November 1973 WRRI Report No. 033

A CONJUNCTIVE USE SURFACE WATER-GROUND WATER SIMULATOR

Partial Technical Completion Report
Project No. A-045-NMEX

A CONJUNCTIVE USE SURFACE WATER-GROUND WATER SIMULATOR

W. Brutsaert, *Former Assistant Professor*
Geoscience Department

C. Way, *Research Assistant*
Geoscience Department

PARTIAL TECHNICAL COMPLETION REPORT
Project No. A-045-NMEX

New Mexico Water Resources Research Institute
in cooperation with
Geoscience Department, New Mexico Institute of Mining and Technology
Socorro, New Mexico 87801

November 1973

The work upon which this publication is based was supported in part by funds provided through the New Mexico Water Resources Research Institute by the United States Department of Interior, Office of Water Resources Research, as authorized under the Water Resources Research Act of 1964, Public Law 88-379, under project number: A-045-NMEX.

ABSTRACT

As part of an interdisciplinary analysis of the economic evaluation of the water resources of the Rio Grande Region of New Mexico, a conjunctive use surface water-ground water simulator was developed. The simulator was used to determine the dynamic availability of ground water and its relationship to flow in the Rio Grande.

This report gives an account of the structure and use of the simulator. It is a computer model in which the governing nonlinear parabolic partial differential equation, used to simulate unconfined aquifer flow in two dimensions, was solved by implicit finite differences. The set of difference equations in residual form are solved by LSOR or Gauss Elimination depending upon the size of the problem.

ACKNOWLEDGEMENTS

This study was part of an interdisciplinary-interuniversity research project entitled "An Analytical Interdisciplinary Evaluation of the Utilization of the Water Resources of the Rio Grande in New Mexico", conducted under NMWRRI project numbers 3109-108, 3109-111, and 3109-117, further described by OWRR projects B-011, B-019, and B-026-NMEX, through the New Mexico Water Resources Research Institute in cooperation with the Agricultural Experiment Station and Engineering Experiment Station, New Mexico State University; University of New Mexico; and New Mexico Institute of Mining and Technology.

The work upon which this publication is based was supported in part by funds provided through the New Mexico Water Resources Research Institute by the United States Department of Interior, Office of Water Resources Research, as authorized under the Water Resources Research Act of 1964, Public Law 88-379.

The New Mexico Institute of Mining and Technology Computer Center is gratefully acknowledged for a portion of the total computer time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

A CONJUNCTIVE USE SURFACE WATER-GROUND WATER SIMULATOR

by W. Brutsaert and C. Way*

I. GENERAL INTRODUCTION

The complexity of today's water resource systems makes it almost mandatory that their management be achieved through the application of the talents and technology of a variety of disciplines, and the legal and institutional structures of water use make their understanding and application a further requisite to good management. Although economic justification should be the foundation of decisions on alternative uses of water, the social and cultural implications must be fully considered since the optimal use of water implies a maximization of the benefits returned to society through a broad range of beneficial uses. In order to formulate plans and policies for future water resources development, the assessment of future water supplies and requirements necessitates a major consideration of future rates and patterns of economic development.

Before an economic evaluation of water use can be made, the availability of surface water and ground water must be determined. If there is a continual exchange between the surface water of the river and the ground water of the surrounding alluvium, ground water availabilities are not simply related to pumpage but are also controlled by precipitation, amount and frequency of runoff in streams, return of irrigation water, and evapotranspiration. The management of such a system should consequently be based on a conjunctive operation of surface and ground water.

A model of the exchanage of ground water and surface water is necessary if a good description of the pumping effects is desired. In addition, the diversions and depletions must be known in order to have a reliable model.

---

*Formerly Assistant Professor, and Research Assistant, Geoscience Department, New Mexico Institute of Mining and Technology.

For a comprehensive alternative water use analysis it is necessary to know both the ground water availability and the behavior of the aquifer under projected stresses. Since historical records of hydrologic systems, in most instances, are inadequate to permit direct analysis of basin behavior under projected stresses, a ground water system simulator is necessary. The most efficient and practical simulator appears to be a mathematical analogue of the hydrologic basin, solved by digital computer. Such a hydrologic system can be schematically represented as in Figure 1:1.

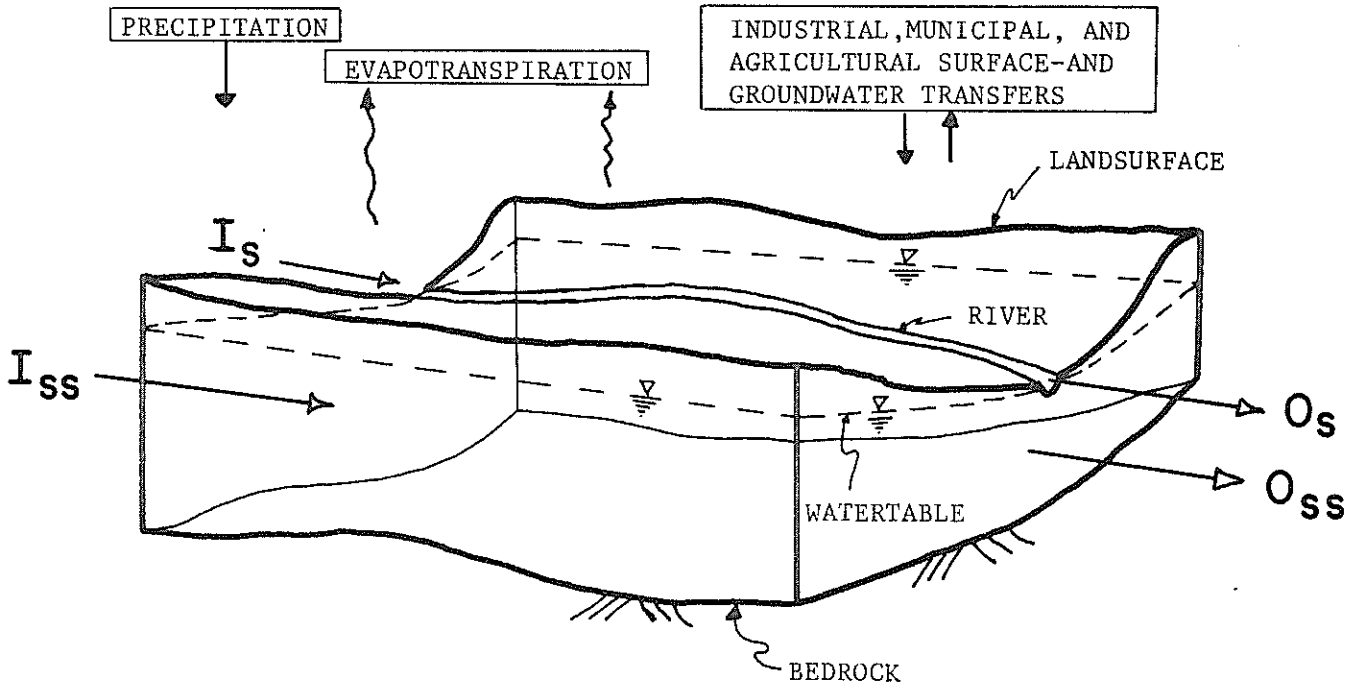


Figure 1:1. Schematic representation of hydrologic system.

In this Figure, I = inflow, O = outflow, and the subscripts s and ss, respectively, stand for surface and subsurface. Figure 1:1 is simply a representation of the statement of continuity: i.e., the conservation of

mass which is the basis for developing the fundamental flow equation or mathematical analogue which is written as:

$$\frac{\partial}{\partial x}\left(Kh\frac{\partial H}{\partial x}\right)+\frac{\partial}{\partial y}\left(Kh\frac{\partial H}{\partial y}\right)=S\frac{\partial H}{\partial t}+\frac{Q}{\delta x\delta y} \qquad (1:1)$$

where K = hydraulic conductivity; h = saturated thickness; H = water table elevation (total head) above datum; S = effective porosity (storage coefficient); Q = net withdrawal rate from aquifer; x, y = space dimensions; t = time dimensions; δx, δy = surface area of element.

The above equation is a nonlinear partial differential equation obtained by combining Darcy's law with the continuity principle and is applicable to transient, two-dimensional flow in heterogeneous, anisotropic, incompressible, unconfined, saturated, porous media.

Implicit finite differences are utilized to solve equation 1:1 on a digital computer. The resulting system of simultaneous equations is either solved directly by Gauss-Elimination or by Line Successive Over Relaxation, depending upon the size of the problem: i.e., number of equations.

The first step when applying the model to a particular study area is to verify or calibrate its behavior. This verification consists of simulating a historical period, for which both ground water levels and stream records are available, until model and prototype match. Checking and updating should be continued as more data become available.

The next step is the simulation (or extrapolation) of future possible conditions. Conditions can be altered to give different responses. Conditions from extreme dry to extreme wet combined with a set of water demands ranging from large to small may be included.

A vast amount of data is thus obtained in the form of aquifer responses for given conditions. Realizing that these data are the result of the solution of a continuity equation of the form

$$I - O = \Delta S/\Delta t \qquad (1:2)$$

it is possible to estimate an analogous relationship from the data obtained. In equation 1:2, I = inflow, O = outflow, and Δ S = change in storage during a time period Δ t. The relationship postulated was of the following form

$$\Delta d = f(d_n, L) \qquad (1:3)$$

where Δd = change in water-table elevation for the time period (year) considered, d_n = water elevation at the end of previous time period (year), and L = a lump factor combining surface water inflow and outflow, precipitation, and beneficial and nonbeneficial water uses.

Results of the simulation runs can be tabulated with averaged spatial responses: i.e., results would not reflect a variation in water-table elevation along lines perpendicular to the river bed. It is assumed that average conditions would suffice since lateral aquifer response would average out.

A stepwise multiple regression analysis, combining linear and nonlinear (exponential and logarithmic) terms of the different factors obtained from the simulation runs, can be performed to obtain the surface-ground water interrelationship equation.

This conjunctive surface-ground water model was developed and applied to the Rio Grande region in New Mexico. A presentation of the results from this application is given in WRRI reports 020-024 (Lansford et al., 1973).

II. TECHNICAL MODEL DESCRIPTION

Adequate simulation of complex ground water systems can be accomplished using digital computer solutions of a mathematical model. This model was developed by using a general mass balance equation and then calibrated by simulating historical conditions. If computed and actual water-table elevations agree within some preset tolerance, then the mathematical model provides a means of predicting the effects of future ground water development.

Ideal boundary conditions and water-bearing characteristics are rarely if ever found in nature, and therefore analytical approaches, when applied to ground water systems, become oversimplified. Most aquifers are heterogeneous and anisotropic, with irregular boundaries. Mathematical models are ideal for handling such natural systems. However, hydrogeologic judgement should be utilized in designing these mathematical models to approximate actual conditions and in properly qualifying the results according to the amount of discrepancy.

Selection of the space and time dimensions (ΔX, ΔY, Δt) is dependent upon the availability of geologic and hydrologic data and upon the desired accuracy and detail of analysis. The accuracy of the solution is enhanced as the dimensions of ΔX, ΔY, and Δt are decreased, providing the availability of geologic and hydrologic data justify the additional computational time required. Space dimensions should be small enough so that the geologic and hydrologic conditions may be assumed reasonably uniform over an entire grid. The maximum size of the time increment, Δt, which will provide adequate accuracy should be used. Solutions of sample problems with representative data can be used to determine the optimum value of Δt for selected grid dimensions. Smaller grid dimensions may require shorter time increments.

A variable $\Delta X, \Delta Y$-gridsystem was utilized in this model in order to adjust the size of the gridblocks to the required detail of analysis.

Continuous and Discretized Mass Balance Equation

The non-linear partial differential equation describing transient, incompressible, isothermal, two-dimensional flow in an unconfined saturated porous medium may be derived from the mass continuity equation and Darcy's Law and can be written as:

$$\frac{\partial}{\partial x}(K_{xx}\ h\ \delta y\ \frac{\partial H}{\partial y})\delta x + \frac{\partial}{\partial y}(K_{yy}\ h\ \delta x\ \frac{\partial H}{\partial y})\delta y = Q + \delta x\ \delta y\ S\ \frac{\partial H}{\partial t} \qquad (2:1)$$

where,

K_{xx} = principal component of hydraulic conductivity tensor (coinciding with x-direction). [L/T]

Kyy = principal component of hydraulic conductivity tensor (coinciding with y-direction). [L/T]

h = saturated thickness of aquifer. [L]

S = storage coefficient. (dimensionless)

Q = net rate of withdrawal. [L^3/T]

$\delta x, \delta y$ = dimensions of differential element.

x,y = space dimensions. [L]

t = time dimension. [T]

H = water table elevation above a datum (potentiometric head). [L]

This equation is valid for heterogeneous and anisotropic media but is subject to the Dupuit-Forchheimer assumptions and should be used with these limitations in mind. The most important limitations are flow in the horizontal x,y-plane, and fully penetrating constant head (river, lake) boundary conditions.

Equation 2:1 has no general analytical solution; therefore, a finite difference approximation is utilized to obtain a numerical solution with the aid of a digital computer. Application of the finite difference approach requires subdivision of the study area into a system of finite grids.

