SYNTRON, a tree-dimensional flux synthesis programme

Larsen, Hans Hvidtfeldt

Publication date:
1971

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Larsen, H. H. (1971). SYNTRON, a tree-dimensional flux synthesis programme. (Risø-M; No. 1346).

A. E. K. Risø

Risø-M-1346

Risø-M-1346

Title and author(s) SYNTRON, a Three-dimensional Flux Synthesis Programme by Hans Larsen	Date
	Department or group Reactor Physics
	Group's own registration number(s)
24 pages + tables + illustrations	
Abstract	Copies to
	Abstract to

Abstract

The report describes the programme SYNTRON, which is a three-dimensional variational single-channel flux synthesis code. Besides the actual synthesis, the programme contains a subroutine for calculation of the two-dimensional expansion functions by use of ordinary difference equation technique.

FI 25-204

CONTENTS

Page

1. Introduction 1

2. The Physical Principles of Flux Synthesis 1
   - 2.1. The Diffusion Equations 1
   - 2.2. The Synthesis Approximation 2
   - 2.3. The Basic Synthesis Equation 2
   - 2.4. The Mutual Coupling, Weighting Functions 3
   - 2.5. The z-Dependence of the Mixing Functions 5

3. Practical Solution Technique 7
   - 3.1. Cross Section Integration and Coefficient-Matrix Preparation 7
   - 3.2. Boundary Conditions 9
   - 3.3. Solution of the Equation System 10
   - 3.4. Convergence Acceleration Technique 13
   - 3.5. Three-Dimensional Flux and Power Shapes 14
   - 3.6. Calculation of Trial and Weighting Functions 15

4. Summary 16

References 20

Appendix I 21

Appendix II 26

1. INTRODUCTION

Several reactor physics problems involve three-dimensional neutron flux calculations. The classical technique for solution of the multi-dimensional diffusion equations is the difference approximation technique. Actually, the difference technique is the most straightforward and the most accurate solution technique, and its only drawback is that it requires too much computing time. For many purposes such as burn-up and void calculations where many flux calculations are necessary, the computing time is quite unacceptable if ordinary difference equation techniques are used. Therefore many approximate methods have been developed, and one these is flux synthesis. In a few words flux synthesis consists in expanding of the three-dimensional flux after some two-dimensional precalculated flux functions. The first approach to flux synthesis was made by Meyer in the so-called single-channel flux synthesis (ref. 1), and later the multi-channel synthesis was developed by Wachspress (ref. 2). In this report a computer programme called SYNTRON will be presented. SYNTRON uses a variational single-channel flux synthesis technique primarily based on refs. 3 and 7.

Besides the actual synthesis, the SYNTRON programme contains a subroutine for calculation of the two-dimensional expansion functions by use of ordinary difference equation technique.

2. THE PHYSICAL PRINCIPLES OF FLUX SYNTHESIS

2.1. The Diffusion Equations

In this chapter the physical principles for solution of the three-dimensional diffusion equations by use of flux synthesis will be presented.

We have the multi-group diffusion equations:

$$-D^g\nabla^2 \phi^g + A^g\phi^g = S^g \tag{1}$$

where

D^g = diffusion coefficient in group g

A^g = absorption cross section in group g

S^g = source in group g.

The source term is given by the following expression:

$$S^g = \sum_{g'=1}^{GR} (\Sigma_s^{g \leftarrow g'} + xi^g \cdot \gamma \Sigma_f^{g'} / k_{eff}) \cdot \emptyset^{g'}. \tag{2}$$

Here

GR = number of groups

$\Sigma_s^{g \leftarrow g'}$ = scattering cross section from group g' to group g

xi^g = fission spectrum

$\gamma \Sigma_f^{g'}$ = γ-fission cross section in group g'

k_{eff} = effective multiplication factor (eigenvalue of the problem).

2.2. The Synthesis Approximation

In the synthesis approximation for solution of the multi-group diffusion equations (1) an attempt is made to find the three-dimensional solution $\emptyset^g$ (z,y,x) by expansion of $\emptyset^g$(z,y,x) after some two-dimensional precalculated flux functions, H^g(y,x), called trial functions. In the SYNTRON programme a variational single-channel flux synthesis formalism primarily based on refs. 3 and 7 was used.

As Kaplan describes in ref. 3 the group flux $\emptyset^g$(z,y,x) is found by means of the following expansion:

$$\emptyset^g(z,y,x) = \sum_{k=1}^{K_g} Z_k^g(z) \cdot H_k^g(y,x), \tag{3}$$

where

K_g = number of trial functions in group g

$H_k^g(y,x)$ = trial function number k in group g

$Z_k^g(z)$ = mixing function number k in group g.

If the $H_k^g(y,x)$ function set is assumed to be known, then the problem is how to find the mixing functions.

2.3. The Basic Synthesis Equation

If the flux expansion (3) is substituted into the group diffusion equations (1) and (2), we have

$$(-D^g\nabla^2 + A^g) \cdot \left(\sum_{k=1}^{K_g} Z_k^g(z) \cdot H_k^g(y,x) \right) =$$

$$\sum_{g'=1}^{GR} \sum_{i=1}^{K_{g'}} (\Sigma_s^{g+g'} + xi^g \cdot \gamma\Sigma_f^{g'}/k_{eff}) \cdot Z_i^{g'}(z) \cdot H_i^{g'}(y,x). \tag{4}$$

Equation (4) is the basic equation for finding the unknown mixing functions $Z_k^g(z)$.

The variational formalism for finding the mixing functions is fully discribed in refs. 6 and 3.

In the mixing function set there are two sorts of coupling:

1. The mutual coupling between $Z_1^g(z) \dots Z_{K_g}^g(z)$ for fixed z.
2. The z-dependence of the mixing functions.

2.4. The Mutual Coupling, Weighting Functions

The mutual coupling between the mixing functions $Z_1^g(z) \dots Z_{K_g}^g(z)$ for fixed z is found by introduction of a new function set, the weighting function set $W_k^g(y,x)$. For weighting functions one may either select the trial functions or the adjoint trial functions. In the first case, $W_k^g(y,x) = H_k^g(y,x)$, we have Galerkin weighting, in the second, $W_k^g(y,x) = H_k^{*g}(y,x)$, Selengut weighting, compare refs. 3, 5, 6 and 7. The following treatment is independent of the method used.

Eq. (4) is multiplied by $W_j^g(y,x)$, and a integration over the (y,x) plane is performed for fixed z.

Then we have

$$\int_y \int_x \left((-D^g\nabla^2 + A^g) \cdot \left(\sum_{k=1}^{K_g} Z_k^g(z) \cdot H_k^g(y,x) \cdot W_j^g(y,x) \right) \right) dy\,dx =$$

$$\int_y \int_x \sum_{g'=1}^{GR} \sum_{i=1}^{K_{g'}} (\Sigma_s^{g+g'} + xi^g \cdot \gamma\Sigma_f^{g'}/k_{eff}) \times \tag{5}$$

$$Z_i^{g'}(z) \cdot H_i^{g'}(y,x) \cdot W_j^g(y,x)\, dy\,dx$$

$$j = 1 \dots K_g.$$

The first term of the left-hand side in eq. (5) is treated as follows:

$$-D^g \nabla^2 = -D^g(\nabla_z^2 + \nabla_{yx}^2) \quad . \tag{7}$$

From (7) we get two new terms

$$-DZ_{kj}^g(z) \cdot \nabla_z^2 Z_k^g(z) \quad ,$$

where

$$DZ_{kj}^g(z) = \int_y \int_x D^g \cdot H_k^g(y,x) \cdot W_j^g(y,x)\, dy\, dx \quad , \tag{8}$$

and

$$DBZ_{kj}^g(z) \cdot Z_k^g(z) \quad ,$$

where

$$DBZ_{kj}^g(z) = \int_y \int_x D^g \cdot W_j^g(y,x) \cdot \nabla_{yx}^2\, H_k^g(y,x)\, dy\, dx \quad . \tag{9}$$

The subscript k represents trial function no. k and the subscript j weight function no. j in group g. The terms $DBZ_{kj}^g(z)$ express the radial neutron leakage at the axial position z.

The second term in eq. (5) is treated just like (8)

$$A_{kj}^g(z) = \int_y \int_x A^g \cdot H_k^g(y,x) \cdot W_j^g(y,x)\, dy\, dx \quad . \tag{10}$$

Eq. (10) represents the total absorption terms.

