concern to the user.

Basically, the convergence criteria for the outer iterations are set up so that the relative point error of the group fluxes is less than two times the input quantity ε_1. This will only be valid, though, if enough iterations are done so that certain assumptions are satisfied or very nearly satisfied. Thus, the user must make ε_1 small enough so that the program has time to do its job.

2. OUTPUT FROM THE OUTER ITERATIONS

The information printed by the program during outer iteration ℓ is given below. We assume that ℓ is written as ℓ^*+t, where $t = 0$ implies that the power method (3.33) is being carried out and $t \geq 1$ implies that a Chebyshev polynomial is being generated by (3.34).

(1) DEGREE $\equiv t$

(2) NORM $\equiv \gamma(\ell^*+t)$ γ is defined in (3.33) and (3.34)

For a convergent problem NORM must be converging to unity.

(3) LAMBDA $\equiv [\gamma(\ell^*+t)][\lambda(\ell^*+t-1)]$ for $\ell^*+t > 1$ and

LAMBDA $\equiv \lambda(\ell^*+t)$ for $\ell^*+t = 1$. λ is defined in (3.33) and (3.34).

We have not given any specific advice concerning the accuracy of the eigenvalue estimate LAMBDA. The user may often obtain a "feel" for the eigenvalue accuracy by examining successive LAMBDA's, i.e., by noting how the LAMBDA's are changing from outer to outer. Normally, the relative eigenvalue error $\left|\frac{\text{LAMBDA} - \lambda_1}{\lambda_1}\right|$ should not be taken to be less than $\left|\text{NORM} - 1.0\right|$.

(4) MAX $\equiv [\lambda(\ell^* + t - 1)][\bar{\lambda}(\ell^* + t)]$ $\qquad \bar{\lambda}$ is defined by (4.5)

(5) MIN $\equiv [\lambda(\ell^* + t - 1)][\underline{\lambda}(\ell^* + t)]$ $\qquad \underline{\lambda}$ is defined by (4.5)

Actually, the program uses the absolute value of $\frac{\underline{e}_j^T \underline{s}_g(\ell)}{\underline{e}_j^T \underline{\varphi}_g(\ell-1)}$ to determine $\bar{\lambda}(\ell)$ and $\underline{\lambda}(\ell)$. But if any component of $\underline{S}(\ell)$ which is in a fissionable material is negative, then a negative MIN is printed. Thus, a negative MIN means some component of $\underline{S}(\ell)$ is negative. A negative component of $\underline{S}(\ell)$ in non-fissionable material, however, is not detected.

If an infinite number of inner iterations were done, then MAX and MIN would be rigorous upper and lower bounds for the eigenvalue λ_1. However, when a finite number of inner iterations are done, MAX and MIN need not always bound λ_1.

(6) EPS $\equiv \widetilde{\text{EPS}}(\ell^* + t)$ $\qquad \widetilde{\text{EPS}}$ is given by (4.4)

(7) PT/AV $\equiv \frac{\widetilde{\text{EPS}}(\ell^* + t)}{\text{BOUND}(\ell^* + t)}$ $\qquad$ BOUND is given by (3.17)

Pt/AV compares the two measures we have for the error vector. It is the ratio of the pointwise measure to the aggregate or "average" measure. As to be expected, this ratio usually is greater than one. If PT/AV is large, say greater than 10 or so, this often implies that a small or insignificant region of the reactor is causing trouble in the pointwise measure $\widetilde{\text{EPS}}$. Frequently this trouble can be corrected by changing the flux guess or the description of the problem. We will give an example of this in Chapter V.

(8) SIGMA $\equiv$ an estimate for the dominance ratio $\hat{\sigma}$ and is obtained by

SIGMA $\equiv$ ER(ℓ^*+t) for power iterations. ER is given by Eq. (3.23).

SIGMA $\equiv$ largest positive solution of Eq. (3.27) for Chebyshev iterations.

(9) SIGMA EST. $\equiv$ another estimate for the dominance ratio $\hat{\sigma}$.

Although SIGMA is easy to compute, it is not easy to compute at the right time, i.e., one would like to use the SIGMA estimate for $\hat{\sigma}$ before it is convenient for the program to compute SIGMA. However, a good approximation for SIGMA, which is the SIGMA EST., is available at the proper time. If t = 1, SIGMA EST is the estimate, σ_o, for $\hat{\sigma}$ which is used in the polynomial generation. The user should take SIGMA as the best estimate for $\hat{\sigma}$.

(10) RATIO $\equiv$ 1.0 if outer iteration (ℓ-1) was a power iteration.

$$\text{RATIO} \equiv \frac{\text{Actual convergence rate of outer iteration } (\ell\text{-}1)}{\text{Theoretical convergence rate of outer iteration } (\ell\text{-}1)}$$

if outer iteration (ℓ-1) was a Chebyshev iteration.

RATIO is used by the program to measure the effectiveness of the Chebyshev polynomial presently being generated. If RATIO < 1.0, then we are not doing as well as expected and if RATIO > 1.0, we are doing better than expected. RATIO probably is of limited interest to the user.

In the next section, we will define the information printed by the program during the inner iterations.

3. OUTPUT FROM THE INNER ITERATIONS

We recall from Chapter 3 that the inner iterations for group g are performed in order to obtain an approximation to the solution of the matrix equation

$$A_g \underline{s}_g(\ell) = \underline{b}_g(\ell) ,$$

where A is a non-singular matrix and $\underline{b}_g(\ell)$ is a known column vector.

Basically, these inner iterations generate this approximation to the true solution by repeated application of a matrix algorithm which successively improves an initial guess for the solution. The matrix algorithm depends on the iterative method used for the inner iterations. The PDQ-5 program uses the one-line cyclic Chebyshev semi-iterative method [Ref. 16, pp. 149] for the inner iterations. This method can be viewed as a variant of the one-line successive overrelaxation method which had been used in the previous version of the PDQ-5 program.

The PDQ-5 program does a fixed number, m_g, of inner iterations in group g every outer iteration. The program tries to choose m_g such that the inner iteration error vector after performing the m_g iterations is about .1 of the initial error vector. Essentially, this prediction is made in what is called the "omega routine", which is carried out before the first outer iteration is done. We will not describe how this prediction is made.

If there is not sufficient storage to perform all m_g iterations concurrently, the smallest odd integer q is found such that the data for (m_g/q) concurrent iterations will fit into storage. Then (m_g/q) concurrent iterations are performed during each of q passes through the mesh. The mesh is swept forward on the odd passes and backward on the even passes or vice versa. On the backward sweep some of the data tapes are read backwards.

If $\underline{s}_g^{(m)}(\ell)$ denotes the approximation for $\underline{s}_g(\ell)$ after m iterations in group g, then the flux residual vector $\underline{R}_g^{(m)}(\ell)$ for the m-th inner iteration is defined to be

$$(4.6) \qquad \underline{R}_g^{(m)}(\ell) \equiv \underline{s}_g^{(m)}(\ell) - \underline{s}_g^{(m-1)}(\ell) \ .$$

The result of the last outer iteration, $\varphi_g(\ell-1)$, is used as the initial guess vector, i.e., $\underline{s}_g^{(o)}(\ell) = \underline{\varphi}_g(\ell-1)$.

We are now ready to define the output from the inner iterations. During outer iteration ℓ, the output from the inner iterations for each group g is

$$(1)\quad R(1) \equiv \left\{\left[\underline{R}_g^{(1)}(\ell)\right]^T\left[\underline{R}_g^{(1)}(\ell)\right]\right\}^{1/2}$$

R(1), which is called the initial residual, should be decreasing from outer to outer. The SIGMA EST estimate for $\hat{\sigma}$ is determined from the initial residuals.

$$(2)\quad \mathrm{DELTA}(q,m_g) \equiv \left\{\frac{\left[\underline{R}_g^{(m_g)}(\ell)\right]^T\left[\underline{R}_g^{(m_g)}(\ell)\right]}{\left[R_g^{(1)}(1)\right]^T\left[\underline{R}_g^{(1)}(\ell)\right]}\right\}^{1/2}, \text{ where}$$

q is the number of tape passes needed to perform the m_g inner iterations in group g.

DELTA is directly proportional to the error reduction in the inner iterations. However, since the constant of proportionality varies from problem to problem, this quantity probably is of limited interest to the user. After the first few outer iterations, a sudden large change in DELTA sometimes implies a machine error.

V. DISCUSSION AND NUMERICAL EXAMPLES

In this chapter we shall discuss various topics pertaining to the practical use of the PDQ program. We shall discuss

(a) the importance of the flux guess,

(b) estimates for the first overtone mode eigenvalue, and

(c) the convergence rate of the inner iterations.

A. THE FLUX GUESS

In this section we give no specific advice on how to specify an initial flux guess. Instead, we seek only to discuss the importance of the flux guess and how the efficiency of the PDQ program can be affected adversely by a "bad" flux guess.