Consider the following two blocks (Figure 2:1) of this grid system:

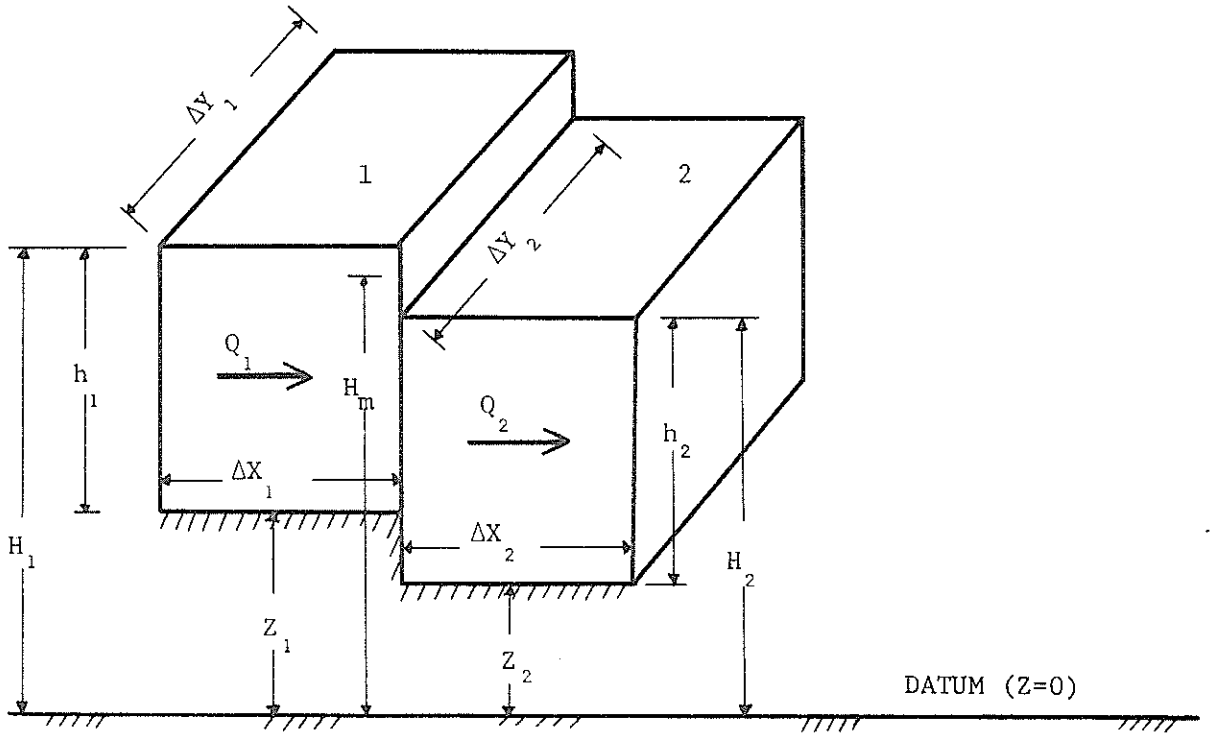


Figure 2:1. Idealization of two adjacent blocks.

According to Darcy's Law:

$$Q_1 = K_1 h_1 \Delta Y_1 \frac{H_m - H_1}{\Delta X_1 / 2} \qquad (2:2)$$

$$Q_2 = K_2 h_2 \Delta Y_2 \frac{H_2 - H_m}{\Delta X_2 / 2} \qquad (2:3)$$

where

H_m = potentiometric head at the interface of block 1 and 2. [L]

Z = bedrock elevation above an arbitrary datum. [L]

Since, by continuity, $Q_1 = Q_2 = Q$, H_m in equations 2:2 and 2:3 can be eliminated to obtain the following relation:

$$Q = \frac{2(K_1 h_1 \Delta Y_1)\ (K_2 h_2 \Delta Y_2)}{K_2 h_2 \Delta Y_2 \Delta X_1 + K_1 h_1 \Delta Y_1 \Delta X_2} (H_2 - H_1) \qquad (2:4)$$

this expression can be approximated by:

$$Q = \frac{2K_1 \Delta Y_1 K_2 \Delta Y_2 \bar{h}_{12}}{K_1 \Delta Y_1 \Delta X_2 + K_2 \Delta Y_2 \Delta X_1} (H_2 - H_1) \qquad (2:5)$$

where,

$$\bar{h}_{12} = \text{MAX}\ (H_2, H_1) - \text{MAX}\ (Z_2, Z_1) \qquad (2:6)$$

MAX (A,B) means the largest of the two quantities A and B.

By using this technique for computing flow coefficients and an implicit central finite difference form, equation 2:1 becomes:

$$\begin{aligned} & C_E H_{i+1,j}^{t+\Delta t} + C_W H_{i-1,j}^{t+\Delta t} + C_S H_{i,j+1}^{t+\Delta t} + C_N H_{i,j-1}^{t+\Delta t} \\ & - (C_E + C_W + C_N + C_S + \frac{S\Delta X\Delta Y}{\Delta t})\ H_{i,j}^{t+\Delta t} \\ = & - (\frac{S\Delta X\Delta Y}{\Delta t})\ H_{i,j}^{t} + Q_{i,j} \end{aligned} \qquad (2:7)$$

C_E, C_W, C_N, C_S are the flow coefficients of the mass balance equation. Subscripts E, W, N, S denote the directions east, west, north, and south: they are used for convenience to replace the more standard but lengthy notation of $i+\frac{1}{2},j$, $i-\frac{1}{2},j$, $i,j-\frac{1}{2}$, $i,j+\frac{1}{2}$, and they reflect properties of two adjacent gridblocks:

$$C_W = \frac{2K_{i-1,j}K_{i,j}\Delta Y_{i-1,j}\Delta Y_{i,j}\bar{h}_W^{\,t}}{\Delta X_{i,j}K_{i-1,j}\Delta Y_{i-1,j} + \Delta X_{i-1,j}K_{i,j}\Delta Y_{i,j}} \qquad (2:8)$$

$$C_E = \frac{2K_{i,j}K_{i+1,j}\Delta Y_{i,j}\Delta Y_{i+1,j}\bar{h}_E^t}{\Delta X_{i+1,j}K_{i,j}\Delta Y_{i,j} + \Delta X_{i,j}K_{i+1,j}\Delta Y_{i+1,j}}$$

$$C_N = \frac{2K_{i,j-1}K_{i,j}\Delta X_{i,j-1}\Delta X_{i,j}\bar{h}_N^t}{\Delta Y_{i,j}K_{i,j-1}\Delta X_{i,j-1} + \Delta Y_{i,j-1}K_{i,j}\Delta X_{i,j}} \qquad (2:8)\ \text{Cont'd.}$$

$$C_S = \frac{2K_{i,j+1}K_{i,j}\Delta X_{i,j+1}\Delta X_{i,j}\bar{h}_S^t}{\Delta Y_{i,j}K_{i,j+1}\Delta X_{i,j+1} + \Delta Y_{i,j+1}K_{i,j}\Delta X_{i,j}}$$

The i,j notation (Figure 2:2) refers to the grid block at which a particular equation is formulated, and the superscripts represent the time level of computation.

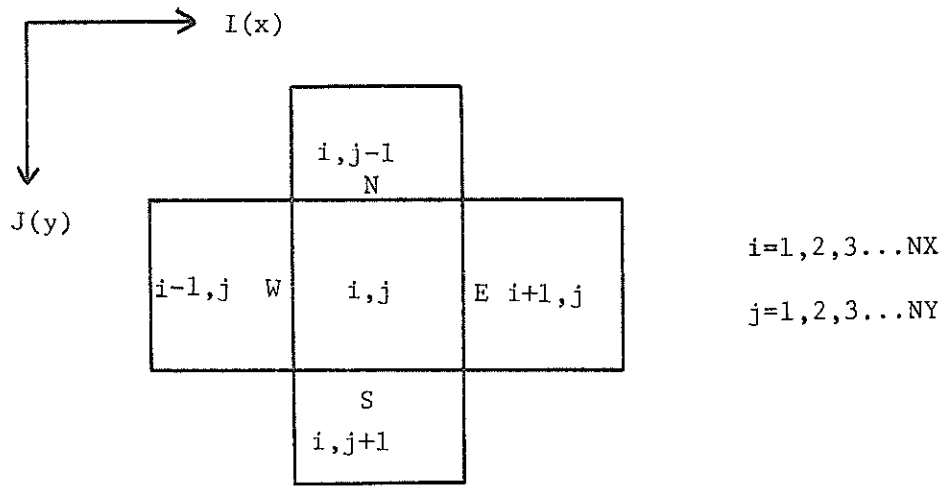


Figure 2:2. Numbering system for i,j notation.

In Figure 2:2, I and J denote block numbers in the coordinate system (X,Y), and i and j are subscripts used to identify grid blocks. NX and NY are numbers of blocks in x and y directions.

Let

$$C_E + C_W + C_N + C_S + \frac{S\Delta X\Delta Y}{\Delta t} = A$$

$$-\left(\frac{S\Delta X\Delta Y}{\Delta t}\right) H_{i,j}^t + Q_{i,j} = -B \qquad (2:9)$$

Equation 2:7 then becomes:

$$A_{i,j}H_{i,j}^{t+\Delta t} - C_E H_{i+1,j}^{t+\Delta t} - C_W H_{i-1,j}^{t+\Delta t} - C_N H_{i,j-1}^{t+\Delta t} - C_S H_{i,j+1}^{t+\Delta t} = B_{i,j} \quad (2:10)$$

To save computer time, single instead of double dimensioned arrays were used. To understand the relationship between single (ij) and double (i,j) subscripted variables, compare Figure 2:2 with Figure 2:3. Equation 2:10 written with ij notation (Figure 2:3) becomes:

$$A_{ij}H_{ij}^{t+\Delta t} - C_E H_{ij+1}^{t+\Delta t} - C_W H_{ij-1}^{t+\Delta t} - C_N H_{ij-NX}^{t+\Delta t} - C_S H_{ij+NX}^{t+\Delta t} = B_{ij} \quad (2:11)$$

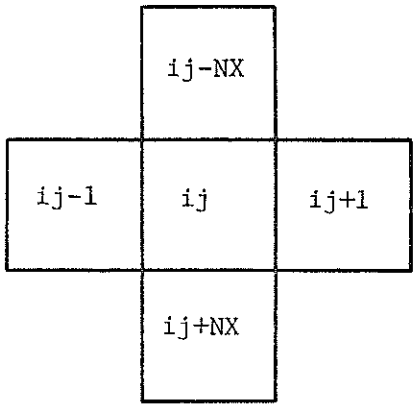


Figure 2:3. Numbering system for ij notation.

It should be noted that the coefficients C_E, C_W, C_N, C_S and the net rate of withdrawal Q are held constant during each time step.

Equation 2:11 is written for each grid for a particular time step and the entire system of equations thus obtained is solved simultaneously. The solution will be the predicted value of H, the water-table elevation, at the end of the time step. This predicted value will be used as the initial value for the next time step, and the entire process is repeated.

III. METHODS OF SOLUTION

Residual Approach. The residual approach as used in this program is simply an aid to improve accuracy through single precision calculations for the solution of systems of equations. IBM SYSTEM 360, MOD 44, with its 6 to 7 digit accuracy, makes this approach obligatory.

Equation 2:11 can be expressed in matrix notation as follows:

$$[D]\ [H]^{t+\Delta t} = [RHS] \tag{3:1}$$

where

$[D]$ = the coefficient matrix

$[H]^{t+\Delta t}$ = water elevation vector at time $t+\Delta t$ (variable)

$[RHS]$ = right-hand side constant vector

Define

$$H^{t+\Delta t} = H^{k} + H^{*} \tag{3:2}$$

where

H^{k} = the computed value for k-th iteration

H^{*} = the correction between the computed value and the actual result.

In matrix form equation 3:2 can be written as:

$$[H]^{t+\Delta t} = [H]^{k} + [H]^{*} \tag{3:3}$$

$[H]^{*}$ is then called the correction vector. Substituting 3:3 into 3:1 produces:

$$[D]\ [H]^{*} = [r] \tag{3:4}$$

where $[r] = [RHS] - [D]\ [H]^{k}$, and is called residual vector.

Instead of equation 3:1, equation 3:4 is solved. The standard procedure is as follows:

(1) Calculate the residual vector $[r]$ in double precision (using local variables).

(2) Solve for the correction vector $[H]^{*}$ in single precision.

(3) Solve for $[H]^{t+\Delta t} = [H]^{k} + [H]^{*}$ in double precision (using local variables).

If the desired accuracy is not obtained, new residuals are computed and the entire process is repeated until equation 3:1 is satisfied (i.e., until residuals have vanished).

The obvious advantage of the residual approach is that a solution can be made accurate beyond the normal number of single precision digits, and extreme accuracy can be obtained when required.

Gaussian Elimination

Gaussian Elimination is a direct method for solving sets of linear algebraic equations. In general, this method tends to be more effective than iterative methods when:

(1) the matrix of coefficients is too complex for a rapid estimate of the optimum over-relaxation factor, or

(2) the matrix is almost singular in that small residuals do not imply small errors in the solution, or

(3) several sets of equations with the same coefficients but different constants have to be solved, or

(4) one-dimensional or two-dimensional problems with relatively small number of grids are solved.

The disadvantages of Gaussian Elimination are that round-off error may affect accuracy seriously; that it is time consuming for two dimensional problems; and that storage requirements increase almost exponentially with the size of the grid system.