The right-hand side of eq. (5) is treated in a similar fashion.

We have

$$SP_{ij}^{gg'}(z) = \int_y \int_x \Sigma_s^{g \leftarrow g'} \cdot H_i^{g'}(y,x) \cdot W_j^g(y,x)\, dy\, dx \quad ,$$

and (11)

$$XIF_{ij}^{gg'}(z) = \int_y \int_x xi^g \cdot \gamma \Sigma_f^{g'} \cdot H_i^{g'}(y,x) \cdot W_j^g(y,x)\, dy\, dx \quad .$$

According to the transformations above eq. (5) is replaced by:

$$\sum_{k=1}^{K_g} (-DZ_{kj}^g(z)\nabla_z^2 + DBZ_{kj}^g(z) + A_{kj}^g(z)) \cdot Z_k^g(z) =$$

$$\sum_{g'=1}^{GR} \sum_{i=1}^{K_{g'}} (SP_{ij}^{gg'}(z) + XIF_{ij}^{gg'}(z)/k_{eff}) \cdot Z_i^{g'}(z) \quad . \qquad (12)$$

This is performed for each j, $1{:}K_g$.

In this way we have K_g linear inhomogeneous equations for determining the K_g mixing function terms at any axial point z, in group no. g.

$$\sum_{k=1}^{K_g} L_{kj}^g \cdot Z_k^g(z) = Q_j^g \quad .$$

2.5. The z-Dependence of the Mixing Functions

We now return to the problem of finding the z-dependence of the mixing functions. We only consider mixing functions that are continuous in the argument z, through the whole reactor. On the other hand, we do not make any assumptions of complete continuity of the integrated cross sections, we only assume that the cross sections are constant piece by piece, that is, there are no variations in the cross sections in the z-direction in each axial zone.

The mixing equations (12) are solved approximately by a finite difference method. The axial zones are subdivided into a fine-mesh structure, and a mesh point is chosen in the middle of each mesh. Eq. (12) is transformed into a set of difference equations by means of an integration over the remaining direction, z.

If we neglect all subscripts (k, j, g) and only look at one term on the left-hand side of eq. (11), we have

$$\int_z (-DZ(z)\nabla_z^2 + DBZ(z) + A(z))\, Z(z)\, dz =$$

$$- \int_z DZ(z)\nabla_z^2\, Z(z)\, dz + \int_z DBZ(z)\, Z(z)\, dz + \int_z A(z)\, Z(z)\, dz \quad . \tag{14}$$

When the integration is performed mesh by mesh we have (compare ref. 10), if the length of mesh no. z is called LZ(z):

$$\int_z DBZ(z) \cdot Z(z)\, dz = DBZ(z) \cdot Z(z) \cdot LZ(z) \quad ,$$

$$\int_z A(z) \cdot Z(z)\, dz = A(z) \cdot Z(z) \cdot LZ(z) \quad , \tag{15}$$

and

$$\int_z DZ(z)\, \nabla_z^2\, Z(z)\, dz = \int_S DZ(z) \cdot \nabla_z\, Z(z)\, dS \quad ,$$

where S represents surface integration; in this one-dimensional case dS is taken as one area unit.

According to further approximations we have

$$\int_S DZ(z) \cdot \nabla_z\, Z(z)\, dS = \overline{DZ}_1 (Z(z) - Z(z-1)) + \overline{DZ}_2 \cdot (Z(z) - Z(z+1)) \quad , \tag{16}$$

where

$$\overline{DZ}_1 = 2/(LZ(z)/DZ(z) + LZ(z-1)/DZ(z-1)) \quad ,$$

and

$$\overline{DZ}_2 = 2/(LZ(z)/DZ(z) + LZ(z+1)/DZ(z+1)) \quad .$$

The right-hand side of eq. (12) is treated in the same way as eq. (15). Thus eq. (12) is transformed into a set of finite difference equations. The remaining problems are how to perform these transformations in practice, and how to solve the equation system.

3. PRACTICAL SOLUTION TECHNIQUE

3.1. Cross Section Integration and Coefficient - Matrix Preparation

In this chapter a brief description of the methods used to perform the cross section integration is presented. As mentioned before the trial functions and the weighting functions are taken to be precalculated function sets containing one flux point per mesh point in the x-y plane. How these functions are calculated will be shown later in this section.

When it is assumed that there is no variation in cross sections in axial direction, i.e. z-direction, in each zone, it is only necessary to perform one set of cross section integrations in each zone.

Fine and coarse mesh divisions in the x-y planes are shown in fig. 2.

The integrations of eq. (8), (9) and (11) are equivalent. Let us look at eq. (8).

$$DZ^g_{jk} = \sum_{y}^{I} \sum_{x}^{I} D^g(y,x) \cdot H^g_k(y,x) \cdot W^g_j(y,x) \cdot ARE(y,x) \quad , \qquad (17)$$

where

$$ARE(y,x) = \text{area of mesh } (y,x) \quad .$$

The most interesting and most complicated calculation is that of calculating the radial leakage terms $DBZ^g_{jk}(z)$. This calculation is performed in accordance with the method suhhested in ref. 7.

In the programme SYNTRON it is possible to handle an arbitrary distribution of meshes represented by cross sections, and meshes represented as boundary conditions, i.e. extrapolation factors $EXT^g(y,x)$. Boundary conditions will be further described in section 3.2.

Let us define the leakage terms from mesh (y,x) to its adjacent meshes

QA = leakage term from (y, x) to (y, x-1) ,
QB = leakage term from (y, x) to (y, x+1) ,
QC = leakage term from (y, x) to (y-1, x) ,
QD = leakage term from (y, x) to (y+1, x) .

We shall look at QA. If both mesh (y, x) and mesh (y, x-1) are represented by cross sections, we have just as in eq. (16), section 2.5,

$$QA = 2/(LX(x-1)/D^g(y,x-1) + LX(x)/D^g(y,x)) \ .$$

If mesh (y, x-1) is represented as boundary conditions,

$$QA = 2/(2/EXT^g(y,x-1) + LX(x)/D^g(y,x)) \ .$$

QB, QC, QD are treated in a similar way.

We are now ready to construct the $DBZ^g_{jk}(z)$ terms

$$DBZ^g_{jk}(z) = \sum_y{}' \sum_x{}' W^g_j(y,x) \times ($$

$$(QA(y,x) \cdot (H^g_k(y,x) - H^g_k(y,x-1)) +$$

$$QB(y,x) \cdot (H^g_k(y,x) - H^g_k(y,x+1))) \cdot LY(y) + \qquad (18)$$

$$(QC(y,x) \cdot (H^g_k(y,x) - H^g_k(y-1,x)) +$$

$$QD(y,x) \cdot (H^g_k(y,x) - H^g_k(y+1,x))) \cdot LX(x) \ ) \ .$$

The subscript ' indicates that the integrations are only performed at meshes tnat are represented by cross sections.

From these integrated quantities it is possible to construct the necessary coefficient matrices for the mixing function difference equation system.

Leakage terms between meshes nos. z and z+1 and vice versa are constructed in a fashion similar to that used for leakage terms in the x-y plane. We have

$$DD^g_{jk}(z) = 2/(LZ(z)/DZ^g_{jk}(z) + LZ(z+1)/DZ^g_{jk}(z+1)) \quad . \tag{19}$$

It is now possible to construct the total absorption terms $KOB^g_{jk}(z)$, i. e. the terms which include all the neutrons which leave the mesh z and the group g.

$$KOB^g_{jk}(z) = DD^g_{jk}(z-1) + DD^g_{jk}(z) + DBZ^g_{jk}(z) + A^g_{jk}(z) \quad , \tag{20}$$

where

DD^g_{jk} = axial leakage terms to the neighbouring meshes

DBZ^g_{jk} = radial leakage terms

A^g_{jk} = total absorption terms.

3. 2. Boundary Conditions

At each boundary a boundary condition is specified. As boundary conditions "gamma-matrix" formalism is used (ref. 11).