For the present we shall neglect the effect of the inner iterations. We also shall assume that the eigenvectors $\{\underline{\Phi}_i\}$ of $M^{-1}F$ are normalized such that $\underline{\Phi}_i^T\underline{\Phi}_i = 1.0.$[1]

From assumption (b) of (3.2), the initial flux guess $\underline{\Phi}(0)$ may be expressed uniquely as a linear combination of the $\underline{\Phi}_i$,

$$\underline{\Phi}(0) = \sum_{i=1}^{GN} \tilde{c}_i \underline{\Phi}_i \quad , \tag{5.1}$$

where the $\tilde{c}_i$ are the scalar expansion coefficients. If we assume[2] that $\tilde{c}_1 \neq 0$, then we may write $\underline{\Phi}(0)$ as

$$\underline{\Phi}(0) = \tilde{c}_1 \left\{ \underline{\Phi}_1 + \sum_{i=2}^{GN} c_i \underline{\Phi}_i \right\} \quad ,$$

[1] We do this to make the "size" or "length" of each eigenvector the same.

[2] This must be true if all the components of $\underline{\Phi}(0)$ are positive.

where $c_i = \tilde{c}_i/\tilde{c}_1$. Since $\tilde{c}_1$ reflects only the level of the flux guess, we may take $\tilde{c}_1$ to be unity so that $\Phi(0)$ may be written as

$$(5.2) \qquad \Phi(0) = \Phi_1 + \sum_{i=2}^{GN} c_i \Phi_i \quad .$$

Since the eigenvectors Φ_i have been normalized to the same "size", the scalars c_i determine the degree of "goodness" of the initial flux guess. In successive outer iterations, the PDQ program seeks to reduce[1] the coefficients of the overtone modes[2] in the flux guess. Convergence of the outer iterations is achieved when the coefficients have been reduced far enough so that $\Phi(\ell)$ approximates Φ_1 to the accuracy desired. How much these coefficients have to be reduced depends on the flux guess.

The magnitude of an expansion coefficient corresponding to a slowly decaying overtone mode is more crucial than that corresponding to a rapidly decaying overtone mode. Also, as we shall see later, the efficiency of the Chebyshev polynomial method of iteration is affected to some degree by the magnitude of the expansion coefficients of the initial flux guess.

For a given problem and a given flux guess, one usually does not have the foggiest idea as to the values of the coefficients c_i. Thus, to discuss flux guesses in terms of the c_i is fruitless. Likewise, it is fruitless to discuss the degree of "goodness" of the guess since, obviously, the best initial flux guess is the answer, Φ_1. Hence, we will discuss "bad" flux guesses.

[1] For example, from Eq. (3.6) we see that the coefficient c_i is multiplied by a factor (λ_i/λ_1) for every outer iteration of the power type.

[2] The eigenvector Φ_i is called an overtone mode if $i \neq 1$. The first overtone mode corresponds to Φ_2.

For a given problem with a given flux guess, the job of the PDQ program is to obtain the desired solution in the fewest number of outer iterations. Usually, the program can do this only if the Chebyshev polynomial method of iteration is used efficiently. But, as seen in Chapter III, the efficient use of the Chebyshev polynomial method depends on knowing λ_1 and the dominance ratio, both of which must be estimated by the program. How well the program can estimate these quantities is determined to some extent by the flux guess. Hence, by a "bad" flux guess we mean a flux guess which hinders the efficiency of the Chebyshev polynomial method of iteration.

Basically, the Chebyshev strategy of the PDQ program can be divided into three parts as follows:

(1) Initially, at least three iterations of the power type are carried out in order to obtain an initial estimate for the dominance ratio and a reasonable estimate for λ_1. We note that these power iterations will practically eliminate from the flux guess those overtone modes with small eigenvalues. (See Fig. 3.1) The Chebyshev method of iteration is started on outer iteration ℓ_1, where ℓ_1 is the smallest integer greater than or equal to 4 for which the following conditions are satisfied:

(a) $|\text{NORM}(\ell_1-1) - 1.0| \leq .025$

(b) $\text{EPS}(\ell_1-1) \leq .2$

(c) $\text{SIGMA EST.}(\ell_1) \leq 1.0$

(d) $\text{SIGMA EST.}(\ell_1) > .4$

(2) Then, low degree Chebyshev polynomials are applied repeatedly with the estimates for the dominance ratio being updated continuously. If the low degree Chebyshev polynomials are generated with the dominance ratio under-estimated,

then these polynomials will greatly reduce[1] all overtone modes in the flux guess except those with the largest eigenvalues, thus allowing relatively good convergence of the SIGMA estimates to the correct value for the dominance ratio.

(3) After the SIGMA estimates have converged, high degree Chebyshev polynomials are applied, if needed, to reduce those overtone modes with the largest eigenvalues.

In summary, the strategy of the PDQ program is to eliminate first the more rapidly decaying modes in the flux guess and then concentrate on the more slowly decaying overtone modes. Normally, for this strategy to be effective an increasing sequence of dominance ratio estimates is needed in the generation of successive Chebyshev polynomials.

In practice, the estimates for the dominance ratio provided by the program are usually smaller (at least for the early outer iterations) than the correct value. However, a flux guess which does not contain enough of the fundamental mode[2] may cause the initial eigenvalue estimates, $\lambda(\ell)$, to be too small which in turn may cause the initial estimates for the dominance ratio to be too large. Usually, a "bad" flux guess such as this is implied whenever conditions (a), (b), or (c) given above prevent the Chebyshev method of iteration from starting on outer iteration 4 or 5 and/or whenever the initial estimates for the dominance ratio over-estimate the correct value.

An implied "bad" flux guess as described above should be taken as a signal to the user that the program possibly could have solved his problem more

[1] For example, if $(\lambda_2/\lambda_1) = .889$ and if a 5-th degree Chebyshev polynomial is generated with $\sigma_0 = .8$, then all overtone modes, Φ_i, with $(\lambda_i/\lambda_1) \leq .8$ are multiplied by a factor smaller than .017 while the first overtone mode, Φ_2, is multiplied by .211.

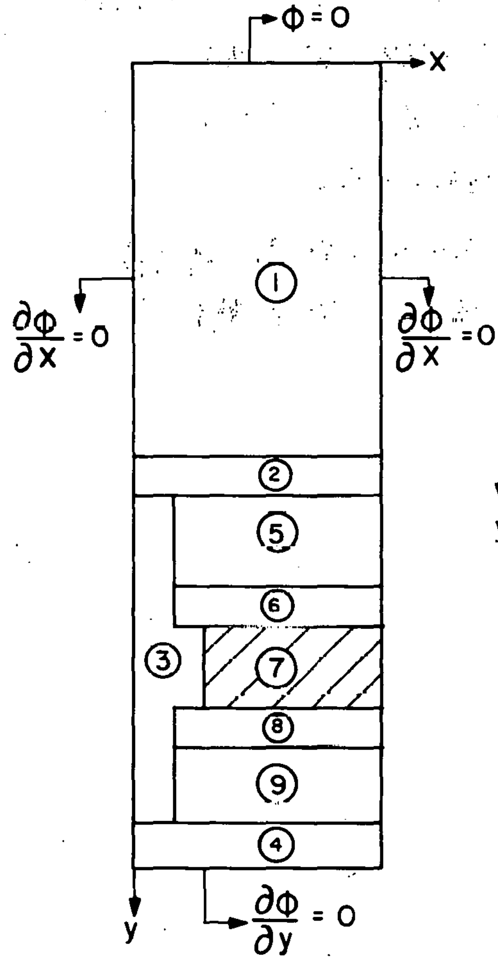
[2] That is, in Eq. (5.1), $\tilde{c}_1$ is smaller relative to the other $\tilde{c}_i$'s than we would like.

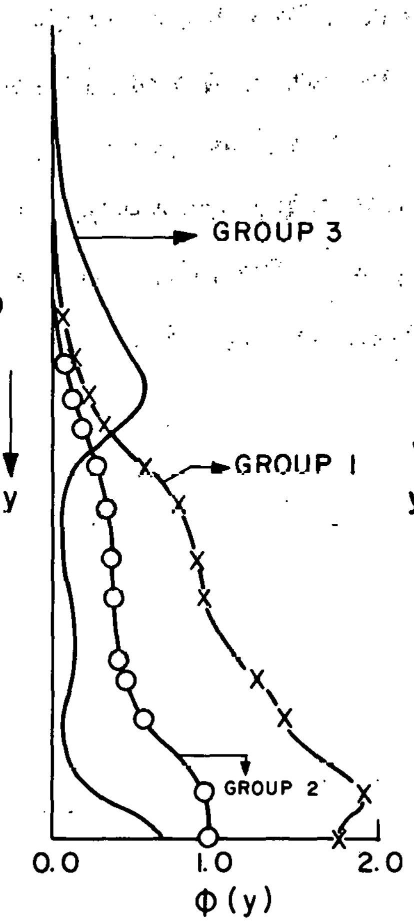
efficiently had he used a little better flux guess. We recognize that this is hindsight but hopefully this type of information will enable the user to obtain a better feel as to what constitutes a good flux guess.