In a two-dimensional problem each equation (2:11) consists of five unknowns (H_{ij}, H_{ij-1}, H_{ij+1}, H_{ij-NX}, H_{ij+NX}) (see Fig. 2:3). In a block system with m rows and n columns, there will be m x n equations with m x n unknowns. Therefore, in equation 3:1, [D] is a coefficient matrix with the size of m x n by m x n.

As an example, for a 3 x 3 block system (Fig. 3:1), m = NX = 3, n = NY = 3.

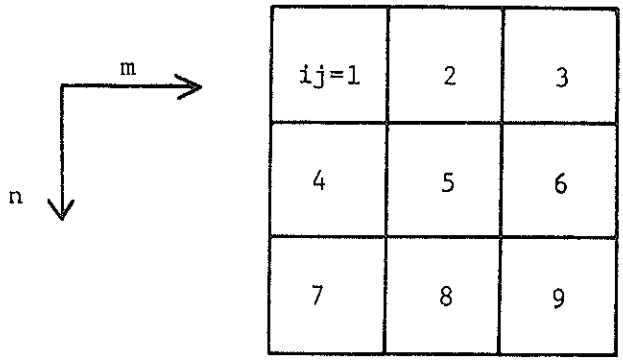


Figure 3:1. Simple 3 x 3 grid system.

For this special case, the coefficient matrix is a square matrix with the size 9 x 9. (Fig. 3:2).

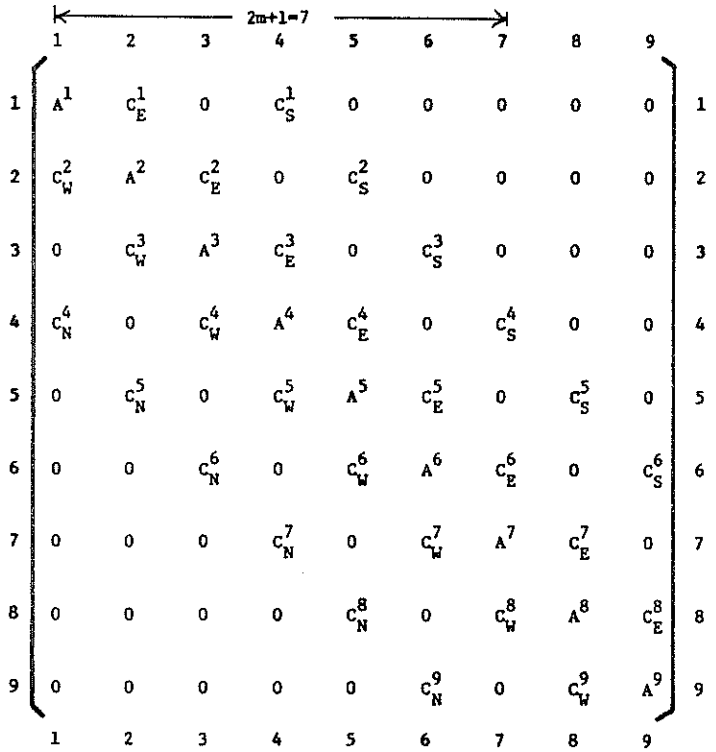


Figure 3:2. Coefficient matrix [D].

In Figure 3:2, superscripts denote the block numbers, subscripts denote the directions. The coefficient matrix has values only on five diagonals: i.e., the main diagonal, two adjacent diagonals, and two outer diagonals.

The distance (in terms of matrix columns) between the outer diagonals is the distance between C_N and C_S. This can be calculated by using a simple relation: i.e., distance = 2m + 1. In this case it is 2 x 3 + 1 = 7.

This matrix is a sparse five diagonal matrix, and it is obvious that much storage would be wasted if all zero values were stored in computer memory. How to transform this matrix to a more condensed form in order to save storage is the basis for "band algorithm" (Rosenberg, 1969; Thurnau, 1963). This technique rotates the matrix to the form shown in Figure 3:3. The size of this new matrix is m x n by 2m + 1.

	1	2	3	4	5	6	7	
1	0	0	0	A^1	C_E^1	0	C_S^1	1
2	0	0	C_W^2	A^2	C_E^2	0	C_S^2	2
3	0	0	C_W^3	A^3	C_E^3	0	C_S^3	3
4	C_N^4	0	C_W^4	A^4	C_E^4	0	C_S^4	4
5	C_W^5	0	C_W^5	A^5	C_E^5	0	C_S^5	5
6	C_N^6	0	C_W^6	A^6	C_E^6	0	C_S^6	6
7	C_N^7	0	C_W^7	A^7	C_E^7	0	0	7
8	C_N^8	0	C_W^8	A^8	C_E^8	0	0	8
9	C_N^9	0	C_W^9	A^9	0	0	0	9
	1	2	3	4	5	6	7	

Figure 3:3. Band matrix.

All coefficients are now stored in a [m x n] by [2m + 1] matrix rather than a [m x n] by [m x n] matrix. In this particular case a 9 by 7 matrix does not seem to be much smaller than a 9 by 9 matrix. However, consider for example m = n = 20: a 400 by 41 matrix and a 400 by 400 matrix are considerably different. For grid systems in which the number of rows is not equal

to the number of columns (NX ≠ NY), it is important to organize the grid system such that NX is less than NY.

Line Successive Over Relaxation Method (LSOR)

The algorithm for point successive over relaxation method (SOR) in residual form is (see Figure 3:4):

$$H_{ij}^{*k+1} = (1 - \omega)H_{ij}^{*k} + \frac{\omega}{A}\left(C_N H_{ij-NX}^{*k+1} + C_W H_{ij-1}^{*k+1} + C_E H_{ij+1}^{*k} + C_S H_{ij+NX}^{*k} - r_{ij}\right) \quad (3:5)$$

where superscripts k and k + 1 represent k^{th} and $k + 1^{th}$ iterations, and ω is the over-relaxation parameter omega with a value between 1 and 2.

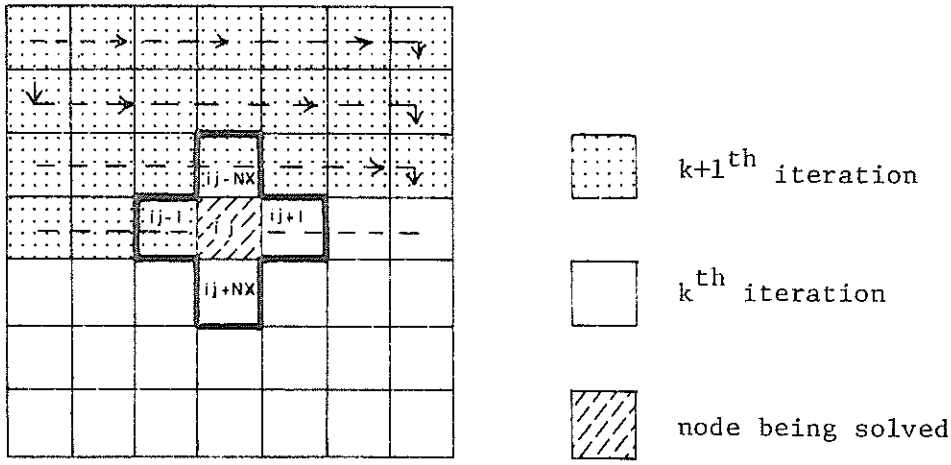


Figure 3:4. SOR method.

The only unknown in equation 3:5 is H_{ij}^{*k+1}. Consideration of equation 3:5 and Figure 3:4 shows that in order to solve for H_{ij}^{*k+1}, the old k-th iteration values ahead (H_{ij}^{*k}, H_{ij+1}^{*k}, H_{ij+NX}^{*k}) and the new (k + 1)-th iteration values behind (H_{ij-NX}^{*k+1}, H_{ij-1}^{*k+1}) are used; hence, only one matrix

value of $[H]^*$ needs to be stored. Equation 3:5 is set up and solved point by point until the whole system is completed. This procedure is called point successive over relaxation (SOR).

Instead of solving an equation point by point, the SOR algorithm can be modified to solve the problem row by row (or column by column): this process is called line successive over relaxation method (LSOR). (Figure 3:5.)

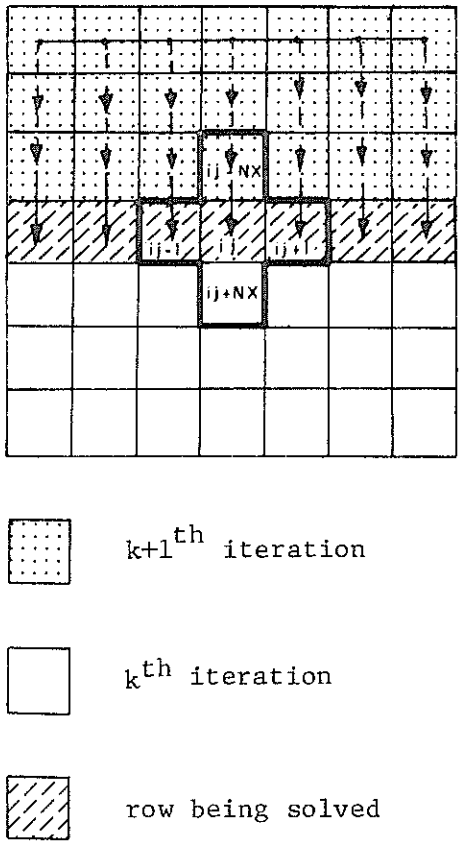


Figure 3:5. LSOR method.

The algorithm for LSOR is:

$$C_W H^{*k+1}_{ij-1} - A H^{*k+1}_{ij} + C_E H^{*k+1}_{ij+1}$$

$$= \omega[r_{ij} - C_N H^{*k+1}_{ij-NX} - C_S H^{*k}_{ij+NX}] \qquad (3:6)$$

$$+ (\omega - 1)\ [A H^{*k}_{ij} - C_W H^{*k}_{ij-1} - C_E H^{*k}_{ij+1}]$$

Equation 3:6 consists of at most three unknowns (H_{ij}^{*k+1}, H_{ij+1}^{*k+1}, H_{ij-1}^{*k+1}). A complete set of LSOR equations is then set up for a particular row and solved simultaneously by the use of Thomas Algorithm (explained below). This process is repeated row by row until the desired convergence is reached.

Thomas Algorithm

Tridiagonal matrices result from one-dimensional problems (such as a row or a column). The tridiagonal is the main and the two adjacent diagonals. Equation 3:6 rewritten for each block of a row (or a column) becomes:

$$
\begin{aligned}
b_1u_1 + c_1u_2 &= d_1 \\
a_2u_1 + b_2u_2 + c_2u_3 &= d_2 \\
a_3u_2 + b_3u_3 + c_3u_4 &= d_3 \\
\vdots \quad \vdots \quad \vdots &\quad \vdots \\
a_iu_{i-1} + b_iu_i + c_iu_{i+1} &= d_i \\
\vdots \quad \vdots \quad \vdots &\quad \vdots \\
a_{n-1}u_{n-2} + b_{n-1}u_{n-1} + c_{n-1}u_n &= d_{n-1} \\
a_nu_{n-1} + b_nu_n &= dn
\end{aligned}
\qquad (3:7)
$$

This system can be solved explicitly for the unknowns $u_1, u_2, \ldots, u_n$ thereby eliminating any matrix operations. The method is called the Thomas Algorithm (Rosenberg, 1969).

In general, the equations are:

$$a_iu_{i-1} + b_iu_i + c_iu_{i+1} = d_i \qquad (3:8)$$

for $1 \leq i \leq n$

with $a_i = c_n = 0$

$$\beta_i = b_i \frac{a_ic_{i-1}}{\beta_{i-1}} \qquad (3:9)$$

with $\beta_1 = b_1$

and

$$\gamma = \frac{d_i - a_i \gamma_{i-1}}{\beta_i} \tag{3:10}$$

with $\gamma_i = \frac{d_1}{b_1}$.

The values of the dependent variable are then computed from:

$$u_n = \gamma_n \tag{3:11}$$

$$u_i = \gamma_i - \frac{c_i u_{i+1}}{\beta_i} \tag{3:12}$$

This method has been shown to be stable for round-off error as long as $|b_i| \geq |a_i + c_u|$ for each equation.

IV. DATA

For each grid, the following parameters have to be defined.

(1) Initial Water Table Elevation

The water table elevation, preferably to the nearest 0.1 foot, needs to be read from the water table contour map at each node of the grid system.

(2) Final Water Table Elevation

A map of final water table elevation is used to determine the average final water level of each grid. The figures obtained should have the same accuracy as those of the initial water table. These figures are not read into the computer but are used for comparison with the predicted result.

(3) Simulating Boundary Conditions

The boundaries of the study area are designated in this program by the following codes:

0 represents an aquifer node.

1 represents an impermeable boundary.

2 represents underflow; i.e., groundwater boundary inflow or outflow (gradient boundary condition). The rate of outflow as well as the rate of inflow ought to be computed (or estimated).

3 and 4 are constant head boundary conditions. Hydraulically connected surface water sources such as a river or lake may be simulated by programming a time-varying or constant water surface elevation in the appropriate grids. The value of 3 is used to represent this kind of boundary condition along model boundaries; otherwise, inside the model the value 4 is assigned.

(4) Bedrock Elevation

The bedrock elevation data are read into the computer initially and are kept constant for all time steps.