The flux at the boundary Ø and the current out through the boundary I are coupled by the γ-matrix

$$I = \gamma \cdot Ø$$

$$\begin{bmatrix} I^1 \\ I^2 \\ \cdot \\ I^G \end{bmatrix} = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1G} \\ \cdot & & & \\ \cdot & & & \\ \gamma_{G1} & \cdots & & \gamma_{GG} \end{bmatrix} \cdot \begin{bmatrix} Ø^1 \\ \cdot \\ \cdot \\ Ø^G \end{bmatrix} \tag{21}$$

The diagonal elements express the ordinary extrapolation factors

$$EXT^g = \gamma^{gg} \; .$$

The off-diagonal elements represent down- and up-scattering; these terms are added to the scattering matrices in the adjacent meshes.

In this way it is possible to treat control rods as boundary conditions and take into account the down- and up-scattering.

When only the diagonal elements are non-zero, "gamma-matrix" formalism is equivalent to the method of ordinary extrapolation factors in each group.

3. 3. Solution of the Equation System

The basic problem of flux synthesis is the solution of eq. (13). It is of great importance that the solution technique is fast and reliable.

In this programme a semi-iterative technique is used. The fine mesh structure in the axial zones is shown in fig. 1, one mesh point taken in the middle of each mesh.

The source terms in eq. (12) are calculated as follows:

$$Q_j^g(z) = \sum_{g'=1}^{GR} \sum_{i=1}^{K_{g'}} (SP'^{gg'}_{ij}(z) + XIF^{gg'}_{ij}(z)/k_{eff}) \cdot Z_i^{g'}(z) \qquad (22)$$

The ' on SP indicates that only terms where $g \neq g'$ are taken into account.

When the difference approximation and the coefficient matrices (19) and (20) are inserted into eq. (12), we get the following matrix equations:

$$\begin{aligned} &-\overline{\overline{DM}}^g(z-1) \cdot \overline{Z}^g(z-1) + \overline{\overline{AA}}^g(z) \cdot \overline{Z}^g(z) \\ &- \overline{\overline{DM}}^g(z) \cdot \overline{Z}^g(z+1) = \overline{Q}^g(z) \quad , \end{aligned} \qquad (23)$$

where

$$\overline{\overline{DM}}^g = \begin{bmatrix} DD^g_{11} \; DD^g_{12} & \text{-------------} & DD^g_{1K_g} \\ \vdots & & \vdots \\ DD^g_{K_g 1} & \text{-----------------} & DD^g_{K_g K_g} \end{bmatrix} ,$$

$$\overline{\overline{AA}}^g = \begin{bmatrix} KOB^g_{11} \; KOB^g_{12} & \text{-----------} & KOB^g_{1K_g} \\ \vdots & & \vdots \\ KOB^g_{K_g 1} & \text{---------------} & KOB^g_{K_g K_g} \end{bmatrix} , \qquad (24)$$

$$\overline{Z}^g = \begin{bmatrix} Z_1^g \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ Z_{K_g}^g \end{bmatrix}, \qquad \overline{Q}^g = \begin{bmatrix} Q_1^g \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ Q_{K_g}^g \end{bmatrix} .$$

Argument z is omitted.

To simplify the notation and to illustrate the principle of the solution technique, let us first look at the simple case where $K_g = 1$. The matrices in eq. (24) are then degenerating into constants, and thus we simply deal with ordinary line difference equations as described in ref. 9; eq. (23) is then simple.

$$-DD^g(z-1) \cdot Z^g(z-1) + KOB^g(z) \cdot Z^g(z) - DD^g(z) \cdot Z^g(z+1) = Q^g(z)$$

$z = 1{:}N$ N = number of mesh point in z-direction,

or in matrix formalism

$$\begin{bmatrix} KOB^g(1) & -DD^g(1) & & 0 \\ -DD^g(1) & KOB^g(2) & -DD^g(2) & \\ \diagdown & \diagdown & \diagdown & \\ & & -DD^g(N-1) & \\ 0 & -DD^g(N-1) & KOB^g(N) & \end{bmatrix} \cdot \begin{bmatrix} Z^g(1) \\ \cdot \\ \cdot \\ \cdot \\ Z^g(N) \end{bmatrix} = \begin{bmatrix} Q^g(1) \\ \cdot \\ \cdot \\ \cdot \\ Q^g(N) \end{bmatrix} \qquad (25)$$

This matrix problem can be solved directly in the following manner. Two auxiliary vectors $W^g(z)$ and $G^g(z)$ are defined as

$$W^g(1) = -\frac{DD^g(1)}{KOB^g(1)} ,$$

$$W^g(z) = \frac{DD^g(z)}{KOB^g(z) + DD^g(z-1) \cdot W^g(z-1)} \qquad 2 \leq z \leq N$$

and

$$G^g(1) = \frac{Q^g(1)}{KOB^g(1)} \quad , \tag{26}$$

$$G^g(z) = \frac{Q^g(z) + DD^g(z-1)\cdot G^g(z-1)}{KOB^g(z) + DD^g(z-1)\cdot W^g(z-1)} \qquad 2 \leq z \leq N \ .$$

The solution vectors $Z^g(z)$ are then given recursively by

$$Z^g(N) = G^g(N) \quad , \tag{27}$$

$$Z^g(z) = G^g(z) - W^g(z)\cdot Z^g(z+1) \ . \qquad 1 \leq z \leq N-1 \ .$$

The solution technique used for the synthesis problem is similar to that of line difference technique only, the elements in the matrix equation are now submatrices.

Eq. (23) is now in matrix formalism

$$\begin{bmatrix} \overline{\overline{AA}}^g(1) & -\overline{\overline{DM}}^g(1) & & 0 \\ -\overline{\overline{DM}}^g(1) & \overline{\overline{AA}}^g(2) & -\overline{\overline{DM}}^g(2) & \\ & \ddots & \ddots & \ddots \\ & & & \overline{\overline{DM}}^g(N-1) \\ 0 & & -\overline{\overline{DM}}^g(N-1) & \overline{\overline{AA}}^g(N) \end{bmatrix} \cdot \begin{bmatrix} \overline{Z}^g(1) \\ \vdots \\ \vdots \\ \overline{Z}^g(N) \end{bmatrix} = \begin{bmatrix} \overline{Q}^g(1) \\ \vdots \\ \vdots \\ \overline{Q}^g(N) \end{bmatrix} \tag{28}$$

Auxiliary quantities equivalent to W and G are introduced. $\overline{G}^g(z)$ vectors and $\overline{\overline{W}}^g(z)$ square matrices and further auxiliary matrices $\overline{\overline{KAS}}^g(z)$ are introduced.

We have

$$\overline{\overline{KAS}}^g(z) = \overline{\overline{AA}}^g(z) + \overline{\overline{DD}}^g(z-1)\cdot\overline{\overline{W}}^g(z-1) \ .$$

The inverted KAS is calculated with a subroutine, INVERT (ref. 12). Equivalent to eq. (26) we have

$$\overline{\overline{W}}^g(z) = -\overline{\overline{KAS}}^g(z)^{-1}\cdot\overline{\overline{DD}}^g(z) \tag{29}$$

$$\overline{G}^g(z) = \overline{\overline{KAS}}^g(z)^{-1} \cdot (\overline{Q}^g(z) + \overline{\overline{DD}}^g(z-1) \cdot \overline{G}^g(z-1)) \quad . \tag{30}$$

It is seen that only the vectors $\overline{G}^g(z)$. contain terms that are altered from iteration to iteration, the source terms $\overline{Q}^g(z)$. It is then possible to precalculate the matrices $\overline{\overline{W}}^g(z)$ and $\overline{\overline{KAS}}^g(z)^{-1}$ once before the iterations are started. It is of great importance for the fastness of the iteration that it is only necessary to compute the $\overline{G}^g(z)$ vectors during each iteration.

By means of a backsubstitution like eq. (27) the mixing functions are found.

$$\overline{Z}^g(N) = \overline{G}^g(N) \quad , \tag{31}$$

$$\overline{Z}^g(z) = \overline{G}^g(z) - \overline{\overline{W}}^g(z) \cdot \overline{Z}^g(z+1) \quad . \qquad 1 \le z \le N-1 \quad .$$

After each iteration the eigenvalue k_{eff}, the effective multiplication factor, is found by an over-all neutron balance equation.

$$k_{eff} = \text{production}/(\text{absorption} + \text{leakage}) \quad .$$

3.4. Convergence Acceleration Technique

To accelerate the convergence of the iteration we introduce an extrapolation technique based on the largest eigenvalue of the error matrix. We have

$\overline{Z}$ = the exact solution vector

$\overline{Z}_n$ = the solution vector at the nth iteration.