In order to provide a flux guess for the PDQ program the user must specify an initial flux level and distribution for each group. For the flux level, it is the group-to-group flux ratios which are important. Thus, to specify an initial flux guess, the user should consider both the flux distribution within a group and the group-to-group flux levels.

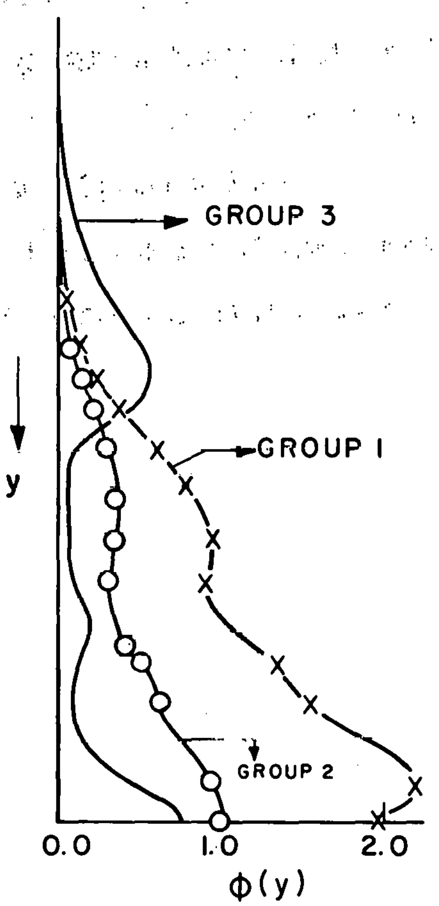
The two examples given below are production problems for which the user specified a bad flux guess. For both problems the ε_1 used in the termination of the outer iterations was .05.

PROBLEM I: 3 GROUPS





GRAPH OF GROUP FLUXES AT THE LEFT BOUNDARY



GRAPH OF GROUP FLUXES AT THE RIGHT BOUNDARY

For Problem 1, regions 1, 2, 3, and 4 are metal and water reflector regions; 5, 6, 8, and 9 are fuel regions; and 7 is a control rod.[1] The reactor is 12.6 centimeters wide and 52.18 centimeters long with 47 mesh points in the x direction and 81 mesh points in the y direction.

The flux distribution in the y direction is indicated in the above graphs. In the x direction, there are small wiggles in the fluxes in the fuel and rod regions.

	CASE 1			CASE 2		
COMPOSITION	GROUP 1	GROUP 2	GROUP 3	GROUP 1	GROUP 2	GROUP 3
1	.75	1.25	7.5	.11	.055	.33
2	.75	1.0	2.0	.44	.22	.33
3	.75	1.0	2.0	1.0	.44	.11
4	.75	1.0	2.0	2.0	1.0	.66
5	.75	1.0	2.0	1.0	.44	.11
6	.75	1.0	2.5	1.0	.44	.11
7	.75	1.0	2.0	1.0	.44	.11
8	.75	1.0	2.5	1.0	.44	.11
9	.75	1.0	2.0	1.0	.44	.11

INITIAL FLUX GUESS FOR PROBLEM 1

The flux guess for case 1 was supplied by the user who submitted the problem. The outer iteration data for both cases are listed in Table 5.1.

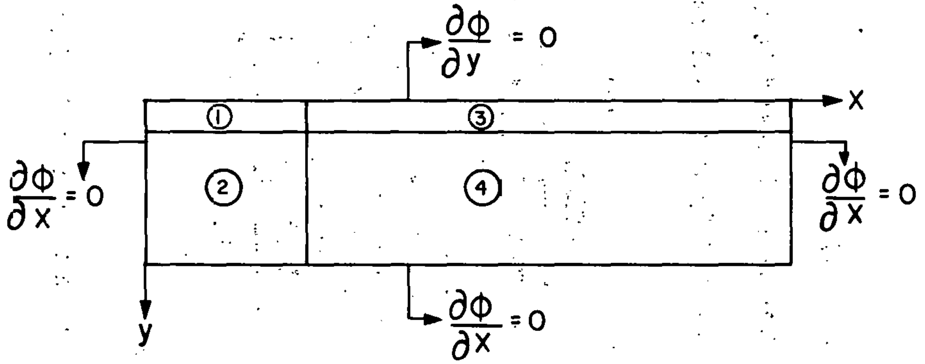
For the flux guess of case 1, we note that both the group-to-group flux ratios and the general flux shape within groups were bad. The reader should especially compare the LAMBDA and SIGMA EST. values for the two cases.

[1] The fuel and rod regions for problem 1 are actually more complicated than indicated in the picture.

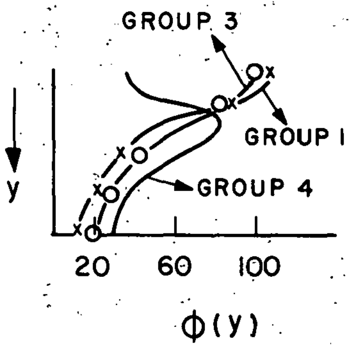
	CASE 1							CASE 2						
ITERATION	NORM	EPS	LAMBDA	PT/AV	RATIO	SIGMA EST.	DEGREE	NORM	EPS	LAMBDA	PT/AV	RATIO	SIGMA EST.	DEGREE
1	10.3060	10.1100	.67468	3.42	1.00	.000	0	.8849	1.2566	.93919	3.90	1.00	.000	0
2	1.2012	.3470	.81039	1.71	1.00	.099	0	1.0319	.1384	.96819	2.08	1.00	.141	0
3	1.0605	.1368	.84440	1.36	1.00	.399	0	1.0159	.0507	.91656	1.66	1.00	.129	0
4	1.0458	.0995	.87362	1.28	1.00	.911	0	1.0138	.0339	.91471	1.42	1.00	.499	1
5	1.0361	.0753	.88972	1.28	1.00	.856	0	1.0100	.0221	.91618	1.43	.52	.687	2
6	1.0278	.0569	.89983	1.30	1.00	.774	0	1.0063	.0129	.91876	1.42	.44	.678	3
7	1.0209	.0425	.90669	1.33	1.00	.748	0	1.0039	.0078	.92070	1.40	.34	.686	1
8	1.0154	.0313	.91154	1.36	1.00	.731	1	1.0024	.0049	.92188	1.42	.87	.719	2
9	1.0098	.0197	.91619	1.39	1.05	.720	2	1.0010	.0019	.92297	1.43	.85	.713	3
10	1.0036	.0071	.92092	1.40	1.15	.713	3							
11	1.0011	.0021	.92289	1.40	1.25	.713	4							

TABLE 5.1

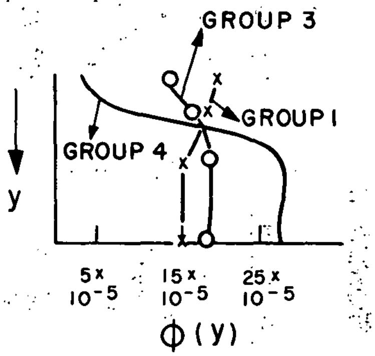
PROBLEM 2: 4 GROUPS



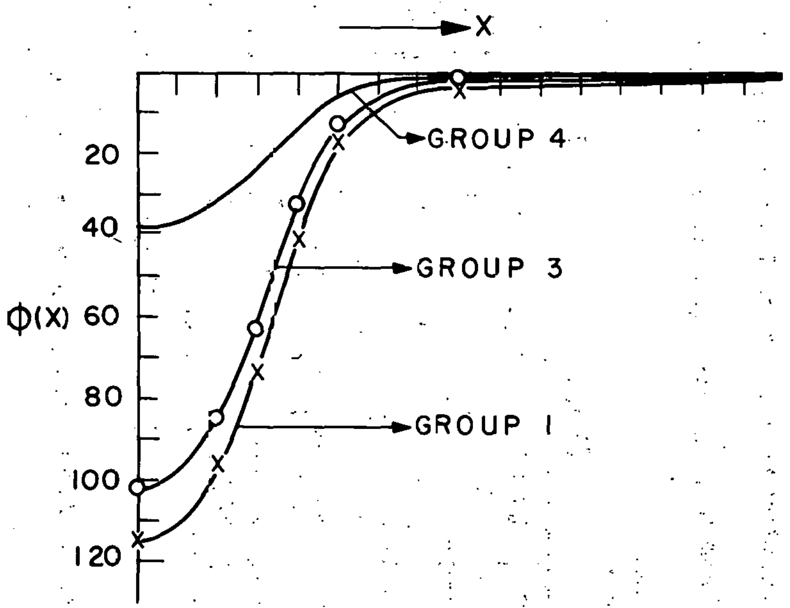
$\frac{\partial \Phi}{\partial y} = 0$
x
1
3
$\frac{\partial \Phi}{\partial x} = 0$
2
4
$\frac{\partial \Phi}{\partial x} = 0$
y
$\frac{\partial \Phi}{\partial x} = 0$



GRAPH OF GROUP FLUXES AT THE LEFT BOUNDARY



GRAPH OF GROUP FLUXES AT THE RIGHT BOUNDARY



GRAPH OF GROUP FLUXES AT THE TOP BOUNDARY

THE FLUX DISTRIBUTION FOR GROUP 2 IS SIMILAR TO THAT OF GROUP 1 AND IS NOT GIVEN IN THE GRAPHS. THE FLUX LEVEL FOR GROUP 2 IS SLIGHTY HIGHER THAN GROUP 1.