(5) Hydraulic Conductivity

The values of hydraulic conductivity can be obtained from pumping tests or specific capacity data. Since the model can handle anisotropy in x- and y-directions, as well as nonhomogeneity, hydraulic conductivities in x- and y-directions and as a function of location are read into the computer for each node and are held constant for all time steps.

(6) Storage Coefficient (or specific yield)

These values can also be obtained from pumping tests. In this model storage coefficient is a function of space only. Therefore, the values are read into the computer initially and held constant for all time steps.

(7) Ground Water Withdrawals

Groundwater pumping for irrigation, municipal, and industrial needs may have considerable effect upon water levels. Heavy pumping almost always causes marked declines in the water levels and results in a decrease somewhere in the natural discharge of ground water. Annual pumped volumes can be estimated from certain State reports and from declaration forms of owners. The average monthly distribution pumping figures are used to estimate the volume pumped in each time step.

The coefficient of pumping, the fraction of pumped water not returning to the aquifer when applied during usage, depends on the different usages of ground water.

(8) Precipitation

Average daily, monthly, and yearly precipitation data are available from published records of selected stations. For example, data for Socorro, New Mexico, shows that most of the precipitation occurs during the summer months in the form of local thunderstorms: light snowfalls in winter are

infrequent. The average annual precipitation (1920-1953) in this area is 8.6 inches and the period of greatest precipitation is from July to September. Soil moisture measurements by the Research and Development Division of the New Mexico Institute of Mining and Technology show that 85 percent of the precipitation from a relatively heavy storm is lost by evaporation or run-off. Losses of 10 percent occur due to the effects of climate, vegetation, and topography. The remaining 5 percent of precipitation is usually considered to infiltrate and reach the ground water. This is referred to as the coefficient of precipitation in this model.

The monthly distribution of annual precipitation and the coefficient of precipitation are read into the computer initially and held constant throughout the study period.

(9) Phreatophytes

Annual consumptive use by phreatophytes can be estimated from technical reports published by the State Engineer office. Many factors influence the amount of water consumed by plants. The important natural influences are climate, water supply, soil, and topography. A monthly step distribution of consumptive use is assumed. These figures are read into the computer initially and held constant for all time steps.

(10) Boundary Flow

Darcy's Law is used to calculate boundary inflow and outflow by estimating potential gradients, cross-sectional areas, and hydraulic conductivities. These values are read into the computer in acre-feet/year and held constant throughout all time steps.

(11) Water Leaking Into or Out of Aquifer

These values are read into the computer in acre-feet/year and held constant throughout all time steps. Leakage can be deduced from excessive differences between computed and historical water levels which cannot otherwise be explained.

(12) Water Seeping to Aquifer from Irrigation and Drainage Canals

These values are read into the computer in acre-feet/year and held constant during the irrigation season or varied as a function of the amount of water applied.

(13) Finite Increments

Selection of space dimensions (ΔX, ΔY) is dependent upon the availability of geologic and hydraulic data and upon the desired accuracy and detail of analysis. These figures are read into the computer initially and held constant for all time steps.

(14) Ground Surface Elevation

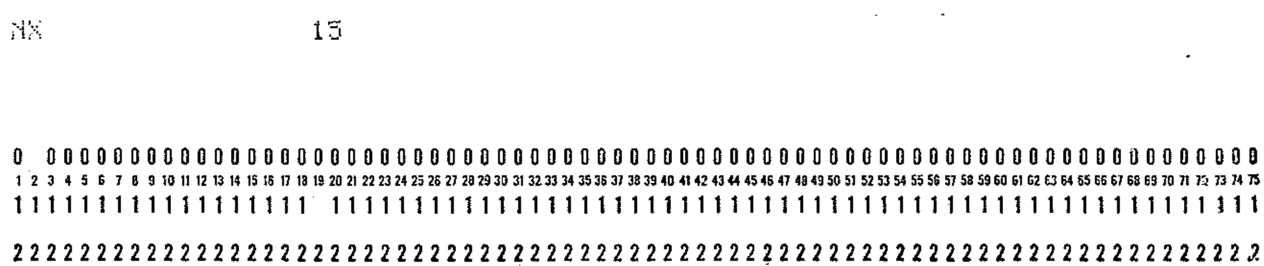
These values are only for reference, not for computation.

Data Arrangement

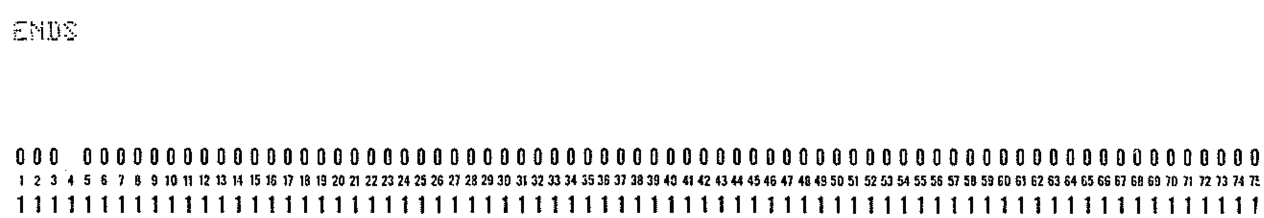
Since so many data are involved in this program, a good data arrangement is essential. There are three subroutines to meet this objective. A system of mnemonic name cards is used throughout the data deck. These mnemonic names consist of four characters. Their significance is explained in Section V on description of program subroutines. CMNT cards are comment cards. They can be put anywhere in the data deck and are disregarded by the READ statement.

(1) Subroutine SINGLE

This subroutine reads in the following single value data: NCOM, OMEG, IGAU, IWAT, HCONV, IDEB, NX, NY, DELT. The name of the single value will appear in the first 4 columns of the data card, then the value is punched in columns 11-20. (See next section for explanation of symbols.)



The computer checks the name first, then the numerical value; therefore, the sequence of single data cards is not important. The last one should be ENDS.



(2) Subroutine ARRAYS

This subroutine reads in array data. The array name will appear in the first 4 columns of the first card of that particular array.

If the array is a constant (for example, the initial water table is 50 feet in every grid), then the value of the constant should be punched in columns 21-30 on the array name card.

```
H                    50.
```

```
0000000000000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75
1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
```

If the array is variable, then a non-zero value should be punched in columns 11-20 on the array name card, and the numerical values of the array are punched per row with a format 7F10.0. For each new row, a new card is started. (Columns 71-80 can be used for identification.)

```
 4581.9    4579.6    4577.7    4575.1    4572.0    4568.6              H   ROW  2
4603.8    4600.8    4597.5    4594.1    4590.0    4586.3    4583.9    H   ROW  2
4583.9    4582.0    4580.0    4578.1    4575.9    4572.6              H   ROW  1
4605.6    4602.7    4600.0    4596.3    4593.3    4589.2    4586.3    H   ROW  1
H         1.0
```

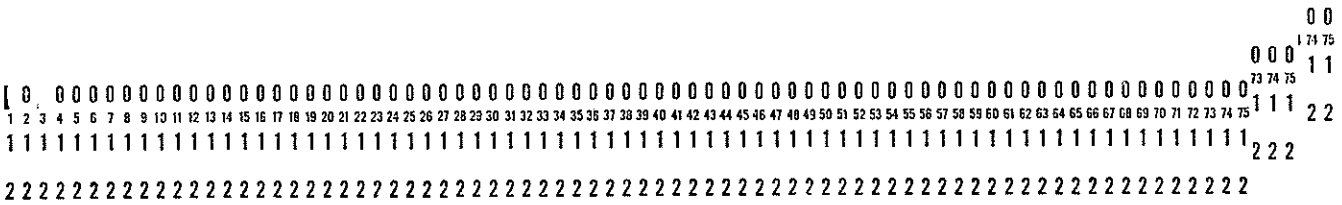
```
0000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
1111111111 1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
2222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
```

An "ENDA" card is needed at the end of the array data. (See next section for definition of array names.)

(3) Subroutine VALUES

This subroutine reads in all monthly distribution or variation values with format 7F10.0. The first card of every distribution set should be a name card to indicate different distributions. (See next section for definition of names.)

```
     0.171      0.145      0.074      0.034      0.053
   0.049      0.045      0.045      0.048      0.092      0.073      0.171
YPT
```



An "ENDV" card is needed at the end of all the values.

V. DESCRIPTION OF PROGRAM SUBROUTINES

Main Program

The main program initializes some values and calls for subroutine OLAY1 .

Subroutine OLAY1

This subroutine allocates the storage required for a particular size grid system.

Subroutine SINGLE

This subroutine reads in single value data.

(1) NCOM = number of computations

(2) OMEG = over-relaxation omega

(3) IGAU = assigned value for Gauss method (if zero, then this method is not used)

(4) IWAT = assigned value for LSOR method (if zero, then this method is not used)

(5) HCONV = convergence criterion

(6) IDEB = debugging switch (if 1, then intermediate calculations are printed out)

(7) NX = number of columns

(8) NY = number of rows

(9) DELT = one time-period of computation (days)

Subroutine ARRAYS

This subroutine reads in array data and writes out all the initial arrays.

(1) H = water table elevation (ft)

(2) IBC = boundary condition tag

0 - water table

1 - impermeable boundary

2 - inflow or outflow

3 - constant head (along boundaries)

4 - river or lake (within model boundaries)

(3) PPT = precipitation (in/year)

(4) RPUM = pumping rate (acre-feet/year)

(5) RFREA = phreatophyte water use (acre-feet/year)

(6) RLEAK = water leaking in or out (acre-feet/year)

(7) RSEEP = water seeping to aquifer from irrigation canals (acre-feet/year)

(8) QBC = inflow (+) or outflow (-) along model boundaries (acre-feet/year)

(9) Z = bedrock elevation (ft)

(10) G = ground surface elevation (ft)

(11) FKX = hydraulic conductivity in X-direction (ft/day)

(12) FKY = hydraulic conductivity in Y-direction (ft/day)

(13) DX = finite increment in X-direction (ft)

(14) DY = finite increment in Y-direction (ft)

(15) PHI = effective porosity

Subroutine VALUES

This subroutine reads in:

YPT = monthly precipitation distribution values

YPM = monthly pumping distribution values

YPR = monthly phreatophyte distribution values

YHH = monthly variations of river stage

CPT = the fraction of precipitation reaching ground water

CPM = coefficient of pumping (effective pumping, consumed).

Subroutine COEF

This subroutine computes east and south coefficients of flow. (West and north coefficients are obtained by symmetry.)

Subroutine DISTRI

This subroutine combines distribution values according to time step size.

Subroutine QFIX

This subroutine computes the net withdrawal rates, Q, for each grid during each time increment as a result of pumping, precipitation, boundary inflow and outflow, phreatophyte use, water leakage, and water seepage.

Subroutine TRANSG

This subroutine computes the transmissivities for each grid during each time increment and sets them up in reduced matrix form for Gauss Elimination (see Section III).

Subroutine RESIDG

This subroutine computes the residual term of the k-th iterate, which lies between time level t and $t+\Delta t$.

Subroutine GAUS1

This subroutine calls for subroutine BSOLVE and checks for convergence.

Subroutine BSOLVE

This subroutine solves the matrix equation by Gauss Elimination.

Subroutine TRANSW

This subroutine computes the transmissivities for each grid during each time increment for the use of Line Successive over Relaxation Method.

Subroutine RESIDW

This subroutine computes the residuals which will be used in Line Successive over Relaxation Method.

Subroutine LSOR

This subroutine calls for subroutine THOMAS and solves all the equations using Line Successive over Relaxation Method.

Subroutine THOMAS

This subroutine solves a tridiagonal matrix by the "THOMAS algorithm."

Subroutine BALCOM

This subroutine computes the material balance for each time step.

Subroutine BNDFLO

This subroutine computes the boundary inflow and/or outflow.

Subroutine MAP

This subroutine sets up different formats for printing corresponding arrays.

VI. MODEL CHARACTERISTICS

Size of Grid System

In this program, a blank common array A (see main program and subroutine OLAY1) is used to replace double-dimensional arrays. The biggest dimension allowed to be used for A in the IBM 360/44 computer is about 9000, and, therefore, the choice of optimum values for NX (No. of rows) and NY (No. of columns) is important.

Cards #10 to #31 in subroutine OLAY1 state the locations of all the arrays; consequently, the index ILAST will be the total number of storage locations required in the A array.

For Gaussian Elimination, some arrays are reallocated (cards #41 to #43). Card #43 states:

$$\text{ILAST} = \text{IA16} + \text{NX} * \text{NY} *(2 * \text{NY} + 1) - 1 \qquad (6:1)$$

which means that the last array needs NX * NY *(2 * NY + 1) words which is the largest array required. Apparently, much storage can be saved if NX is always greater than NY and should be considered when setting up the grid system.

Combining the information obtained from cards #10 to #31 and cards #41 to #43, the total number of words needed for the A array in case of Gauss Eliminations is:

$$\begin{aligned} &20 * \text{NX} * \text{NY} + \text{NX} * \text{NY} *(2 * \text{NY} + 1) \\ &= \text{NX} * \text{NY} *(21 + 2 * \text{NY}) \end{aligned} \qquad (6:2)$$

If this number is equated to the largest number allowed to be used for array A, then it will always give the maximum dimension of the system. That is,

$$\text{NX} * \text{NY} *(21 + 2 * \text{NY}) \leq 9000 \qquad (6:3)$$

or

$$NX \leq \frac{9000}{(21 + 2 * NY) * NY} \qquad (6:4)$$

For the LSOR method (some arrays are also reallocated [cards #51 to #67]), the array size can be calculated:

$$25 * NX * NY + 4 * NX + 2 * NY \leq 9000 \qquad (6:5)$$

or

$$NX \leq \frac{9000 - 2 * NY}{4 + 25 * NY} \qquad (6:6)$$

Table 6:1. Some maximum grid sizes for different NX and NY values.