The deviation from iteration to iteration is coupled by a matrix equation.

$$\overline{Z}_{n+1} - \overline{Z}_n = \overline{\overline{A}} \cdot (\overline{Z}_n - \overline{Z}_{n-1})$$

$\overline{\overline{A}}$ is the error matrix of the system, μ its largest eigenvalue. μ must be less than 1 if the iteration is to be convergent. After some iterations the largest eigenvalue dominates, and we have

$$\overline{Z}_{n+1} - \overline{Z}_n = \mu \cdot (\overline{Z}_n - \overline{Z}_{n-1}) \quad . \tag{32}$$

Let us set up the error vector $\overline{Z} - \overline{Z}_n$ for the nth iteration. The error vector is expanded after the deviation vectors.

$$\begin{aligned}\overline{Z} - \overline{Z}_n &= (\overline{Z}_{n+1} - \overline{Z}_n) + (\overline{Z}_{n+2} - \overline{Z}_{n-1}) + (\overline{Z}_{n+3} - \overline{Z}_{n+2}) \dots \\ &= (\overline{Z}_{n+1} - \overline{Z}_n) + \mu(\overline{Z}_{n+1} - \overline{Z}_n) + \mu^2(\overline{Z}_{n+1} - \overline{Z}_n) \dots \\ &= (\mu + \mu^2 + \mu^3 + \mu^2 \dots\dots) \div (\overline{Z}_n - \overline{Z}_{n-1}) .\end{aligned}$$

$$\overline{Z} = \overline{Z}_n + \frac{\mu}{1-\mu} \cdot (\overline{Z}_n - \overline{Z}_{n-1}) \qquad (33)$$

After each iteration μ is calculated from the norm of the deviation vector.

$$\text{Ress} = | \overline{Z}_{n-1} - \overline{Z}_{n-2} | = \sqrt{\sum_{i=1}^{N} (Z_{n-1}(i) - Z_{n-2}(i))^2} ,$$

$$\text{Res} = | \overline{Z}_n - \overline{Z}_{n-1} | = \sqrt{\sum_{i=1}^{N} (Z_n(i) - Z_{n-1}(i))^2} ,$$

$$\mu = \text{Res}/\text{Ress} . \qquad (34)$$

When μ has been converged to a certain extent, an extrapolation by means of eq. (33) is made, supplying a guess at the solution vector $\overline{Z}$.

For good utilization of this extrapolation technique it is necessary to take into account the sign of the error eigenvalue μ. This is done by comparing the sign of the largest deviation term with the sign of the deviation term at the same point at the next iteration. If these signs are equal, μ is selected positive, if not, negative. If μ is positive, eq. (33) represents an extrapolation. If μ is negative, eq. (33) actually represents an interpolation.

3.5. Three-Dimensional Flux and Power Shapes

When the mixing functions are found, it is easy to calculate the three-dimensional flux distribution from eq. (3).

$$\emptyset^{g}(z,y,x) = \sum_{k=1}^{K_g} Z_k^g(z) \cdot H_k^g(y,x) \quad .$$

The power density distribution is then simply found as

$$\mathrm{POW}(z,y,x) = \sum_{g=1}^{GR} \Sigma_f^g \cdot \emptyset^g = \sum_{g=1}^{GR} \sum_{k=1}^{K_g} \Sigma_f^g \cdot Z_k^g(z) \cdot H_k^g(y,x) \quad . \qquad (35)$$

3. 6. Calculation of Trial and Weighting Functions

So far the trial functions have been treated as a known set of precalculated two-dimensional flux shapes. In this section a brief description of a two-dimensional difference equation subroutine called DIFFERENS will be given. The subroutine was constructed to make the programme self-sufficient with trial functions, i. e. to make it possible to calculate the necessary trial and weighting functions within the programme.

The subroutine performs two-dimensional x-y multi-group flux calculations by ordinary difference approximation technique. The diffusion equations are transformed into difference equations in the same fashion as described in section (2. 5). An iterative solution technique based on line relaxation technique is used. For further acceleration of the convergence an extrapolation technique similar to that described in section (3. 1) is used.

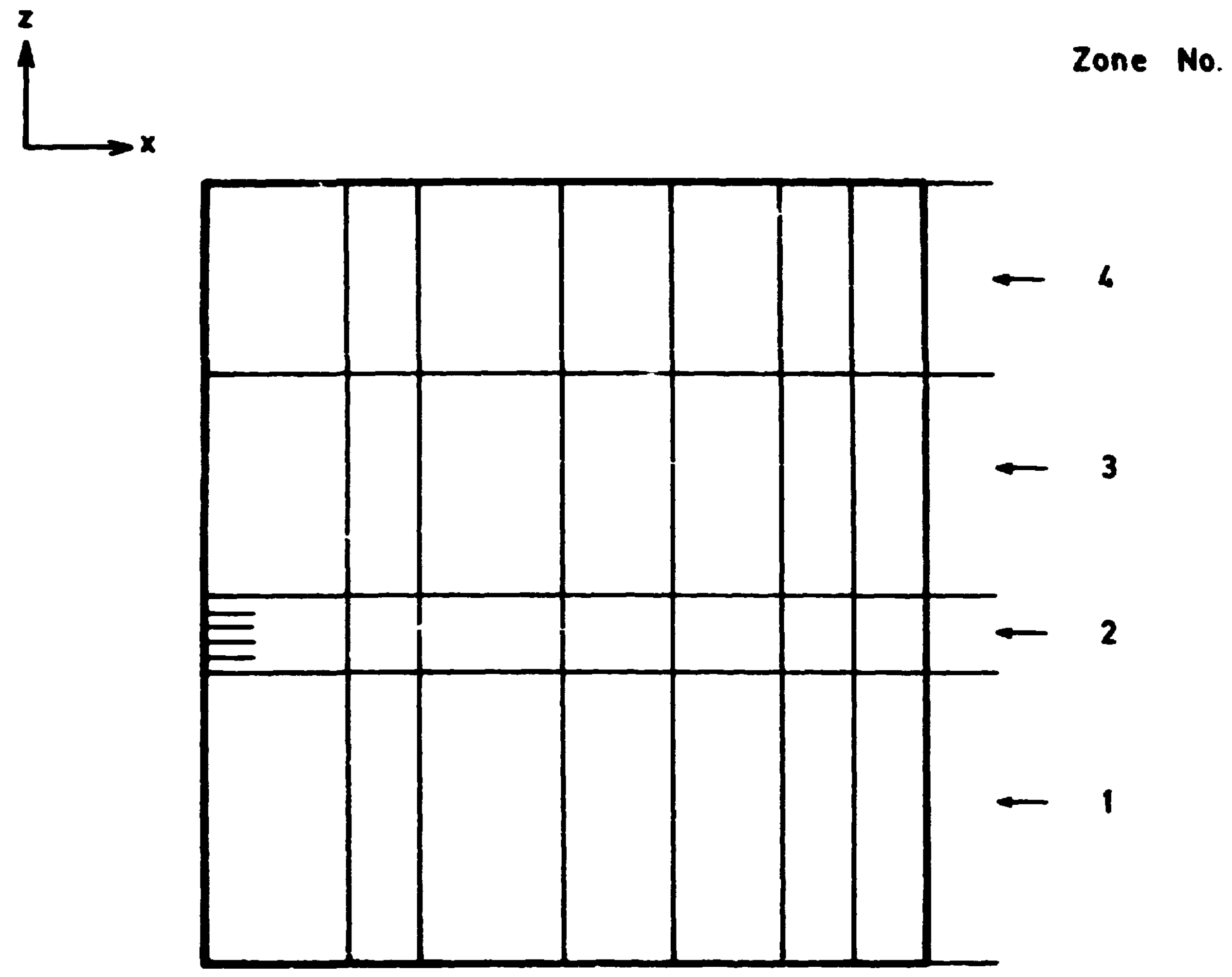
No detailed description of this routine will be given here because the purpose of this report is to describe flux synthesis and not two-dimensional difference equation technique.