MACROSCOPIC CROSS SECTIONS FOR PROBLEM 2

GROUP	COMP	D	SIGMA A	SIGMA R	NU SIGMA F
1	1	1.9749	.00260	.07751	.00175
1	2	1.9239	.00488	.08167	.00725
1	3	1.9749	.00260	.07751	.00175
1	4	1.9239	.00488	.08167	.00725
2	1	.9884	.00149	.08783	.00219
2	2	.9668	.00220	.08311	.00022
2	3	.9884	.00159	.08783	.00219
2	4	.9668	.00220	.08311	.00022
3	1	.7201	.02164	.07985	.03078
3	2	.7139	.01882	.07769	.00331
3	3	.7201	.07164	.07985	.03078
3	4	.7139	.01882	.07769	.00331
4	1	.30916	.21700	0.0	.42896
4	2	.26956	.05405	0.0	.05795
4	3	.30916	.31700	0.0	.42896
4	4	.26956	.05405	0.0	.05795

The buckling for all groups and all compositions is .000147.

The reactor model for problem 2 is 187.8 centimeters wide and 30.1 centimeters long with 57 mesh points in the x direction and 23 mesh points in the y direction

CASE 1				
COMP	GROUP 1	GROUP 2	GROUP 3	GROUP 4
1	66.0	73.0	57.0	20.0
2	66.0	73.0	57.0	20.0
3	66.0	73.0	57.0	20.0
4	66.0	73.0	57.0	20.0

CASE 2			
GROUP 1	GROUP 2	GROUP 3	GROUP 4
66.0	73.0	57.0	20.0
14.0	18.0	16.0	21.0
.86	1.14	.98	1.34
.86	1.14	.98	1.34

INITIAL FLUX GUESS FOR PROBLEM 2

The outer iteration results for problem 2 are summarized in Tables 5.2A and 5.2B.

	CASE 1						
OUTER ITERATION	NORM	EPS	LAMBDA	PT/AV	RATIO	SIGMA EST.	DEGREE
1	.7138	1.0530	1.0006	1.96	1.00	0.0	0
2	.9428	.3107	.9433	1.62	1.00	.102	0
3	1.0089	.2042	.9973	1.40	1.00	.653	0
4	1.0299	.1727	1.0442	1.29	1.00	.942	0
5	1.0430	.1567	1.0812	1.29	1.00	1.017	0
6	1.0497	.1449	1.1087	1.34	1.00	.997	0
7	1.0510	.1359	1.1283	1.44	1.00	.963	0
8	1.0486	.1289	1.1418	1.59	1.00	.931	0
9	1.0440	.1235	1.1510	1.81	1.00	.903	0
10	1.0385	.1193	1.1573	2.11	1.00	.881	0
11	1.0328	.1161	1.1615	2.49	1.00	.863	0
12	1.0275	.1136	1.1644	2.99	1.00	.848	0
13	1.0227	.1117	1.1665	3.63	1.00	.836	0
14	1.0186	.1102	1.1679	4.44	1.00	.826	1
15	1.0130	.1083	1.1695	6.33	1.07	.816	2
16	1.0054	.1069	1.1711	1.5x10	1.21	.808	3
17	1.0016	.1075	1.1718	5.4x10	1.49	.802	4
18	1.0002	.1060	1.1720	3.9x10^{2}	2.32	.799	5
19	.9997	1.0517	1.1720	3.4x10^{3}	.15	.813	1
20	1.0001	2.1327	1.1720	2.5x10^{4}	2.34	.645	2
21	1.0000	88.99	1.1720	2.2x10^{6}	1.38	.749	3
22	1.0000	132.1	1.1720	3.5x10^{6}	.06	.816	1
23	1.0000	80.2	1.1720	7.2x10^{6}	2.92	.598	2
24	1.0000	360.6	1.1720	9.3x10^{7}	1.72	.729	3
25	1.0000	.1071	1.1720	2.3x10^{4}	-.42	.728	1
26	1.0000	.0522	1.1720	3.8x10^{4}	2.06	.557	2
27	1.0000	.0222	1.1720	5.9x10^{4}	1.53	.653	3
28	1.0000	.0115	1.1720	2.7x10^{4}	-.18	.655	1
29	1.0000	.0076	1.1720	8.7x10^{4}	2.06	.466	2
30	1.0000	.0043	1.1720	6.2x10^{4}	.61	.635	3
31	1.0000	.0026	1.1720	5.0x10^{4}	.17	.700	1
32	1.0000	.0018	1.1720	7.1x10^{4}	1.30	.639	2

TABLE 5.2A

	CASE 2						
OUTER ITERATION	NORM	EPS	LAMBDA	PT/AV	RATIO	SIGMA EST.	DEGREE
1	1.0320	16.845	1.1238	3.7x10	1.00	0.0	0
2	1.0253	.7290	1.1522	7.40	1.00	.096	0
3	1.0252	.3013	1.1603	6.36	1.00	.363	0
4	1.0204	.2147	1.1648	6.83	1.00	.569	0
5	1.0157	.1717	1.1673	7.68	1.00	.686	0
6	1.0121	.1448	1.1689	8.85	1.00	.734	1
7	1.0079	.1204	1.1702	1.2x10	.90	.755	2
8	1.0036	.1089	1.1714	2.5x10	.74	.770	3
9	1.0017	.1124	1.1718	5.4x10	.61	.777	1
10	1.0011	.1092	1.1719	7.8x10	1.02	.773	2
11	1.0005	.1081	1.1719	$1.7x10^2$	.97	.779	3
12	1.0002	.1103	1.1720	$4.0x10^2$	.89	.783	4
13	1.0001	.1077	1.1720	$9.3x10^2$	.85	.785	5
14	1.0000	.1052	1.1720	$2.2x10^3$	.83	.786	6
15	1.0000	.1033	1.1720	$4.9x10^3$	.83	.786	7
16	1.0000	.0944	1.1720	$9.9x10^3$	.83	.786	8
17	1.0000	.0802	1.1720	$1.8x10^4$	.81	.786	9
18	1.0000	.0618	1.1720	$2.9x10^4$	.79	.786	10
19	1.0000	.0409	1.1720	$4.1x10^4$	.78	.786	11
20	1.0000	.0241	1.1720	$5.2x10^4$	.78	.786	12
21	1.0000	.0132	1.1720	$6.2x10^4$	.78	.786	13
22	1.0000	.0069	1.1720	$6.9x10^4$	.79	.786	14
23	1.0000	.0034	1.1720	$7.3x10^4$	.79	.786	15
24	1.0000	.0016	1.1720	$7.2x10^4$	.79	.786	1

TABLE 5.2B

It is seen that the behavior of the LAMBDA and SIGMA estimates are the same as for problem 1, i.e., the bad guess gave lower estimates for λ_1 and overestimates for the dominance ratio.

The two examples given above illustrate the effect of the flux guess. In the solution of problem 1, the "bad" flux guess required 2 additional outer iterations and increased the total number of outer iterations by 22%. In the solution of problem 2, the "bad" flux guess required 8 addition outer iterations and increased the total number of outer iterations of 33%. Note that the dominance ratio for problem 2 is higher than that for problem 1. Generally, a "bad" flux guess can do more damage for problems which have a high dominance ratio.

The initial estimate for the fundamental eigenvalue usually is not very important. However, a good initial eigenvalue estimate is useful whenever a very good flux guess is used.

Before going on to the next section on the first overtone mode eigenvalue, we would like to make one additional comment concerning problem 2.

In case 2, EPS is the only quantity which is changing to any extent after outer iteration 13. Also, PT/AV is very large which implies that the relative sum error (3.16) is much smaller than the relative point error (3.9). This behavior is caused by the relatively small flux values near the right boundary. (These fluxes are roughly 10^{-7} times those on the left boundary.)