	Gaussian Elimination		LSOR	
NY	NX	NX * NY	NX	NX * NY
5	54	270	69	345
10	21	210	36	360
15	11	165	33	345
20	7	140	17	340
25	5	125	14	350
30	3	90	11	330

Table 6:1 lists some maximum grid sizes for different values of NX and NY. It can be seen that LSOR is less restrictive on storage than Gauss Elimination and that it does not require that NX be greater than NY.

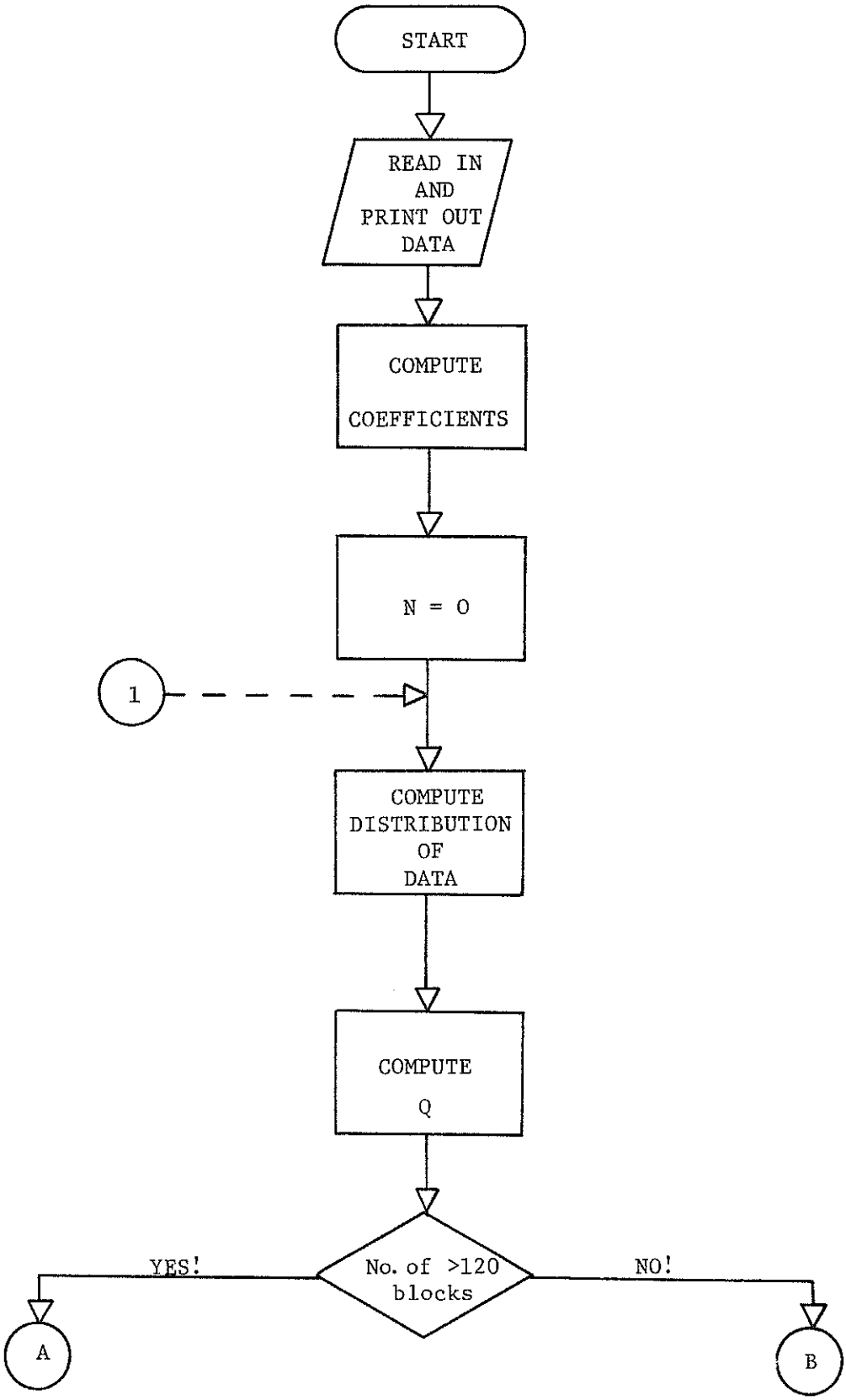
Time Step

For time steps smaller than one month, multiples of 5 days are required.

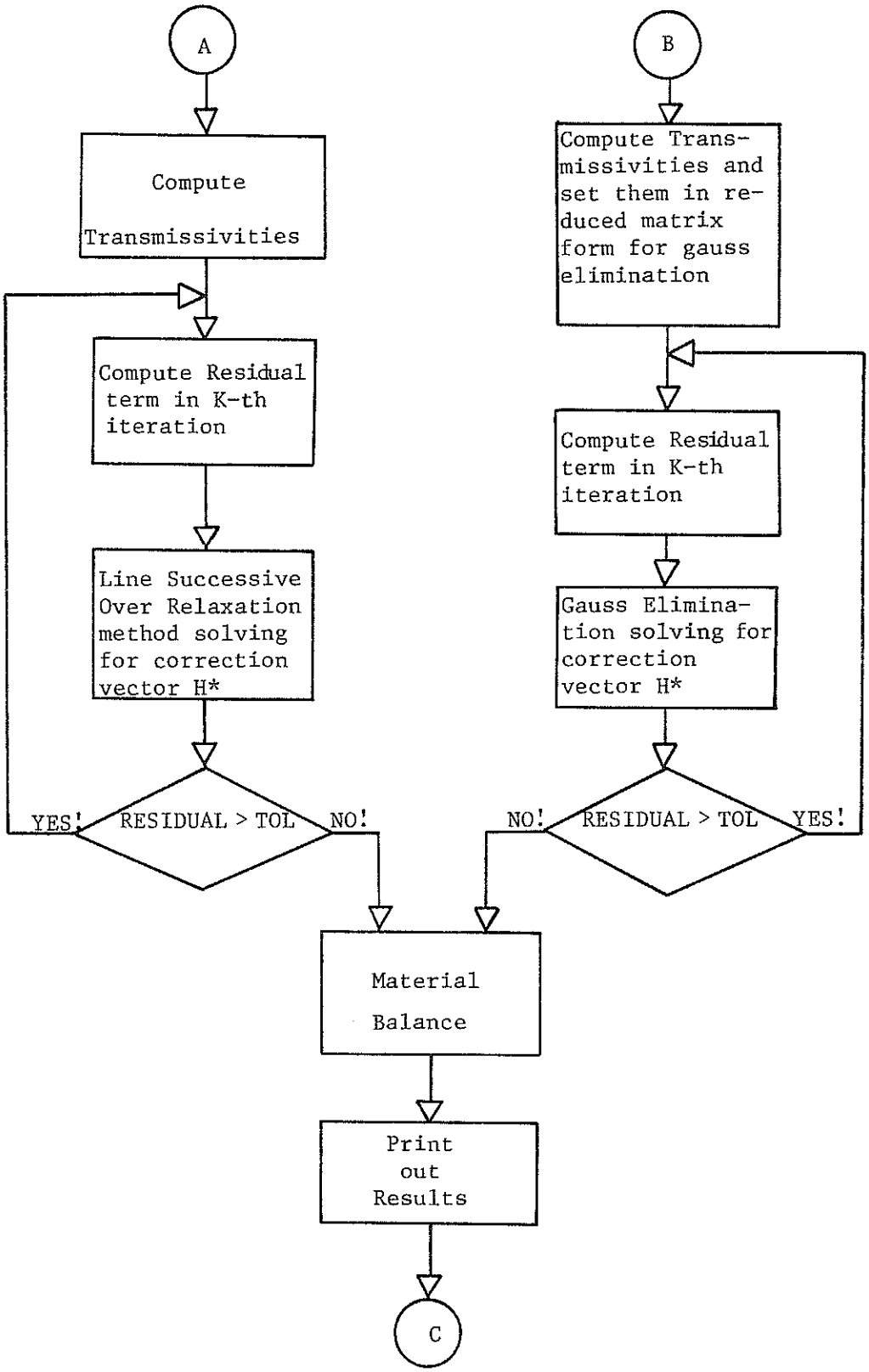
For time steps equal to or greater than 1 month, 1, 2, 3, 4, or 6 months can be utilized.

The largest time step is 1 year (or 365 days). The time step unit is in days.

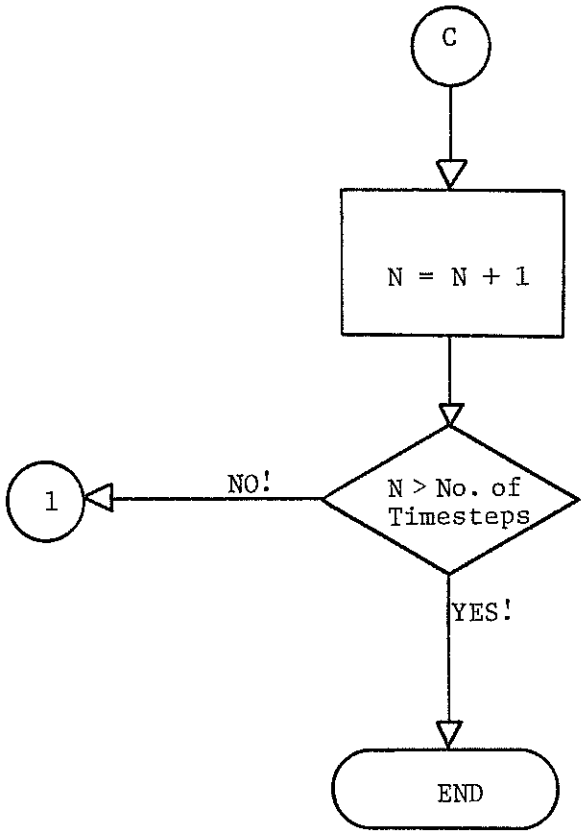
VII. FLOW CHART



(VII. cont.)


A
Compute
Transmissivities
Compute Residual term in K-th iteration
Line Successive Over Relaxation method solving for correction vector H*
YES!
RESIDUAL > TOL
NO!
B
Compute Transmissivities and set them in reduced matrix form for gauss elimination
Compute Residual term in K-th iteration
Gauss Elimination solving for correction vector H*
NO!
RESIDUAL > TOL
YES!
Material
Balance
Print
out
Results
C

(VII. cont.)



REFERENCES

Lansford, R. R., S. Ben-David, T. Gebhard, Jr., W. Brutsaert, and B. J. Creel, *An Analytical Interdisciplinary Evaluation of the Utilization of the Water Resources of the Rio Grande in New Mexico*, New Mexico Water Resources Research Institute Report 020, 1973, 152 pp.

_____, *An Analytical Interdisciplinary Evaluation of the Utilization of the Water Resources of the Rio Grande in New Mexico: Upper Rio Grande Region*, New Mexico Water Resources Research Institute Report 021, 1973.

_____, *An Analytical Interdisciplinary Evaluation of the Utilization of the Water Resources of the Rio Grande in New Mexico: Middle Rio Grande Region*, New Mexico Water Resources Research Institute Report 022, 1973.

_____, *An Analytical Interdisciplinary Evaluation of the Utilization of the Water Resources of the Rio Grande in New Mexico: Socorro Region*, New Mexico Water Resources Research Institute Report 023, 1973.

_____, *An Analytical Interdisciplinary Evaluation of the Utilization of the Water Resources of the Rio Grande in New Mexico: Lower Rio Grande Region*, New Mexico Water Resources Research Institute Report 024, 1973.

Rosenberg, O., *Methods for the Numerical Solution of Partial Differential Equations*, American Elsevier Publishing Company, Inc., New York, 1969.

Thurnau, D. H., "Algorithm 195, BANDSOLVE," *Communications of the ACM*, (1963), Vol. 6, No. 8, p. 441.