4. SUMMARY

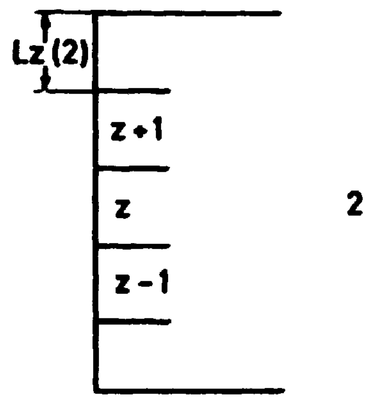
The purpose of this report has been to present the mathematical methods used in the synthesis programme SYNTRON. A sample problem was calculated through by means of SYNTRON and by means of an ordinary three-dimensional difference code WHIRLAWAY (appendix II). This example is not intended to be a verification of the code, but just an illustration. It is not possible to verify an approximate code by comparing the flux point by point with a more exact code, because no definite importance can be attached to a 5% deviation of the flux at a certain point. Not until the code is incorporated in a greater system including burn-up, void and so on will it be possible to make a real test of it by comparing more meaningful quantities with the results of measurements and more accurate calculations.

Fig. 1

Axial zone division



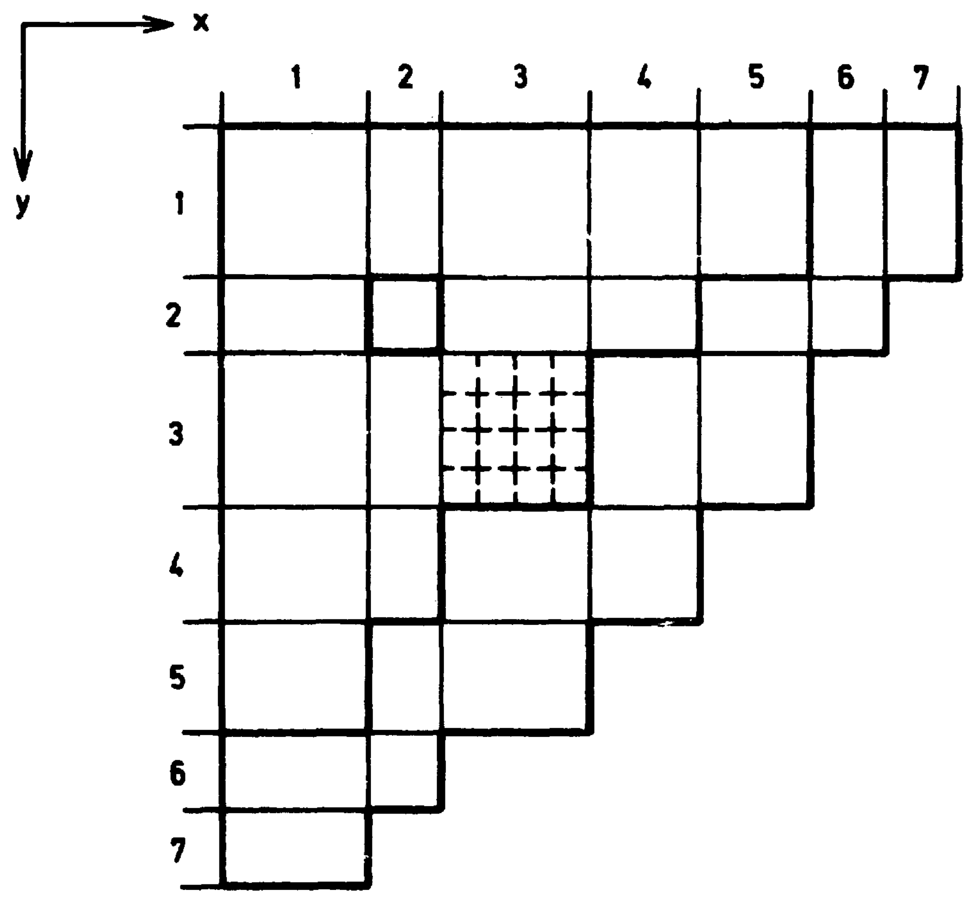
Fine mesh structure in axial zone no. 2



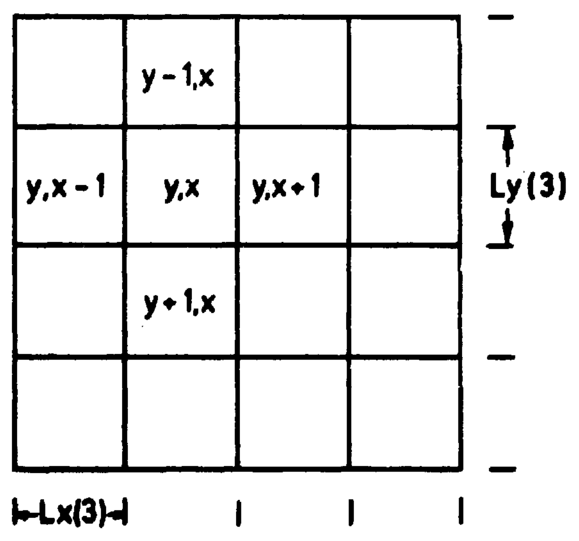
Lz(2) = The length of a fine mesh in axial zone no. 2

Fig. 2

Coarse and fine mesh division of a quarter of a reactor



Fine mesh structure of coarse mesh (3,3)

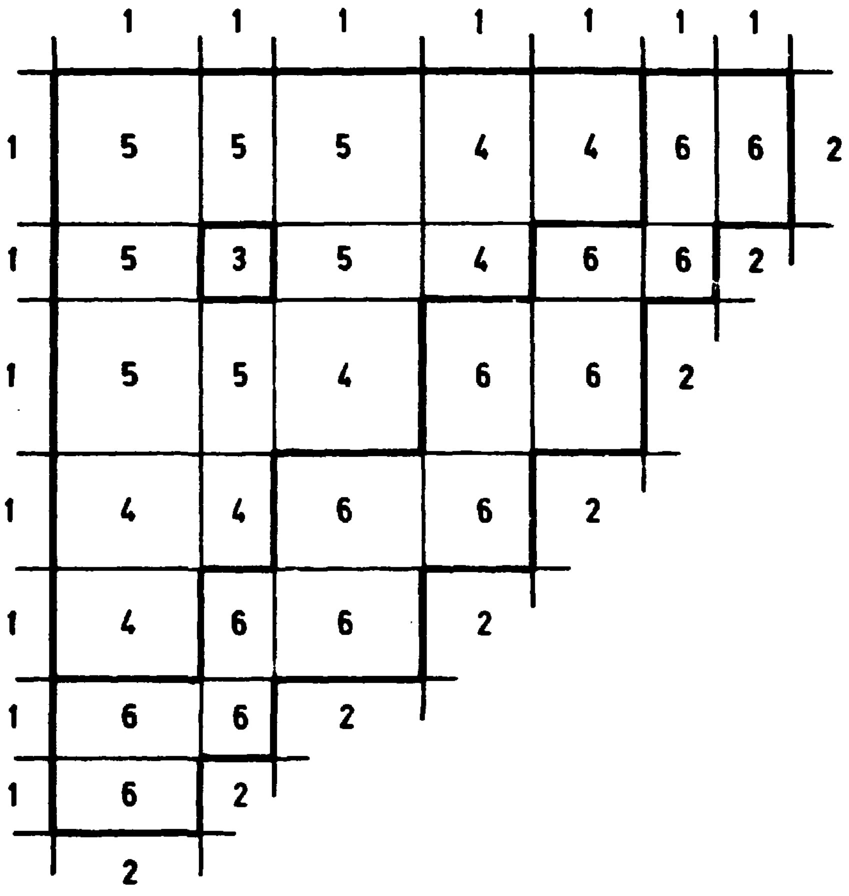


Lx(3) = The x-length of a fine mesh in coarse mesh (3,3)

Ly(3) = The y-length of a fine mesh in coarse mesh (3,3)

Fig. 3

Composition number distribution in axial zone no. k



1 : 3	Boundary conditions
4 : 6	Cross section representation

1	Symmetry boundary
2	Zero flux
3	Control rod, gamma matrix representation
4	Fuel
5	Fuel
6	Reflector

REFERENCES

1) J. E. Meyer, Synthesis of Three-Dimensional Power Shapes - A Flux-Weighting Synthesis Technique. Proceedings of the 2nd United Nations International Conference on the Peaceful Uses of Atomic Energy, Geneva, 1-3 September 1958, 4 (IAEA, Vienna, 1958) 519-22.

2) E. L. Wachspress et al., Multichannel Flux Synthesis, Nucl. Sci. Eng. 12 (1962) 381-389.