To see why these small flux values can cause trouble one need only examine the expression (3.13) for EPS. For the problem under consideration, the a_j for some of the points near the right boundary are probably much larger[1]

[1] We remark that the overtone modes need only satisfy the zero derivative condition at the right boundary.

than the a_j for the points near the left boundary. Since the program is set up to reduce all a_j (for all j in fissioning regions) below a certain value, we see that the points with large a_j will determine when the problem is converged.

Probably, for problem 2, the numbers of interest to the user were sufficiently accurate after 13 or 14 outer iterations. Thus, 10 outer iterations were wasted obtaining accurate answers for an insignificant region of the reactor. Whenever the group flux in fissioning regions varies by orders of magnitude and the boundary conditions do not force the overtone modes to behave similarly, this type of behavior in the outer iterations can be expected.[1] This particular bad program behavior probably could be eliminated by changing the program so that $\bar{\lambda}$ and λ are not computed over the insignificant regions.

B. THE FIRST OVERTONE MODE EIGENVALUE

In studying the xenon spatial stability of a core or the susceptibility of flux tilting due to perturbations in nuclear properties, the eigenvalue (λ_2) of the first overtone mode often must be calculated. This first overtone mode eigenvalue can be determined in a straightforward manner for one-dimensional (Ref. 6) and certain two-dimensional problems. The two-dimensional solution requires that the core have a symmetry axis that can be identified as a node line of the first overtone mode so that the first overtone mode and corresponding eigenvalue may be calculated directly via the PDQ program by placing a zero flux boundary along the node line.

Many cores have first overtone node lines which are not straight and can not easily be located exactly. For such cores, the program's estimate for

[1] For some problems, this type of outer iteration behavior would also occur if the program used EPS (Eq. 3.10) instead of $\overline{\text{EPS}}$ (Eq. 4.4) as the basis for convergence.

the dominance ratio often provides a good approximation for the first overtone eigenvalue.[1] The purpose of this section is to discuss this by-product of the PDQ program.

The SIGMA estimates provided by the program are approximations to the dominance ratio $\hat{\sigma}$. But, as seen in section B of Chapter III, $\hat{\sigma}$ differs from the true dominance ratio, $\bar{\sigma} = \frac{\lambda_2}{\lambda_1}$, of the problem because the inner iterations are not converged properly. Thus, in order for the SIGMA estimates to provide a good approximation for the first overtone eigenvalue one requires first good convergence of the SIGMA estimates to $\hat{\sigma}$ and second $\hat{\sigma}$ to differ only slightly from $\bar{\sigma}$.

Since $\hat{\sigma}$ approaches $\bar{\sigma}$ as the number of inner iterations approaches infinity, one can make $\hat{\sigma}$ agree more closely with $\bar{\sigma}$ by doing more inner iterations. As mentioned previously, the PDQ-5 program normally attempts to do only enough inner iterations so that the final error in each group is about .1 times the initial error. However, an option is available to converge the inner iterations more tightly. If ε_2 on input card 010003 is set equal to -1.0, then the program tries to do enough inner iterations so that the final error is about .01 times the initial error.[2]

Table 5.3 indicates how $\hat{\sigma}$ might vary with the number of inner iterations performed. Case 3 corresponds to the normal PDQ program and case 2 corresponds to the special ε_2 option. When ε_2 is set to -1.0, $\hat{\sigma}$ often is in the range $\bar{\sigma} - .04(1 - \bar{\sigma}) \lesssim \hat{\sigma} \leq \bar{\sigma} + .04(1 - \bar{\sigma})$.

[1] Normally, the program's estimate for the dominance ratio can be used to approximate the first overtone eigenvalue of the core only when full core problems are run. The first overtone mode of the core usually is not present in half core, quarter core, and cell problems.

[2] Normally, this option will cause the number of inner iterations to be increased by about 70%.

	INNER ITERATIONS			
CASE	GROUP 1	GROUP 2	GROUP 3	$\hat{\sigma}$
1	∞	∞	∞	.898
2	25	13	15	.900
3	14	7	8	.912
4	13	6	6	.922
5	6	3	2	.958

TABLE 5.3

VARIATION OF σ WITH THE NUMBER OF INNER ITERATIONS FOR A THREE GROUP PROBLEM

The major difficulty in using the SIGMA estimates to approximate λ_2 is the convergence of the SIGMA estimates. Basically, the convergence of the SIGMA estimates to $\hat{\sigma}$ depends on the success of the program strategy to reduce the overtone modes in the flux guess in such a manner that the first overtone mode is the last to go; i.e., a point is reached in the outer iterations where the first overtone mode is the only overtone mode of any significance remaining from the flux guess.[1] The user must examine the outer iteration output data to determine if the Chebyshev strategy described previously in section A is being carried out successfully.

To illustrate how the behavior of the outer iterations may be interpreted relative to the Chebyshev strategy, we shall consider the two cases of problem 2. The behavior of case 2 is very good: the SIGMA estimates basically constitute a convergent increasing sequence and a high degree Chebyshev polynomial was applied.

[1] Throughout this section, we assume that the first overtone mode is present in the flux guess.

The behavior of case 1 is bad: the SIGMA estimates first decrease and then fluctuate. Also, all but the first Chebyshev polynomial were of low degree. Note the behavior of RATIO for the two cases.

The rate at which the SIGMA estimates approach $\hat{\sigma}$ primarily depends on the flux-guess and the separation of the first overtone eigenvalue from the other overtone eigenvalues. We remark that using only the power method of iteration is a safer but much less efficient way to obtain a good estimate for $\hat{\sigma}$.

The intent of this section was not to suggest another primary use of the program but merely to discuss a program by-product which may be of casual interest to the user.

C. INNER ITERATION EFFICIENCY

The inner iterations for group g are used to obtain an approximation to the solution of the matrix equation (3.30),

$$A_g \underline{s}_g(\ell) = \underline{b}_g(\ell) \, , \tag{5.3}$$

where A_g is a non-singular matrix and $\underline{b}_g(\ell)$ is a known column vector. The matrix A_g corresponds to the discrete analogue of the diffusion and total absorption terms in the group equation (2.1). In what follows we shall drop the group subscript g and the outer iteration index ℓ in Eq. (5.3).

In order to specify completely the matrix equation (5.3), we need to backtrack a little and say something about the mesh which is imposed and the ordering of the unknowns.

Let a mesh of V vertical lines and H horizontal lines be imposed on the rectangular region R (Fig. 5.1) with the boundary conditions as indicated. Since

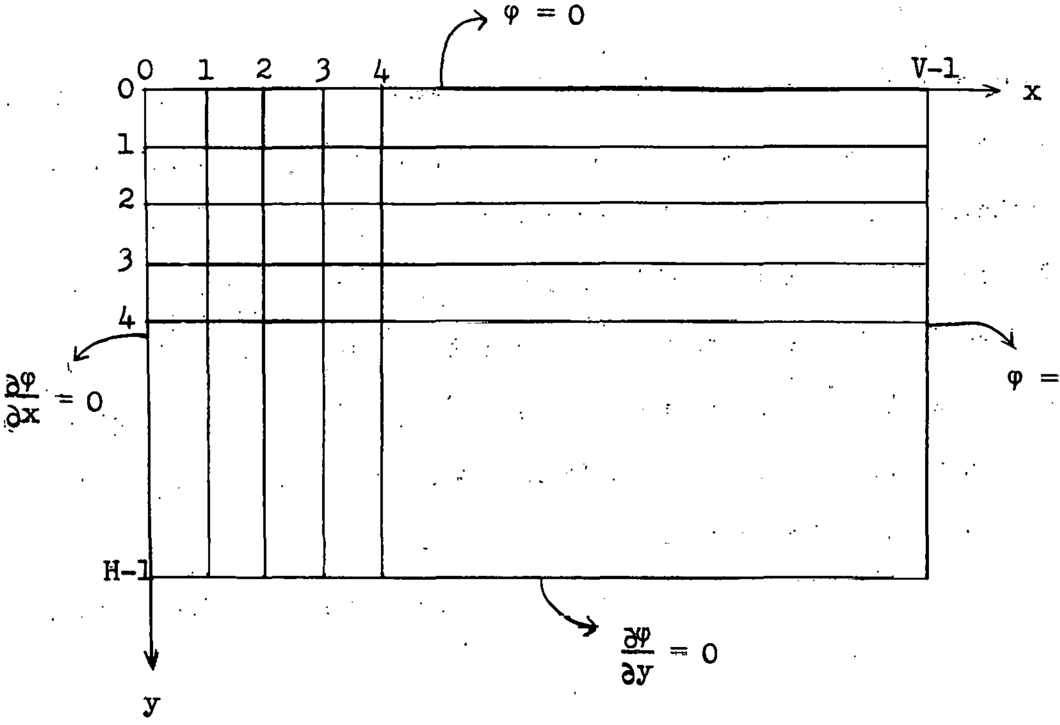


FIGURE 5.1

the flux is known along the top and right boundaries, we have only $(V-1)(H-1) \equiv N$ solution points[1] for the case of Fig. 5.1. Now to each solution point we assign an index number consecutively by rows (see Fig. 5.2) and then order the unknowns such that s_i corresponds to the unknown at the solution point of index i.