APPENDIX - COMPUTER LISTING

```
//SYS002 ACCESS SDSOJ1,1PO='TCCRFS'
//MAIN EXEC FORTRAN(MAP)
C----CONJUNCTIVE USE SURFACE WATER-GROUND WATER RESERVOIR SIMULATOR
      COMMON A(9700)
      COMMON/POINT/NTAPE,NCOM,OMEGA,IGAU,IWAT,HCONV,IDEBUG,JAGAIN,
     1            CUMTIM,CUMMB
      COMMON/STOR/TSTOR,BIN,POUT
      NTAPE  = 10
      JAGAIN = 0
      TSTOR  = 0.0
      CUMMB  = 0.0
      CUMTIM = 0.0
      CALL OLAY1
      END
/*
//OLAY1 EXEC FORTRAN(MAP)
      SUBROUTINE OLAY1
      COMMON A(9700)
      COMMON/POINT/NTAPE,NCOM,OMEGA,IGAU,IWAT,HCONV,IDEBUG,JAGAIN,
     1            CUMTIM,CUMMB
      COMMON/DISTR/CPM,CCP,CPT,DYPM,DYPT,DYPR,DYHH,DYPS,
     1            YPT(12),YPM(12),YPR(12),YHH(12),YPS(12)
      CALL SINGLE (NTAPE,ABORT,NX,NY,DELTAT,OMEGA,IGAU,IWAT,
     1            HCONV,NCOM,IDEBUG)
      NXNY = NX*NY
      NA = NX*NY
      NB = 2*NY+1
      ICOUNT = 0
C
CC
CC-----ALLOCATION OF CORE FOR THE FOLLOWING SUBROUTINES IS AS FOLLOWS
CC
CC
CC    ARRAY      SYMBOL     SIZE       DESCRIPTION
CC    -----      ------     ----       -----------
CC    IA1        CE         (NXNY)
CC    IA2        CS         (NXNY)
CC    IA3        EFAR       (NXNY)     EFFECTIVE AREA (DX*DY*PHI)
CC    IA4        H          (NXNY)     WATERTABLE ELEVATION
CC    IA5        IBC        (NXNY)     BOUNDARY CONDITION TAG:
CC                                     0 = WATERTABLE POINT
CC                                     1 = IMPERMEABLE BOUNDARY
CC                                     2 = INFLOW OR OUTFLOW
CC                                     3 = CONSTANT HEAD (ALONG BOUNDARIES)
CC                                     4 = RIVER OR LAKE (WITHIN MODEL BOUNDARIES)
CC                                          (=CONSTANT HEAD)
CC    IA6        PPT        (NXNY)     RAINFALL (IN/YR)
CC    IA7        RPUM       (NXNY)     PUMPING DISTRIBUTION (AC-FT/YR)
CC    IA8        RETEA      (NXNY)     PHREATOPHYTE WATER USE (AC-FT/YR)
CC    IA9        RLEAK      (NXNY)     WATER LEAKING IN OR OUT (AC-FT/YR)
CC    IA10       RSEEP      (NXNY)     WATER SEEPING TO AQUIFER FROM IRRIG. CANALS
CC    IA11       QBC        (NXNY)     INFLOW(+) OR OUTFLOW ALONG MODEL BOUNDARIES
CC                                          (AC-FT/YR)
CC    IA12       Z          (NXNY)     BEDROCK ELEVATION (FT)
CC    IA13       G          (NXNY)     GROUND SURFACE ELEVATION (FT)
CC    IA14       HP         (NXNY)     WATER TABLE ELEV. OF PREVIOUS TIME STEP (FT)
CC    IA15       Q          (NXNY)     NET RATE OF WITHDRAWAL (AC-FT/DAY)
CC    IA16       FKX        (NXNY)     HYDRAULIC CONDUCTIVITY IN X-DIR. (FT/DAY)
CC    IA17       FKY        (NXNY)     HYDRAULIC CONDUCTIVITY IN Y-DIR. (FT/DAY)
CC    IA18       DX         (NXNY)     FINITE INCREMENTS IN X-DIRECTION
CC    IA19       DY         (NXNY)     FINITE INCREMENTS IN Y-DIRECTION
CC    IA20       PHI        (NXNY)     EFFECTIVE POROSITY
CC
CC
CC
```

```
C           ARRAY INDICES
C           -------------
      IA1 = 1
      IA2 = IA1 + NXNY
      IA3 = IA2 + NXNY
      IA4 = IA3 + NXNY
      IA5 = IA4 + NXNY
      IA6 = IA5 + NXNY
      IA7 = IA6 + NXNY
      IA8 = IA7 + NXNY
      IA9 = IA8 + NXNY
      IA10= IA9 + NXNY
      IA11= IA10+ NXNY
      IA12= IA11+ NXNY
      IA13= IA12+ NXNY
      IA14= IA13+ NXNY
      IA15= IA14+ NXNY
      IA16= IA15+ NXNY
      IA17= IA16+ NXNY
      IA18= IA17+ NXNY
      IA19= IA18+ NXNY
      IA20= IA19+ NXNY
      ILAST = IA20 + NXNY - 1
C
C
C
C                              H       IBC     PPT    RPUM  RFREA  RLEAK
      CALL ARRAYS (NX,NY,A(IA4),A(IA5),A(IA6),A(IA7),A(IA8),A(IA9),
C                   RSEEP    QBC       Z         Q       HP      FKX
     1             A(IA10),A(IA11),A(IA12),A(IA13),A(IA14),A(IA16),
C                    FKY      DX       DY       PHI
     2             A(IA17),A(IA18),A(IA19),A(IA20),CUMTIM)
C
C
      CALL VALUES (YPT,YPM,YPR,YHH,YPS,CPT,CPM)
C
C                             PHI       DX       DY     EFAP     FKX     FKY
      CALL COEF (NX,NY,A(IA20),A(IA18),A(IA19),A(IA3),A(IA16),A(IA17),
C                   CE       CS      IBC
     1           A(IA1),A(IA2),A(IA5),CUMTIM,IDEBUG)
C
C
      IF (NXNY.GT. 20) IWAT=1
C
  500 JAGAIN = 0
      CUMTIM = CUMTIM + DELTAT
C                              PPT     RPUM  RFREA     H       IBC
      CALL DISTRI (NX,NY,A(IA6),A(IA7),A(IA8),A(IA4),A(IA5),
     1              YPM,YPT,YPR,YHH,YPS,DELTAT,ICOUNT,DYPM,DYPT,
     2              DYPR,DYHH,DYPS)
C
C                             H       PPT    PPUM  RFREA  RLEAK  RSEEP
      CALL QFIX (NX,NY,A(IA4),A(IA6),A(IA7),A(IA8),A(IA9),A(IA10),
C                   QBC       Z        Q
     1           A(IA11),A(IA12),A(IA15),
     2           DELTAT,IWAT,SQVOL,DYPM,DYPT,DYPR,DYPS,CPT,CPM,CUMTIM)
C
C
      IF (IWAT.EQ.1) GO TO 200
      CALL LOAD('PHASEGAU')
C
C-----REALLOCATION OF CORE IN CASE OF GAUSS ELIMINATION METHOD BSOLVE
C
C     ARRAY      SYMBOL     SIZE       DESCRIPTION
C     -----      ------     -----      -----------
C     IA15       CR         (NXNY)     R.H.S. VECTOR MATRIX (OVER READS Q-ARRAY)
C     IA16       CMATRX     (NXNY,2*NY+1) REDUCED COEFFICIENT MATRIX (BAND MATRIX)
C
C
C
C           ARRAY INDICES
C           -------------
      IA15 = IA14 + NXNY
      IA16 = IA15 + NXNY
      ILAST= IA16 + NXNY*(2*NY+1) - 1
```

```
C
C
C
C                    Q,CR   CMATRX   EFAR     HP       Z        CE       CS
      CALL TRANSG (A(IA15),A(IA16),A(IA3),A(IA14),A(IA12),A(IA1),A(IA2),
C                  IBC
     1            A(IA5),NX,NY,DELTAT,NA,NB,IDEBUG)
C
C
  400 JAGAIN = JAGAIN + 1
C
C                          CMATRX     CR       H      IBC     EFAR     HP
      CALL RESIDG (NX,NY,A(IA16),A(IA15),A(IA4),A(IA5),A(IA3),A(IA14),
     1             NA,NB,JAGAIN,IDEBUG,DELTAT)
C
C                          CMATRX     CR       H      IBC     EFAR     HP
      CALL GAUS1 (NX,NY,A(IA16),A(IA15),A(IA4),A(IA5),A(IA3),A(IA14),
     1           NA,NB,HCONV,ITER,IDEBUG,CUMTIM,DELTAT,JAGAIN,ICOUNT)
C
C
      IF (ITER.EQ.1) GO TO 400
      GO TO 300
C
C-----REALLOCATION OF CORE IN CASE OF WATTS METHOD
C
C     ARRAY      SYMBOL     SIZE       DESCRIPTION
C     -----      ------     -----      -----------
C     IA15       RHS        (NXNY)     R.H.S. OF FLOW EQUATION (OVER READS Q)
C     IA16       AN         (NXNY)     NORTH COEFFICIENT OF FLOW EQUATION
C     IA17       AS         (NXNY)     SOUTH COEFFICIENT OF FLOW EQUATION
C     IA18       AE         (NXNY)     EAST  COEFFICIENT OF FLOW EQUATION
C     IA19       AW         (NXNY)     WEST  COEFFICIENT OF FLOW EQUATION
C     IA20       AO         (NXNY)     CENTRAL COEFFICIENT OF FLOW EQUATION
C     IA21       AOT        (NXNY)     AO TEMPORARY (USED IN CLSOR)
C     IA22       P          (NXNY)     PRESSURE DIFFERENCE IN CLSOR (HSTAR)
C     IA23       PORVOL     (NXNY)     CONVERGENCE CRITERION OF CLSOR
C     IA24       DELP       (NY)       WORKING ARRAY IN CLSOR
C     IA25       AWATTS     (NX)       WORKING ARRAY IN CLSOR
C     IA26       BWATTS     (NX)       WORKING ARRAY IN CLSOR
C     IA27       CWATTS     (NX)       WORKING ARRAY IN CLSOR
C     IA28       DWATTS     (NX)       WORKING ARRAY IN CLSOR
C     IA29       G          (NY)       WORKING ARRAY IN CLSOR
C     IA30       R          (NXNY)     RESIDUAL TERM
C
C
C
C
C          ARRAY INDICES
C          -------------
  200 CALL LOAD('PHASEWAT')
      IA15 = IA14 + NXNY
      IA16 = IA15 + NXNY
      IA17 = IA16 + NXNY
      IA18 = IA17 + NXNY
      IA19 = IA18 + NXNY
      IA20 = IA19 + NXNY
      IA21 = IA20 + NXNY
      IA22 = IA21 + NXNY
      IA23 = IA22 + NXNY
      IA24 = IA23 + NXNY
      IA25 = IA24 + NY
      IA26 = IA25 + NX
      IA27 = IA26 + NX
      IA28 = IA27 + NX
      IA29 = IA28 + NX
      IA30 = IA29 + NY
      ILAST= IA30 + NXNY - 1
C                     RHS     EFAP     HP        Z        CE       CS       IBC
      CALL TRANSW (A(IA15),A(IA3),A(IA14),A(IA12),A(IA1),A(IA2),A(IA5),
C                                  AN       AS       AE       AW       AO
     1             NX,NY,DELTAT,A(IA16),A(IA17),A(IA18),A(IA19),A(IA20))
C
  401 JAGAIN = JAGAIN + 1
C
C                           RHS      H     IBC     EFAR
```

```
      CALL RESIDW (NX,NY,A(IA15),A(IA4),A(IA5),A(IA3),JAGAIN,IDEBUG,
C                                 AN      AS      AE      AW      AD
     1             DELTAT,A(IA16),A(IA17),A(IA18),A(IA19),A(IA20),
C                        R       HP
     2             A(IA30),A(IA14),CUMTIM)
C
CC
C                     AN      AS      AE      AW      AD      HST
      CALL LSOR (A(IA16),A(IA17),A(IA18),A(IA19),A(IA20),A(IA22),
C                  R       A       B       C       D
     1           A(IA30),A(IA25),A(IA26),A(IA27),A(IA28),NX,NY,
C                                          H      IBC     EFAR    HP
     2           OMEGA,IDEBUG,CUMTIM,A(IA4),A(IA5),A(IA3),A(IA14),
     3           HCONV,ITER,DELTAT,JAGAIN,ICOUNT)
C
      IF (ITER.EQ.1) GO TO 401
  300 CONTINUE
CCCCCCCCCC
C                           H       HP     EFAR    IBC       Z        G
      CALL BALCOM (NX,NY,A(IA4),A(IA14),A(IA3),A(IA5),A(IA12),A(IA13),
C                      CE      CS
     1             A(IA1),A(IA2),DELTAT,CUMTIM,CUMMR,SQVOL,ICOUNT)
      ICOUNT = ICOUNT + 1
      IF (ICOUNT.LT.NCOM) GO TO 500
      RETURN
      END
```

```
/*
//SINGLE EXEC FORTRAN(MAP)
      SUBROUTINE SINGLE (NTAPE,APORT,NX,NY,DELTAT,OMEGA,IGAU,IWAT,
     1                   HCONV,NCOM,IDEBUG)
C
C--VARIABLE DESIGNATION AS USED
CCC           IN PROGRAM          ON CARDS          IN DATA STAT.
CCC           ----------          --------          -------------