3) S. Kaplan, Some New Methods of Flux Synthesis, Nucl. Sci. Eng. 13 (1962) 22-31.

4) J. B. Yasinsky and S. Kaplan, Synthesis of Three-Dimensional Flux Shapes Using Discontinuous Sets of Trial Functions, Nucl. Sci. Eng. 28 (1967) 426-437.

5) M. L. Steele, Variational Techniques as a Method for Multidimensional Reactor Calculations, Reactor Technology 13 (1970) 73-95.

6) L. V. Kantorovich and V. I. Krylov, Approximate Methods of Higher Analysis (P. Noordhoff Ltd., Groningen, 1958) 258-262.

7) S. Kaplan et al., Equations and Programs for Solutions of the Neutron Group Diffusion Equations by Synthesis Approximations, WAPD-TM-377 (1963). 67 pp.

8) E. L. Wachspress et al., Variational Multichannel Synthesis with Discontinuous Trial Functions, KAPL-3095 (1965) 55 pp.

9) R. S. Varga, Matrix Iterative Analysis (Prentice-Hall, Inc., Englewood Cliffs, N. J., 1962) 194-196.

10) A. Hassitt, A Computer Program to Solve the Multigroup Diffusion Equations, TRG Report 229 (R) (1962). 39 pp.

11) J. Pedersen, Calculation of Heterogeneous Constants for Cylinders and Slabs, Risø-M-250 (1969) 23 pp.

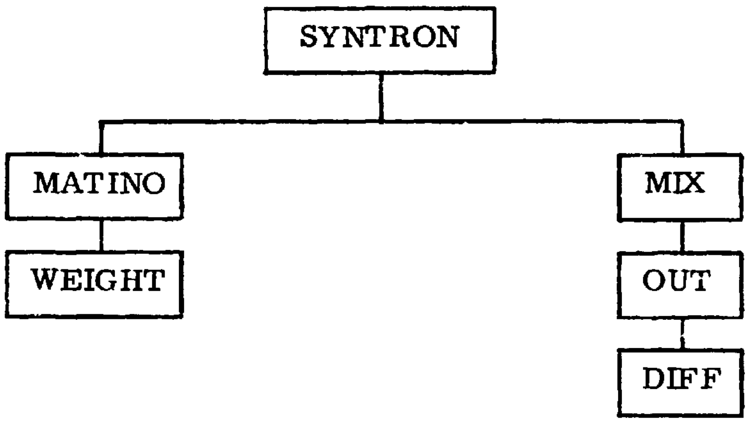
12) B. Molbjerg, INVERT, Danish Atomic Energy Commission, Risö, SA 70 (1965) 2 pp.

13) T. B. Fowler and M. L. Tobias, Whirlaway - A Three-Dimensional, Two-Group Neutron Diffusion Code for the IBM 7090 Computer, ORNL-3150 (1961) 30 pp.

APPENDIX I

The SYNTRON code is written in ILLINOIS ALGOL for the IBM 7094 computer at NEUCC. The code makes use of two overlay tapes and three data storing tapes.

The present version of the programme consists of a main programme, SYNTRON, and five procedures, MATINO, WEIGHT, MIX, OUT, and DIFF, on overlays.



MATINO	input routine
WEIGHT	cross section integration and coefficient matrix preparation routine
MIX	solution routine, calculates the mixing functions and thc eigenvalue k_{eff}.
OUT	output routine, calculates the three-dimensional flux and power distribution
DIFF	this routine computes the two-dimensional trial and weighting functions.

Input Preparation for SYNTRON

FF

...., ..., ..., ..., problem no., day, month, year

...., AS = number of compositions represented as cross sections + number of boundaries

...., NG = number of boundaries, i.e. gamma matrices

...., GR = number of groups

...., ..., ..., CX, CY, CZ = number of coarse mesh in x, y and z direction

...., ..., ..., ..., N [1:GR] = number of trial functions in each group

...., ..., ..., ..., CMX [1:CX] = the length of each coarse mesh in x-direction

...., ..., ..., ..., FMX [1:CX] = number of fine mesh in each coarse mesh in x-direction

...., ..., ..., ..., CMY [1:CY]

...., ..., ..., ..., FMY [1: CY]

...., ..., ..., ..., CMZ [1:CZ]

...., ..., ..., ..., FMZ [1:CZ]

...., EXTUP = boundary composition no. at the top of the reactor

```
                 ⟨0:CX + 1 ⟩

      ^      ...., ...., ...., ..
0:CY + 1     ...., ...., ...., ..     SAM [1:CZ, 0:CY + 1, 0:CX + 1 ] =
      v      ...., ...., ...., ..     coarse mesh composition no. specifi-
                                      cation in the three-dimensional reactor,
                                      surrounded by boundary composition
             ...., ...., ...., ..     numbers. Inside the reactor arbitrary
             ...., ...., ...., ..     mixing of cross section compositions
             ...., ...., ...., ..     and boundary conditions are allowed.


             ...., ...., ...., ..
             ...., ...., ...., ..
             ...., ...., ...., ..
```

...., EXTBOT = boundary composition no. at the bottom of the reactor

```
                 ⟨1:GR ⟩

   ^         ...., ...., ...., ..
 1:GR        ...., ...., ...., ..     GAM [1:NG, 1:GR, 1:GR] =
   v         ...., ...., ...., ..     gamma matrices; if only the diagonal
                                      elements are non-zero, the programme
                                      will only take into account the diagonal
             ...., ...., ...., ..     elements and treat them as ordinary
             ...., ...., ...., ..     extrapolation factors.
             ...., ...., ...., ..
```

```
              D          AB         NFIS        XI         FIS    ┐
     ^      ....,      ....,      ....,      ....,      ....,     │
   1:GR     ....,      ....,      ....,      ....,      ....,     │ 1 + NG:AS
     v      ....,      ....,      ....,      ....,      ....,     │
                                                                  │
                     ⟨1:GR ⟩                                      │
                                                                  │
     ^      ....,      ....,      ....,                           │
 SP:1:GR    ....,      ....,      ....,                           │
     v      ....,      ....,      ....,                           │
                                                                  ┘
```

D	=	diffusion constant,
AB	=	macroscopic absorption cross section,
NFIS	=	$\nu\cdot$ macroscopic fission cross section,
XI	=	fission spectrum,
FIS	=	macroscopic fission cross section,
SP	=	macroscopic scattering matrix.

...., NTFC = number of two-dimensional difference equation calculations for trial functions preserved in a function library on tape.

...., WEIGHT if 1, only the ordinary flux is calculated, otherwise both the ordinary flux and the adjoint are calculated.

Input for the specified number of trial function calculations.

```
                      ⟨ 0:CX + 1 ⟩                                        ┐
     ^      ....,      ....,      .......                                 │
0:CY + 1    ....,      ....,      .......   COMP [0:CY+ 1, 0:CX+ 1]       │
     v      ....,      ....,      .......                                 │ 1:NTF
                                                                          │
            ....,      MAXI       ...., TOL       ...., OVR               │
            ....,      STFLUX                                             │
                                                                          ┘
```

COMP	=	two-dimensional composition no. distribution for difference calculation,
MAXI	=	maximum number of iterations allowed,
TOL	=	convergence criterion (suggestion 10^{-4}),
OVR	=	start value of the overrelaxation factor (suggestion 1.2),

STFLUX = flux start parameter; if 1, all flux points are set equal to 1, otherwise the previous flux distribution is used as input.

The programme will calculate the specified number of two-dimensional flux functions and preserve them on tapes.

...., NTRR = number of flux readings from tape for trial function generation.

...., WEIGHT if 1, trial and weighting functions are equal, otherwise the trial functions are equal to the ordinary fluxes, and the weighting functions are equal to the adjoint fluxes.

...., ..., ..., .. TRRNO 1:NTRR
the numbers on the trial functions in the function library which are used

...., ..., ..., .. NTF 1:NTRR
number of trial functions out of each library function used.

```
              GTR          TRI          GRFL
           ┌
           │  ....,        ....,        ....,  ┐
           │                                   │  1:NTF  [1]
           │  ....,        ....,        ....,  ┘
  1:NTRR   │
           │  ....,        ....,        ....,  ┐
           │  ....,        ....,        ....,  │  1:NTF  [2]
           │  ....,        ....,        ....,  ┘
           └
```

Mixing of library functions for trial functions.