With this ordering of the unknowns, the explicit form of (5.3) is given by Eq. (5.4). The non-zero elements of A are the a_N, a_S, a_E, a_W, and a_P given in the finite difference expression (2.2). For the solution point i, $a_{i,i-1} = a_W$, $a_{i,i+1} = a_E$, $a_{i,i+(V-1)} = a_S$, $a_{i,i-(V-1)} = a_N$, and $a_{i,i} = a_P$. We say that solution point i is <u>coupled</u> to solution point j by $a_{i,j}$. The matrix A is symmetric so that the coupling from point i to point j is the same as the coupling from point j to point i.

[1] A solution point is a mesh point at which the flux is not known.

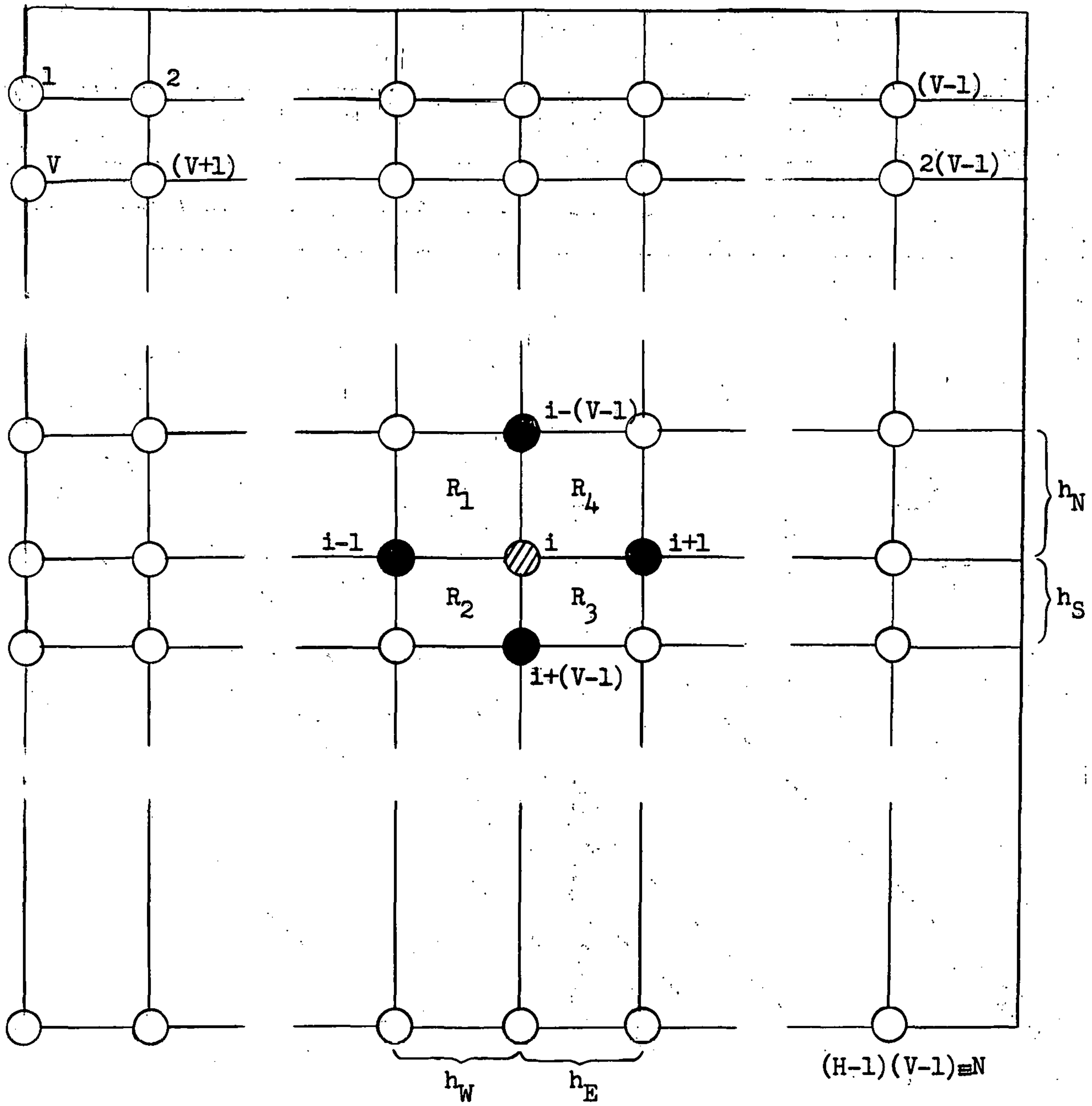


FIGURE 5.2

$$
\begin{bmatrix}
a_{1,1} & a_{1,2} & \ddots & a_{1,V} & \ddots & & \bigcirc \\
a_{2,1} & a_{2,2} & \ddots & & \ddots & & \\
\ddots & \ddots & \ddots & \ddots & \ddots & \bigcirc & \\
a_{V,1} & \ddots & \ddots & \ddots & \ddots & \ddots & \\
\ddots & a_{i,i-(V-1)} & \ddots & a_{i,i-1} \; a_{i,i} \; a_{i,i+1} & \ddots & a_{i,i+(V-1)} & \\
& \ddots & \bigcirc & \ddots & \ddots & \ddots & \ddots \\
\bigcirc & & \ddots & \ddots & \ddots & \ddots & \\
& & & \ddots & \ddots & \ddots & a_{N,N}
\end{bmatrix}
\begin{bmatrix}
s_1 \\ s_2 \\ | \\ s_V \\ | \\ s_i \\ | \\ s_N
\end{bmatrix}
=
\begin{bmatrix}
b_1 \\ b_2 \\ | \\ b_V \\ | \\ b_i \\ | \\ b_N
\end{bmatrix}
\qquad (5.4)
$$

The PDQ-2 program used a "point" iterative method to solve Eq. (5.3). By "point" method, it is meant that the approximate solution is improved one point at a time, i.e., first point 1 is improved, then point 2 and so on. The other PDQ programs used "ℓ-line" iterative methods, for $\ell = 1, 2$, and 3. By a "ℓ-line" method, it is meant that the approximate solution is improved ℓ-lines at a time, i.e., first, lines 1 through ℓ are improved simultaneously, then lines $\ell+1$ through 2ℓ and so on.

For a "ℓ-line" method, the number of horizontal solution rows[1] must be divisible by ℓ. Now if each successive block of ℓ successive solution rows is taken to be a unit, then the number of inner iterations depends on how strongly these units are coupled to each other. More inner iterations are required when the units are strongly coupled.

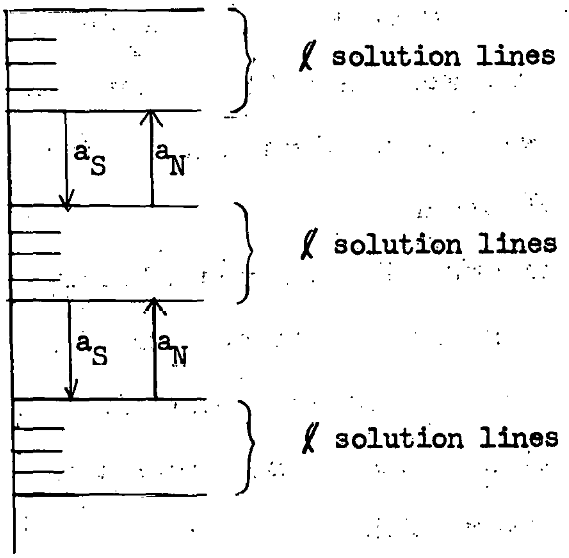
From Fig. 5.2 and the finite difference expression (2.2), we see that the coupling between adjacent units is through the a_S and a_N terms. Actually, it is these terms normalized by a_P which affect the number of inner iterations. We note that each of these terms, (a_S/a_P) and (a_N/a_P), is greater than zero and less than one. The units are strongly coupled when the normalized couplings between units are "close" to one.