CCC                               CMNT              KCMN
CCC           NCOM                NCOM              KNCO
CCC           OMEGA               OMEG              KOME
CCC           IGAU                IGAU              KIGA
CCC           IWAT                IWAT              KIWA
CCC           HCONV               HCON              KHCO
CCC           IDEBUG              DEBU              KDEB
CCC           DELTAT              DELT              KDEL
CCC           NX                  NX                KNX
CCC           NY                  NY                KNY
CCC                               ENDS              KEND
C
      DATA KCMN/'CMNT'/,KNCO/'NCOM'/,KOME/'OMEG'/,KIGA/'IGAU'/,
     1     KIWA/'IWAT'/,KHCO/'HCON'/,KDEB/'DEBU'/,KDEL/'DELT'/,
     2     KNX /'NX  '/,KNY /'NY  '/,KEND/'ENDS'/
C
   10 READ (5,1) NAME,X1
      IF (NAME.EQ.KCMN) GO TO 10
      IF (NAME.EQ.KEND) GO TO 999
      IF (NAME.EQ.KNCO) GO TO 100
      IF (NAME.EQ.KOME) GO TO 200
      IF (NAME.EQ.KIGA) GO TO 300
      IF (NAME.EQ.KIWA) GO TO 400
      IF (NAME.EQ.KHCO) GO TO 500
      IF (NAME.EQ.KDEB) GO TO 600
      IF (NAME.EQ.KDEL) GO TO 700
      IF (NAME.EQ.KNX ) GO TO 800
      IF (NAME.EQ.KNY ) GO TO 900
```

```
C
  100  NCOM   = X1
       GO TO 10
C
  200  OMEGA  = X1
       GO TO 10
C
  300  IGAU   = X1
       GO TO 10
C
  400  IWAT   = X1
       GO TO 10
C
  500  HCONV  = X1
       GO TO 10
C
  600  IDEBUG = X1
       GO TO 10
C
  700  DELTAT = X1
       GO TO 10
C
  800  NX     = X1
       GO TO 10
C
  900  NY     = X1
       GO TO 10
C
  999  CONTINUE
    1  FORMAT (A4,6X,F10.0)
       RETURN
       END
```

```
/*
//ARRAYS EXEC FORTRAN(MAP)
      SUBROUTINE ARRAYS (NX,NY,H,IBC,PPT,RPUM,RFREA,RLEAK,RSEEP,QBC,Z,
     1                   G,HP,FKX,FKY,DX,DY,PHI,CUMTIM)
      DIMENSION H(1),IBC(1),PPT(1),RPUM(1),RFREA(1),RLEAK(1),RSEEP(1),
     1          QBC(1),Z(1),G(1),HP(1),FKX(1),FKY(1),DX(1),DY(1),PHI(1)
C--VARIABLE DESIGNATION AS USED
C              IN GROGRAM          ON CARDS        IN DATA STAT.
C              ----------          --------        -------------
C                                  CMNT            KCMN
C              H                   H               KH
C              IBC                 IBC             KIBC
C              PPT                 PPT             KPPT
C              RPUM                PUM             KPUM
C              RFREA               FREA            KFRE
C              RLEAK               LEAK            KLEA
C              RSEEP               SEEP            KSEE
C              QBC                 QBC             KQBC
C              Z                   Z               KZ
C              G                   G               KG
C              FKX                 FKX             KFKX
C              FKY                 FKY             KFKY
C              DX                  DX              KDX
C              DY                  DY              KDY
C              PHI                 PHI             KPHI
C                                  ENDA            KEND
C
      DATA KCMN/'CMNT'/,KH  /'H   '/,KIBC/'IBC '/,KPPT/'PPT '/,
     1     KPUM/'PUM '/,KFRE/'FREA'/,KLEA/'LEAK'/,KSEE/'SEEP'/,
     2     KQBC/'QBC '/,KZ  /'Z   '/,KG  /'G   '/,KFKX/'FKX '/,
     3     KFKY/'FKY '/,KDX /'DX  '/,KDY /'DY  '/,KPHI/'PHI '/,
     4     KEND/'ENDA'/
C
      NXNY = NX*NY
C
C1    READ (5,10) NAME,X1,X2
C
```

```
      IF (NAME.EQ.KCMU) GO TO 1
      IF (NAME.EQ.KEND) GO TO 9999
      IF (NAME.EQ.KH   ) GO TO 100
      IF (NAME.EQ.KIBC) GO TO 200
      IF (NAME.EQ.KPHI) GO TO 300
      IF (NAME.EQ.KPUM) GO TO 400
      IF (NAME.EQ.KRRE) GO TO 500
      IF (NAME.EQ.KLEA) GO TO 600
      IF (NAME.EQ.KSEE) GO TO 700
      IF (NAME.EQ.KQBC) GO TO 800
      IF (NAME.EQ.KZ   ) GO TO 900
      IF (NAME.EQ.KG   ) GO TO 1000
      IF (NAME.EQ.KEKX) GO TO 1100
      IF (NAME.EQ.KEKY) GO TO 1200
      IF (NAME.EQ.KDX  ) GO TO 1300
      IF (NAME.EQ.KDY  ) GO TO 1400
      IF (NAME.EQ.KPPT) GO TO 1500
C
 100  IF (X1.NE.0.) GO TO 101
      DO 11 I=1,NXNY
      H(I)  = X2
 11   HP(I) = H(I)
      GO TO 1
 101  DO 102 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 102  READ (5,2) (H(K),K=K1,K2)
      DO 3 I=1,NXNY
 3    HP(I) = H(I)
      GO TO 1
C
 200  IF (X1.NE.0.) GO TO 201
      DO 22 I=1,NXNY
 22   IBC(I) = X2
      GO TO 1
 201  DO 202 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 202  READ(5,4) (IBC(K),K=K1,K2)
      GO TO 1
C
 300  IF (X1.NE.0.) GO TO 301
      DO 33  I=1,NXNY
 33   PHI(I) = X2
      GO TO 1
 301  DO 302 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 302  READ (5,2) (PHI(K),K=K1,K2)
      GO TO 1
 400  IF (X1.NE.0.) GO TO 401
      DO 44 I=1,NXNY
 44   RPUM(I) = X2
      GO TO 1
 401  DO 402 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 402  READ (5,2) (RPUM(K),K=K1,K2)
      GO TO 1
C
 500  IF (X1.NE.0.) GO TO 501
      DO 55 I=1,NXNY
 55   RREA(I) = X2
      GO TO 1
 501  DO 502 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 502  READ (5,2) (RREA(K),K=K1,K2)
      GO TO 1
C
 600  IF (X1.NE.0.) GO TO 601
      DO 66 I=1,NXNY
 66   RLEAK(I) = X2
      GO TO 1
 601  DO 602 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 602  READ (5,2) (RLEAK(K),K=K1,K2)
      GO TO 1
C
```

```
  700   IF (X1.NE.0.) GO TO 701
        DO 77 I=1,NXNY
  77    RSEEP(I) = X2
        GO TO 1
  701   DO 702 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
  702   READ (5,2) (RSEEP(K),K=K1,K2)
        GO TO 1
C
  800   IF (X1.NE.0.) GO TO 801
        DO 88 I=1,NXNY
  88    QBC(I) = X2
        GO TO 1
  801   DO 802 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
  802   READ (5,2) (QBC(K),K=K1,K2)
        GO TO 1
C
  900   IF (X1.NE.0.) GO TO 901
        DO 99 I=1,NXNY
  99    Z(I) = X2
        GO TO 1
  901   DO 902 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
  902   READ (5,2) (Z(K),K=K1,K2)
        GO TO 1
C
 1000   IF (X1.NE.0.) GO TO 1001
        DO 111 I=1,NXNY
  111   G(I) = X2
        GO TO 1
 1001   DO 1002 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 1002   READ (5,2) (G(K),K=K1,K2)
        GO TO 1
C
 1100   IF (X1.NE.0.) GO TO 1101
        DO 112 I=1,NXNY
  112   FKX(I) = X2
        GO TO 1
 1101   DO 1102 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 1102   READ (5,2) (FKX(K),K=K1,K2)
        GO TO 1
C
 1200   IF (X1.NE.0.) GO TO 1201
        DO 123 I=1,NXNY
  123   FKY(I) = X2
        GO TO 1
 1201   DO 1202 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 1202   READ (5,2) (FKY(K),K=K1,K2)
        GO TO 1
C
 1300   IF (X1.NE.0.) GO TO 1301
        DO 134 I=1,NXNY
  134   DX(I) = X2
        GO TO 1
 1301   DO 1302 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 1302   READ (5,2) (DX(K),K=K1,K2)
        GO TO 1
C
 1400   IF (X1.NE.0.) GO TO 1401
        DO 145 I=1,NXNY
  145   DY(I) = X2
        GO TO 1
 1401   DO 1402 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 1402   READ (5,2) (DY(K),K=K1,K2)
        GO TO 1
C
```

```
 1500 IF (X1.NE.0.) GO TO 1501
      DO 156 I=1,NX
         DO 156 J=1,NY
              IJ = NX*(J-1)+I
              PPT(IJ) = X2
              PPT(IJ) = PPT(IJ)*DX(IJ)*DY(IJ)/(43560.*12.)
 156  CONTINUE
      GO TO 1
 1501 DO 1502 J=1,NY
           K1 = NX*(J-1)+1
           K2 = K1+NX-1
 1502 READ (5,2) (PPT(K),K=K1,K2)
      DO 256 IJ=1,NXNY
 256  PPT(IJ) = PPT(IJ)*DX(IJ)*DY(IJ)/(43560.*12.)
      GO TO 1
C
 9999 CONTINUE
C-------------------PRINT OUT DATA
      CALL MAP (H,4HH    ,2.,NX,NY,CUMTIM)
      CALL MAP (HP,4HHP   ,2.,NX,NY,CUMTIM)
      CALL MAP (PPT,4HPPT ,2.,NX,NY,CUMTIM)
      CALL MAP (RPUM,4HPUM ,2.,NX,NY,CUMTIM)
      CALL MAP (RPREA,4HPREA,2.,NX,NY,CUMTIM)
      CALL MAP (RLEAK,4HLEAK,2.,NX,NY,CUMTIM)
      CALL MAP (RSEEP,4HSEEP,2.,NX,NY,CUMTIM)
      CALL MAP (QRC,4HQRC ,2.,NX,NY,CUMTIM)
      CALL MAP (FKX,4HFKX ,2.,NX,NY,CUMTIM)
      CALL MAP (FKY,4HFKY ,2.,NX,NY,CUMTIM)
      CALL MAP (DX,4HDX   ,2.,NX,NY,CUMTIM)
      CALL MAP (DY,4HDY   ,2.,NX,NY,CUMTIM)
      CALL MAP (PHI,4HPHI ,2.,NX,NY,CUMTIM)
      CALL MAP (G,4HG    ,2.,NX,NY,CUMTIM)
      CALL MAP (Z,4HZ    ,2.,NX,NY,CUMTIM)
      PRINT 3000, CUMTIM
 3000 FORMAT (1H0/40X,19HBOUNDARY CONDITIONS,2X,3HMAP,20X,7HTIME = ,
     1       F10.4,2X,4HDAYS/)
      DO 5000 J=1,NY
           K1 = NX*(J-1) + 1
           K2 = K1 + NX - 1
 5000 WRITE (6,2000) (IBC(K),K=K1,K2)
 2000 FORMAT (30I4)
 2    FORMAT (7F10.0)
 4    FORMAT (7I10)
 10   FORMAT (A4,6X,2F10.0)
      RETURN
      END
```

```
/*
//VALUES EXEC FORTRAN(MAP)
      SUBROUTINE VALUES (YPT,YPM,YPR,YHH,YPS,CPT,CPM)
      DIMENSION YPT(12),YPM(12),YPR(12),YHH(12),YPS(12)
C---THIS SUBROUTINE READS IN
C         YPT = PRECIPITATION DISTRIBUTION VALUES
C         YPM = PUMPING DISTRIBUTION VALUES
C         YPR = PHREATOPHYTE DISTRIBUTION VALUES
C         YHH = VARIATIONS OF RIVER STAGES
C         YPS = SEEPAGE DISTRIBUTION VALUES
C         CPT = THE FRACTION OF PRECIPITATION REACHING GROUND WATER
C         CPM = COEFFICIENT OF PUMPING (EFFECTIVE PUMPING,CONSUMED)
C
      DATA KCMN/'CMNT'/,KYPT/'YPT '/,KYPM/'YPM '/,KCPT/'CPT '/,
     1     KCPM/'CPM '/,KEND/'ENDV'/,KYPR/'YPR '/,KYHH/'YHH '/,
     2     KYPS/'YPS '/
C
 10   READ (5,1) NAME,X1
C
      IF (NAME.EQ.KCMN) GO TO 10
      IF (NAME.EQ.KEND) GO TO 999
      IF (NAME.EQ.KYPT) GO TO 100
      IF (NAME.EQ.KYPM) GO TO 200
      IF (NAME.EQ.KYPR) GO TO 300
      IF (NAME.EQ.KYHH) GO TO 400
      IF (NAME.EQ.KYPS) GO TO 500
      IF (NAME.EQ.KCPT) GO TO 600
      IF (NAME.EQ.KCPM) GO TO 700
C
```

```
  100   READ (5,2) (YPT(I),I=1,12)
        GO TO 10
C
  200   READ (5,2) (YPM(I),I=1,12)
        GO TO 10
C
  300   READ (5,2) (YPR(I),I=1,12)