GRTR = group no. in trial functions,

TRI = function no. in trial functions,

GRFL = group no. in library functions,

...., MAXI = maximum number of iterations calculation,

...., TOL = convergence criterion in synthesis calculation,

...., POWOUT (if 0, only power mean value calculated,
1, only power distribution on binary tape,
2, power on tape + printing,

3, power on tape + printing + flux printing).

FF

-1 after the last problem.

APPENDIX II

For illustration of the code a sample problem was calculated by means of SYNTRON and by means of an ordinary three-dimensional difference code, WHIRLAWAY (ref. 13).

The sample reactor is a two-zone reactor. For cross sections homogenized Yankee cell cross sections in two energy groups are used. In the centre old fuel, A, on the outside fresh fuel, B; the whole reactor is surrounded by a light-water reflector, C. Cross sections are found in table I. Reactor dimensions and mesh divisions are illustrated in fig. 1.

In figs. 2 and 3 the fast and thermal fluxes are depicted. The flux distribution is illustrated by means of the flux distribution in the z-direction at some selected (y, x) points.

WHIRLAWAY calculates the flux in the corner of each mesh, whereas SYNTRON takes a flux point in the middle of the mesh. For the results to be comparable, linear interpolation between the four neighbouring mesh points in the yx plane was used.

For this problem two trial functions were used in each energy group in SYNTRON. The trial functions are found by two-dimensional calculations on the reactor, one performed at axial zone no. 1 $0 < z < 60$ cm and one at axial zone no. 2 $60 < z < 80$ cm. The fast fluxes are used as fast trial functions and the thermal fluxes as thermal trial functions.

Problem size $15 \times 15 \times 15$ mesh, two energy groups. The size of the sample problem is determined by the capacity of WHIRLAWAY, which is an old and rather slow code in only two energy groups.

Computing times for the sample problem

WHIRLAWAY total	25.60 min.
SYNTRON:	
DIFFERENCE 1	0.459 min.
- " - 2	0.476 min.
MATINO	0.036 min.
WEIGHT	0.115 min.
MIX	0.230 min.
OUT	0.660 min.
SYNTRON total	2.00 min.

If the computing time for trial function calculations is neglected and no print-out of the flux is wanted, the total SYNTRON computing time will be about 0.6 min.

Table I

Cross sections for sample problems

	D_1	D_2	Σ_{a1}	Σ_{a2}	$\gamma\Sigma_{f1}$	$\gamma\Sigma_{f2}$	X_{i1}	X_{i2}
	cm	cm	cm^{-1}	cm^{-1}	cm^{-1}	cm^{-1}		
A	1.2894	0.3915	0.01159	0.12124	0.00716	0.16575	1	0
B	1.2866	0.3966	0.01164	0.09916	0.00779	0.1502	1	0
C	1.7290	0.2392	0.00055	0.0111	0	0	1	0