We remark that it is the couplings <u>between</u> units which have the greatest effect on the inner iterations. For 1-line methods <u>every</u> a_S and a_N term is a coupling between units. However, for multi-line ($\ell \geq 2$) methods, less than $1/\ell$ of the a_S and a_N terms are couplings between units. For example, in a 2-line method the a_S for odd solution lines and a_N for even solution lines are <u>not</u> couplings between units. See Fig. 5.3. The couplings between units are called <u>exterior couplings</u> and the couplings of solution lines within a unit

[1] A solution row is a mesh row on which the flux is not known.

are called <u>interior couplings</u>. The exterior couplings for an ℓ-line method are shown in Fig. 5.3

FIGURE 5.3



The present version of the PDQ-5 program uses a 1-line method and thus <u>every</u> a_S and a_N term is an exterior coupling. The a_S and a_N terms are similar so we will examine only the a_S/a_P term. For the general solution point i of Fig. 5.2, we have

$$(5.5)\quad a_S/a_P = \frac{\dfrac{D_3h_E + D_2h_W}{2h_S}}{a_N + a_S + a_E + a_W + 1/4\ \{\Sigma_1h_Nh_W + \Sigma_2h_Sh_W + \Sigma_3h_Sh_E + \Sigma_4h_Nh_E\}}$$

Using Fig. 5.2 and Eq. (5.5), we make the following observations:

(a) For a 1 region problem with equal mesh spacings, $a_S/a_P \leq 1/4$. 1/4 may be taken as the norm for the (a_S/a_P).

(b) A fine mesh in the y direction and a coarse mesh in the x direction will cause the a_S/a_P to be abnormally high.

(c) Abrupt changes in the y direction mesh spacings can cause a_S/a_P to be close to unity.

(d) Large sigma total terms, the Σ's, are helpful but the relative worth of these Σ terms is diminished by small mesh spacings and/or large diffusion constants.

As indicated by (b) and (c) above, the orientation of the mesh may greatly affect the inner iterations. Given below are some obvious suggestions concerning the orientation of the mesh. We note that the orientation of the mesh is <u>not flexible</u> in r-z geometry (or when rotational symmetry is one of the boundary conditions).

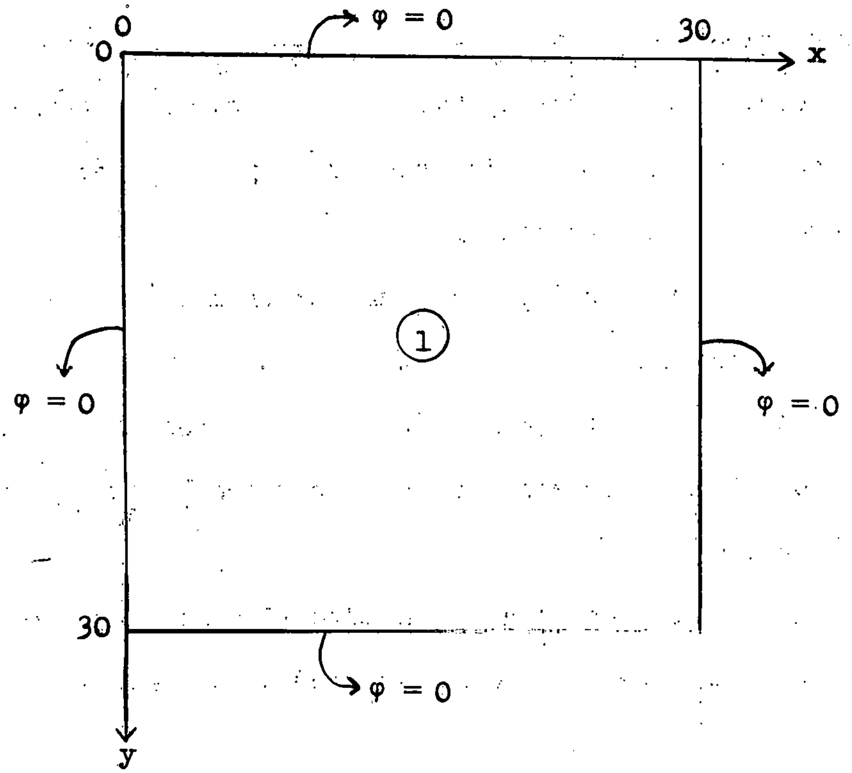
(1) If there is a fine mesh in one direction and a coarse mesh in the other direction, then the mesh should be oriented such that the coarse mesh is in the y direction.

(2) If there are abrupt changes in the mesh spacing in one direction and rather constant mesh in the other direction, then the mesh should be oriented such that the more constant mesh is in the y direction.

Often, it is not clear which orientation of the mesh is preferable. In this case the user must rely on his own experience.

The problems given below illustrate the effects of orientation on the inner iterations. The suffix R means that the mesh has been rotated 90°, i.e., the other orientation of the mesh was used in solving the problem.

PROBLEM 3:

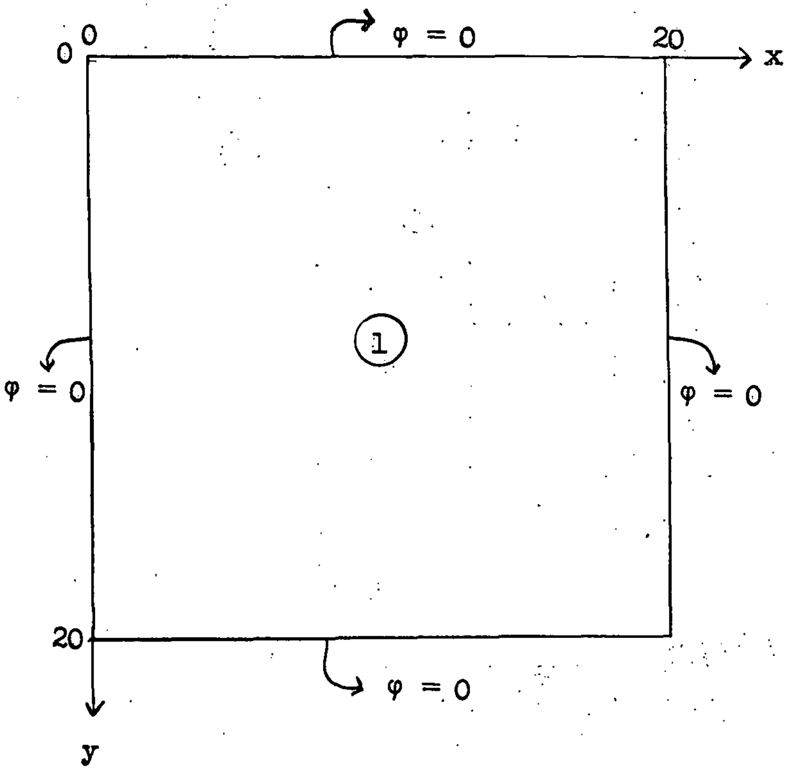


INTERVAL COL.		INTERVAL ROW	
3.0	30	1.0	30

COMP	D	SIGMA A	SIGMA R	BUCKLING
1	1.0	0.0	0.0	0.0

	PROBLEM 3	PROBLEM 3R
NUMBER OF INNER ITERATIONS PER OUTER	15	5

PROBLEM 4:



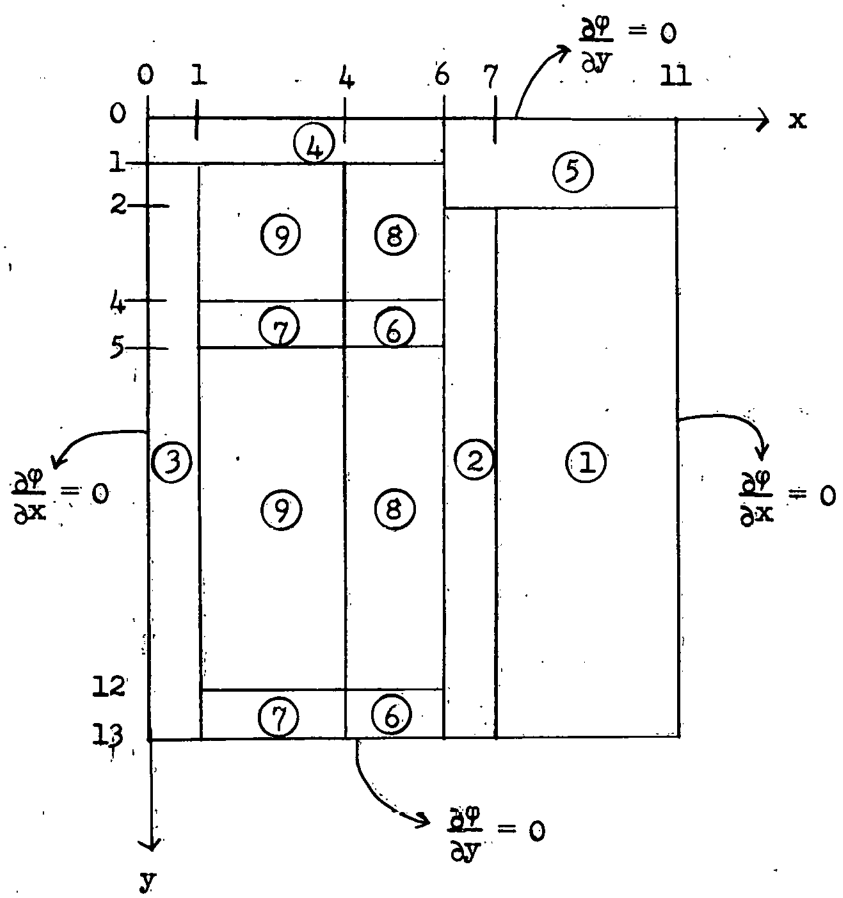
INTERVAL	COL
1.0	20

INTERVAL	ROW	INTERVAL	ROW	INTERVAL	ROW
1.0	10	X	11	1.0	20

COMP	D	SIGMA A	SIGMA R	BUCKLING
1	1.0	0.0	0.0	0.0

	PROBLEM 4			PROBLEM 4R		
	x=.4	x=.08	x=.01	x=.4	x=.08	x=.01
NUMBER OF INNER ITERATIONS PER OUTER	8	12	27	7	7	7

PROBLEM 5:



0
1
4
6
7
11
x
0
1
2
4
5
12
13
y
$\frac{\partial\varphi}{\partial y} = 0$
$\frac{\partial\varphi}{\partial x} = 0$
$\frac{\partial\varphi}{\partial x} = 0$
$\frac{\partial\varphi}{\partial y} = 0$
④
⑤
⑨
⑧
⑦
⑥
③
⑨
⑧
②
①
⑦
⑥

INTERVAL	COL
.79500	1
1.01660	7

INTERVAL	COL
1.8055	2
.3175	8

INTERVAL	COL
3.61010	3
.52917	11

INTERVAL	COL
1.0833	6

INTERVAL	ROW
.99440	1
.19050	5
2.26060	10

INTERVAL	ROW
1.86310	2
1.20400	6
.56515	12

INTERVAL	ROW
.31750	3
.19050	7
.09525	13

INTERVAL	ROW
.7455	4
1.1303	8

MACROSCOPIC CROSS SECTIONS FOR PROBLEM 5

GROUP	COMP	D	SIGMA A	SIGMA R	BUCKLING
1	1	2.557	.0012	.0870	.002
1	2	2.245	.0017	.0570	.002
1	3	2.171	.0018	.0471	.002
1	4	2.122	.0018	.0402	.002
1	5	2.107	.0018	.0380	.002
1	6	25.000	.0011	.0000	.000
1	7	25.000	.0011	.0000	.000
1	8	2.211	.0021	.0543	.002
1	9	2.198	.0025	.0547	.002
2	1	1.314	.00001	.1205	.002
2	2	1.113	.00049	.0573	.002
2	3	1.094	.00062	.0401	.002
2	4	1.072	.00070	.0290	.002
2	5	1.066	.00073	.0256	.002
2	6	25.000	.00572	.0000	.000
2	7	25.000	.00572	.0000	.000
2	8	1.115	.00116	.0518	.002
2	9	1.110	.00181	.0515	.002
3	1	.7383	.00077	.12258	.002
3	2	.9121	.00159	.05786	.002
3	3	.9845	.00160	.04014	.002
3	4	1.0401	.00155	.02862	.002
3	5	1.0585	.00152	.02513	.002
3	6	3.7175	.34145	.00000	.000
3	7	3.9342	.31927	.00000	.000
3	8	.9290	.01150	.04760	.002
3	9	.9245	.02386	.04159	.002
4	1	.2534	.0118	0.0	.002
4	2	.4272	.0082	0.0	.002
4	3	.5287	.0112	0.0	.002
4	4	.6265	.0066	0.0	.002
4	5	.1871	.0064	0.0	.002
4	6	.0659	3.0547	0.0	.000
4	7	.0771	2.8039	0.0	.000
4	8	.4879	.1009	0.0	.002
4	9	.4675	.1865	0.0	.002

	PROBLEM 5		PROBLEM 5R	
NUMBER OF INNER ITERATIONS PER OUTER	GROUP 1	48	GROUP 1	14
	GROUP 2	52	GROUP 2	11
	GROUP 3	21	GROUP 3	10
	GROUP 4	10	GROUP 4	8

For problems in r-z geometry or problems with identical (or similar) mesh spacings in both directions, the orientation of the mesh is either not flexible or immaterial. For these problems, the 1-line method does not offer much flexibility.

Multi-line methods are much more flexible in that it is possible to hide some of the "bad" a_S and a_N couplings as interior couplings. For example, in problem 5 the "bad" couplings are the a_S couplings from mesh lines 0, 2, 4, 6, 10, and 12 and the a_N couplings from mesh lines 1, 3, 5, 7, 11, and 13. But for a 2-line method, all of these couplings are interior couplings. To illustrate the advantages of the multi-line methods, problem 5 was solved using a 2-line method and the results are given below.

	PROBLEM 5 (2-LINE METHOD)	
NUMBER OF INNER ITERATIONS PER OUTER	GROUP 1	7
	GROUP 2	6
	GROUP 3	5
	GROUP 4	4

Problem 5 was chosen[1] judiciously to illustrate the effectiveness of the 2-line method. For problems with rather homogeneous compositions and rather constant y direction mesh spacings, the effect of multi-line methods is not so dramatic.[2]

Perhaps a multi-line method will be feasible on the next computer. Thus, for possible future use, we give a few suggestions concerning the mesh laydown for multi-line methods.

The orientation of the mesh should be chosen, if possible, using suggestions (1) and (2) given previously. One then should try to hide the "bad" a_S and a_N couplings by making them interior couplings. The user may have to add a mesh row or two to do this.

[1] Problem 5 was taken from Ref. 12.

[2] For a one region problem with constant mesh spacings, the improvement of the 2-line mothod over the 1-line method is about 1.4.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to Dr. N. J. Curlee for his many helpful comments concerning the contents of this report.

REFERENCES

1. Birkhoff, G. and Varga, R. S. [1958] "Reactor Criticality and Nonnegative Matrices". Journal of the Soc. for Indust. and Appl. Math., Vol. 6, pp 354-377.

2. Cadwell, W. R. [1963] "PDQ-5--A FORTRAN Program for the Solution of the Two-Dimensional Neutron-Diffusion Problem--Part 1: Steady-State Version". Bettis Atomic Power Laboratory Report WAPD-TM-363.

3. Cadwell, W. R. [1964] "The Input-Output Problem in Two-Dimensional Diffusion Programs with Large Meshes". Transactions of the American Nuclear Society, Vol. 7, No. 1, pp 23-24.

4. Cadwell, W. R., Buerger, P. F. and Pfeifer, C. J. [1964] "The PDQ-5 and PDQ-6 Programs for the Solution of the Two-Dimensional Neutron Diffusion-Depletion Problem". Bettis Atomic Power Laboratory Report WAPD-TM-477.

5. Downing, A. C. Jr. [1960] "On the Convergence of Steady State Multiregion Diffusion Caclulation". Oak Ridge National Laboratory Report ORNL-2961.

6. Ewen, R. [1960] "MULE--A Code for the Calculation of Two Types of Overtone Modes". Trans. Am. Nucl. Soc. 3(2), pp 397-398.

7. Flanders, D. and Shortley, G. [1950] "Numerical Determination of Fundamental Modes". J. of Appl. Phys. Vol. 21, pp 1326-1332.

8. Forsythe, G. and Wasow, W. [1960] Finite Difference Methods for Partial Differential Equations, John Wiley and Sons, Inc., New York, London.

9. Habetler, G. and Martino, M. [1961] "Existence Theorems and Spectral Theory for the Multigroup Diffusion Model". Proceedings of Symposia in Applied Mathematics, Vol. XI, pp 127-139.

10. Hageman, L. [1963] "Numerical Methods and Techniques Used in the Two-Dimensional Neutron Diffusion Program PDQ-5", Bettis Atomic Power Laboratory Report WAPD-TM-364.

11. Marlowe, O. and Suggs, M. [1960] "WANDA-5--A One-Dimensional Neutron Diffusion Equation Program for the Philco-2000 Computer". Bettis Atomic Power Laboratory Report WAPD-TM-241.

12. Marlowe, O. [1961] "Nuclear Depletion Programs for the Philco-2000 Computer". Bettis Atomic Power Laboratory Report WAPD-TM-221.

13. Pfeifer, C. [1963] "Data Flow and Storage Allocation for the PDQ-5 Program on the Philco-2000". Communications of the ACM, Vol. 6, No. 7, pp 365-366.

14. Varga, R. [1957] "Numerical Solution of the Two-Group Diffusion Equation in x-y Geometry". IRE Trans. of the Professional Group on Nuclear Science, Vol. NS-4, pp 52-62.

15. Varga, R. [1961] "Numerical Methods for Solving Multi-Dimensional Multigroup Diffusion Equations". Proceedings of Symposia in Applied Mathematics, Vol. XI, pp 164-189.

16. Varga, R. [1962] Matrix Iterative Analysis, Prentice-Hall, Inc. Englewood Cliffs, New Jersey.