Fig. 1

Sample Problem

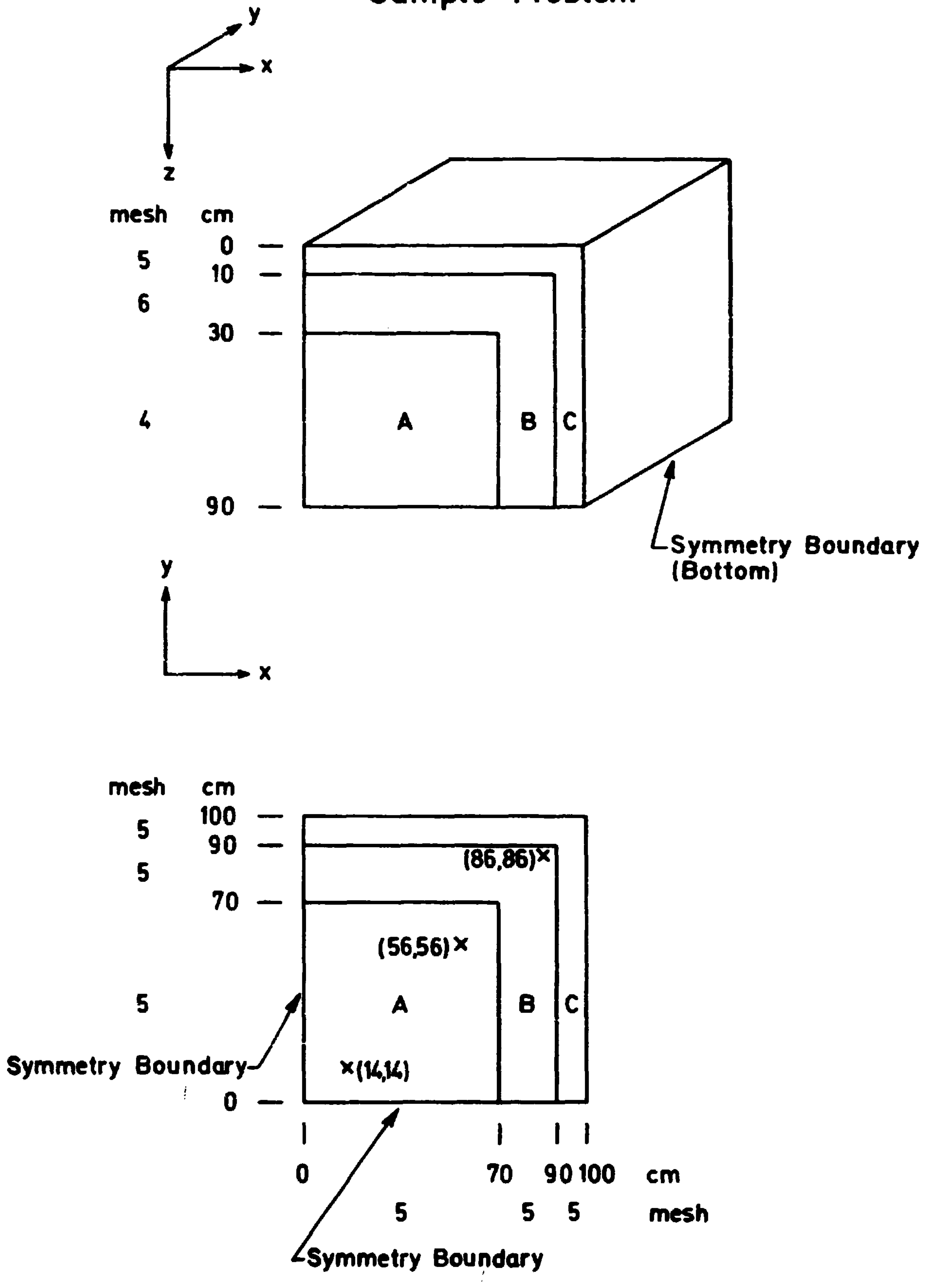


Fig. 2

FAST FLUX

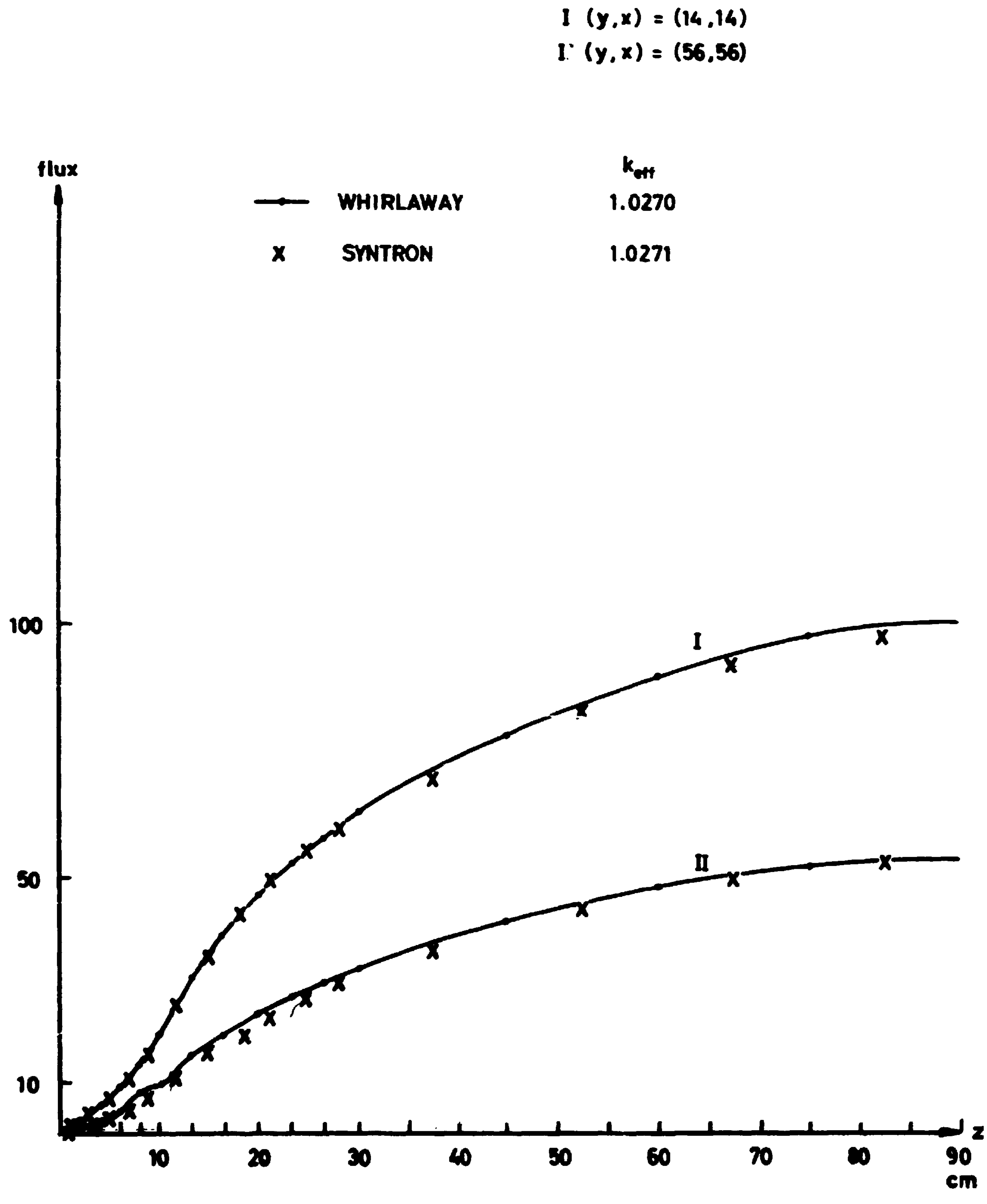
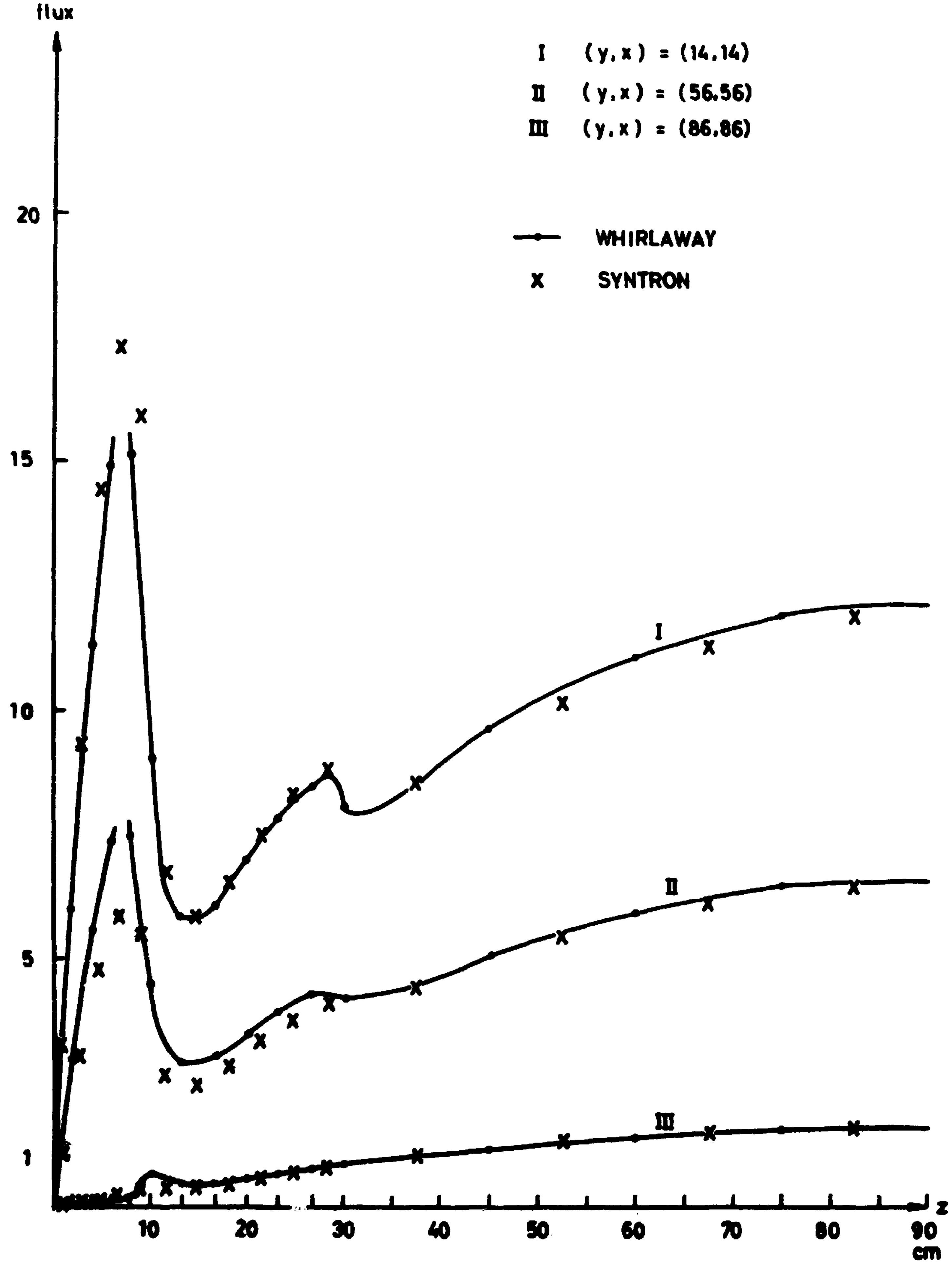


Fig.3

THERMAL FLUX



IBM

IBM Data Centre
Punching Instructio

NAME Hans Larsen	APPLICATION Sample Problem for SYNTRON
Date 12/1- 1977 · Page 1 of	

```
1        10        20        30        40        50        60        70
FF
1,12,1,1977
5,2,2
3,3,3
2,2
70,20,10
5,5,5
70,20,10
5,5,5
10,20,60
5,6,4
1
2,2,2,2,2
2,5,5,5,1
2,5,5,5,1
2,5,5,5,1
2,1,1,1,1
2,2,2,2,2
2,4,4,5,1
2,4,4,5,1
2,5,5,5,1
2,1,1,1,1
2,2,2,2,2
2,3,4,5,1
```

D 2023

IBM

IBM Data Centre
Punching Instruction

NAME *Hans Larsen*

Date *12/1 - 1971* Page *2* of

APPLICATION *Sample Problem for SYNTRON*

```
2,4,4,5,1
2,5,5,5,1
2,1,1,1,1
2
'9,0,0,'9
'-9,0,0,'-9
1.2994,0.01159,0.00762,1,0.0045
0.3975,0.12124,0.16575,0,0.0700
0,0
0.075,0
1.2866,0.01164,0.00779,1,0.0950
0.3966,0.09916,0.15020,0,0.0700
0,0
0.015,0
1.7290,0.00355,0,1,0
0.2392,0.01110,0,0,0
0,0
0.0398,0
2,1
2,2,2,2,2
2,4,4,5,1
2,4,4,5,1
2,5,5,5,1
2,1,1,1,1
```

D 2023

IBM

IBM Data Centre
Punching Instructi

NAME *Hans Larsen*
Date *12/1-1971* Page *3* of
APPLICATION *Sample Problem for SYNTRON*

```
1        10        20        30        40        50        60        70
40,'-4,1.3,1
2,2,2,2,2
2,3,4,5,1
2,4,4,5,1
2,5,5,5,1
2,1,1,1,1
49,'-4,1-3,1
2,1
1,2
2,2
1,1,1
2,1,2
1,2,1
2,2,2
50,'-4
0
FF
-1
```

D 2023