        GO TO 10
C
  400   READ (5,2) (YHH(I),I=1,12)
        GO TO 10
C
  500   READ (5,2) (YPS(I),I=1,12)
        GO TO 10
C
  600   CPT = X1
        GO TO 10
C
  700   CPM = X1
        GO TO 10
C
  999   CONTINUE
    1   FORMAT (A4,6X,F10.0)
    2   FORMAT (7F10.0)
        RETURN
        END


/*
//COEF EXEC FORTRAN(MAP)
      SUBROUTINE COEF (NX,NY,PHI,DX,DY,EEAR,EKX,EKY,CE,CS,IBC,CUMTIM,
     1                 IDEBUG)
      DIMENSION PHI(1),DX(1),DY(1),EEAP(1),EKX(1),EKY(1),CE(1),CS(1),
     1          IBC(1)
      REAL*8 DEEAR,DCE,DCS,DDX,DDY,DEKX,DEKY,DDX1,DDXNX,DEKX1,DEKXNX,
     1       DDY1,DDYNX,DEKY1,DEKYNX,DPHI
C----THIS SUBROUTINE COMPUTES THE EAST AND SOUTH COEFFICIENTS OF THE FLOW
C----EQUATION
C
      DO 100 I = 1,NX
          DO 100 J = 1,NY
                IJ = NX*(J-1)+I
                DPHI = PHI(IJ)
                DDY  = DY(IJ)
                DDX  = DX(IJ)
                IF (I.EQ.1.OR.I.EQ.NX) DDX = DDX/2.
                CE(IJ)    = 0.0
                CS(IJ)    = 0.0
                DEEAR     = DDX*DDY*DPHI
                EEAP(IJ)  = DEEAP
                DDX       = DX(IJ)
                DDX1      = DX(IJ+1)
                DDXNX     = DX(IJ+NX)
                DEKX      = EKX(IJ)
                DEKY      = EKY(IJ)
                DEKX1     = EKX(IJ+1)
                DEKXNX    = EKX(IJ+NX)
                DEKY1     = EKY(IJ+1)
                DEKYNX    = EKY(IJ+NX)
                DDY1      = DY(IJ+1)
                DDYNX     = DY(IJ+NX)
                IF (EKX(IJ).EQ.0.0.OR.EKY(IJ).EQ.0.0) EEAR(IJ)=0.0
                IF (I.EQ.NX) GO TO 50
                IF (EKX(IJ).EQ.0.0.OR.EKX(IJ+1).EQ.0.0) GO TO 50
                DCE = 2.*DEKX*DEKX1*DDY*DDY1/
     1                (DDX*DDY1*DEKX1+DDX1*DEKX*DDY)
                CE(IJ)    = DCE
   50           IF (J.EQ.NY) GO TO 100
                IF (EKY(IJ).EQ.0.0.OR.EKY(IJ+NX).EQ.0.0) GO TO 100
                DCS = 2.*DEKY*DEKYNX*DDX*DDXNX/
     1                (DDY*DEKYNX*DDXNX+DDYNX*DEKY*DDX)
                CS(IJ)    = DCS
  100 CONTINUE
C
      IF (IDEBUG.EQ.1)
     1CALL MAP (EEAR,4HEEAR,2.,NX,NY,CUMTIM)
      IF (IDEBUG.EQ.1)
     1CALL MAP (CE,4HCE  ,2.,NX,NY,CUMTIM)
      IF (IDEBUG.EQ.1)
     1CALL MAP (CS,4HCS  ,2.,NX,NY,CUMTIM)
      RETURN
      END
```

```
/*
//DISTRI EXEC FORTRAN(MAP)
      SUBROUTINE DISTRI (NX,NY,PPT,RPUM,RFREA,H,IBC,YPM,YPT,
     1                   YPR,YHH,YPS,DELTAT,ICOUNT,DYPM,DYPT,DYPR,DYHH,
     2                   DYPS)
      DIMENSION PPT(1),RPUM(1),RFREA(1),H(1),IBC(1),YHH(12),
     1          YPM(12),YPT(12),YPR(12),YPS(12)
C-----THIS SUBROUTINE COMPUTES THE DISTRIBUTION OF DATA.
C
      JCOUNT = ICOUNT + 1
      KD = JCOUNT
      I2 = 365.2/DELTAT
 1    IF (KD.LE.I2) GO TO 3
      KD = KD - I2
      GO TO 1
 3    CONTINUE
C
      IF (I2.LT.12) GO TO 6
      I3 = I2/12
      I = (KD-1)/I3 + 1
         DYPM = YPM(I)/I3
         DYPT = YPT(I)/I3
         DYPR = YPR(I)/I3
         DYHH = YHH(I)/I3
         DYPS = YPS(I)/I3
         GO TO 7
C
 6    I4 = 12/I2
         DYPM = 0.0
         DYPT = 0.0
         DYPR = 0.0
         DYHH = 0.0
         DYPS = 0.0
         N1 = (KD-1)*I4+1
         N2 = N1+I4-1
           DO 5 I=N1,N2
              DYPM = DYPM+YPM(I)
              DYPT = DYPT+YPT(I)
              DYPR = DYPR+YPR(I)
              DYHH = DYHH+YHH(I)
              DYPS = DYPS+YPS(I)
 5            CONTINUE
 7    CONTINUE
C
      DO 4 I=1,NX
           DO 4 J=1,NY
                IJ = NX*(J-1)+I
                IF (IBC(IJ).NE.4) GO TO 4
                H(IJ) = H(IJ) + DYHH
 4    CONTINUE
      RETURN
      END
```

```
/*
//QFIX EXEC FORTRAN(MAP)
      SUBROUTINE QFIX (NX,NY,H,PPT,RPUM,RFREA,RLEAK,RSEEP,QBC,Z,Q,
     1                 DELTAT,IWAT,SQVOL,DYPM,DYPT,DYPR,DYPS,CPT,CPM,
     2                 CUMTIM)
      DIMENSION H(1),PPT(1),RPUM(1),RFREA(1),RLEAK(1),RSEEP(1),QBC(1),
     1          Z(1),Q(1)
      COMMON/STOR/TSTOR,BIN,BOUT
C--THIS SUBROUTINE COMPUTES THE NET RATES OF APPLICATION, Q(NT),FOR EACH BLOCK
C   THE COMPUTATION OF Q(NT) SHOULD COME OUT IN ACRE-FT/DAY
C
      BIN  = 0.0
      BOUT = 0.0
      SQVOL = 0.0
      NT = 0
C
      DO 4 I=1,NX
           DO 4 J=1,NY
                IJ = NX*(J-1)+I
                NT = NT+1
                IF (IWAT.EQ.1) GO TO 3
                Q(NT) = 0.0
                Q(NT) = -RPUM(IJ)*DYPM*CPM/DELTAT
```

```
     1                    +PPT(IJ)*DYPT*CPT/DELTAT
     2                    +QBC(IJ)/365.
     3                    -RFREA(IJ)*DYPR/DELTAT
     4                    -RLEAK(IJ)/365.
     5                    +RSEEP(IJ)*DYPS/DELTAT
                SQVOL = SQVOL + Q(NT) - QBC(IJ)/365.
                IF (QBC(IJ).GE.0.0) BIN  = BIN  + QBC(IJ)/365.
                IF (QBC(IJ).LT.0.0) BOUT = BOUT + QBC(IJ)/365.
                GO TO 4
C
 3     Q(IJ) = 0.0
                Q(IJ) = -RPUM(IJ)*DYPM*CPM/DELTAT
     1                  +PPT(IJ)*DYPT*CPT/DELTAT
     2                  +QBC(IJ)/365.
     3                  -RFREA(IJ)*DYPR/DELTAT
     4                  -RLEAK(IJ)/365.
     5                  +RSEEP(IJ)*DYPS/DELTAT
                SQVOL = SQVOL + Q(IJ) - QBC(IJ)/365.
                IF (QBC(IJ).GE.0.0) BIN  = BIN  + QBC(IJ)/365.
                IF (QBC(IJ).LT.0.0) BOUT = BOUT + QBC(IJ)/365.
 4     CONTINUE
       RETURN
       END



/*
//RESIDG EXEC FORTRAN(MAP)
       SUBROUTINE RESIDG (NX,NY,CMATRX,CR,H,IBC,FFAR,HP,NA,NB,
      1                   JAGAIN,IDEBUG,DELTAT)
       DIMENSION CMATRX(NA,NB),CR(1),H(1),IBC(1),FFAR(1),HP(1)
       REAL*8 DC1,DCIC,DCIB,DCID,DCIM,DH1,DHIC,DHIB,DHID,DHIM,DCR,
      1       DE,DEFAR,DCT,DHPIM
C----THIS SUBROUTINE COMPUTES THE RESIDUAL TERM OF THE K-TH ITERATE, DEFINED AS
C             AD*H(I,J)-AN*H(I,J-1)-AS*H(I,J+1)-AW*H(I-1,J)-AE*H(I+1,J)
C             +RHS(IJ) = R(IJ)
C     WHERE RHS IS THE RIGHT HAND SIDE OF THE FINITE DIFFERENCES EQUATION
C             RHS(IJ) = -(PHI(IJ)*DX(IJ)*DY(IJ)/DELTAT)*HP(IJ) + Q(IJ)*43560.
C     HP IS THE WATERTABLE ELEVATION AT TIME LEVEL T
C     THE K-TH ITERATE LIES BETWEEN TIME LEVEL T AND TIME LEVEL T+DELTAT
C
       NXNY = NX*NY
       IB = NY
       IM = IB+1
       IC = IM+1
       ID = NB
       IF (JAGAIN.EQ.1) GO TO 1
       DO 2 J=1,NB
          DO 2 I=1,NA
 2     CMATRX(I,J) = 0.0
       REWIND 1
       READ(1) (CMATRX(I,1),CMATRX(I,IB),CMATRX(I,IC),CMATRX(I,ID),
      1         CMATRX(I,IM),CR(I),I=1,NXNY)
 1     CONTINUE
       NT = 0
C
       DO 10 I=1,NX
          DO 10 J=1,NY
             NT = NT + 1
             IJ = NX*(J-1)+I
             IF (IBC(IJ).GE.3) GO TO 5
               DC1  = CMATRX(NT,1)
               DCIC = CMATRX(NT,IC)
               DCIB = CMATRX(NT,IB)
               DCID = CMATRX(NT,ID)
               DCIM = CMATRX(NT,IM)
               DH1  = 0.
               DHIC = 0.
               DHIB = 0.
               DHID = 0.
               DHIM = H(IJ)
               IF (I.EQ.1) GO TO 11
               DH1  = H(IJ-1)
               DH1 = DHIM - DH1
 11            IF (J.EQ.NY) GO TO 12
               DHIC = H(IJ+NX)
               DHIC = DHIM - DHIC
 12            IF (J.EQ.1) GO TO 13
               DHIB = H(IJ-NX)
               DHIB = DHIM - DHIB
 13            IF (I.EQ.NX) GO TO 14
```

```
                  DHID = H(IJ+1)
                  DHID = DHIM - DHID
 14               DEFAR = EFAR(IJ)
                  DDT   = DELTAT
                  DHPIM = HP(IJ)
                  DCR   = CR(NT)
                  DCR   = DCR*43560.
                  DE    = (DEFAR/DDT)*(DHIM-DHPIM)
                  DCR   = DCR + DE
                  DCR = -(DC1*DH1 + DCIC*DHIC + DCIB*DHIB + DCID*DHID) +
     1                    DCR
                  CR(NT) =-DCR
                  IF (EFAR(IJ).EQ.0.0) CR(NT) = 0.0
                  IF (EFAR(IJ).EQ.0.0) CMATRX(NT,IM)=1.0
                  GO TO 10
  5             CMATRX(NT,1) = 0.
                CMATRX(NT,IB)= 0.
                CMATRX(NT,IC)= 0.
                CMATRX(NT,ID)= 0.
                CMATRX(NT,IM)= 1.0
                CR(NT) = 0.
 10   CONTINUE
C
      IF (IDEBUG.EQ.1)
     1WRITE (6,50)
      IF (IDEBUG.EQ.1)
     1PRINT 60,((CMATRX(NT,1),CMATRX(NT,IB),CMATRX(NT,IM),
     2           CMATRX(NT,IC),CMATRX(NT,ID),CR(NT),NT=1,NXNY)
   50 FORMAT (1H1,9X,12HCMATRX(NT,1),5X,13HCMATRX(NT,IB),3X,
     1        13HCMATRX(NT,IM),5X,13HCMATRX(NT,IC),6X,13HCMATRX(NT,ID),
     2        7X,5HCR(NT),5X,8H(RESIDG)//)
   60 FORMAT (6F18.7)
      RETURN
      END
```

```
/*
//TRANSG EXEC FORTRAN(MAP)
      SUBROUTINE TRANSG (CR,CMATRX,EFAR,HP,Z,CE,CS,IBC,NX,NY,DELTAT,
     1                   NA,NB,IDEBUG)
      DIMENSION CR(1),CMATRX(NA,NB),EFAR(1),HP(1),Z(1),CE(1),CS(1),
     1          IBC(1)
      REAL*8 DAW,DAE,DAN,DAS,DCMAT,DE,DEFAR,DDT
C----THIS SUBROUTINE COMPUTES THE TRANSMISSIVITIES AND SETS THEM UP IN
C----REDUCED MATRIX FORM FOR GAUSS ELIMINATION (BSOLVE)
C
      NXNY = NX*NY
      NA = NXNY
      NB = 2*NY + 1
      DO 1 J=1,NB
         DO 1 I=1,NA
 1    CMATRX(I,J) = 0.0
      IB = NY
      IM = IB + 1
      IC = IM + 1
      ID = NB
      NT = 0
C
      DO 100 I=1,NX
         DO 100 J=1,NY
               IJ = NX*(J-1)+I
               NT = NT + 1
                    AW = 0.0
                    AN = 0.0
                    AE = 0.0
                    AS = 0.0
                    DAW = 0.0
                    DAE = 0.0
                    DAN = 0.0
                    DAS = 0.0
               HIJ = HP(IJ)
               IF (I.EQ.1) GO TO 3
                    CCW = CE(IJ-1)
                    AW=CCW*(AMAX1(HP(IJ),HP(IJ-1))-AMAX1(Z(IJ),Z(IJ-1)))
                    CMATRX(NT,1) = -AW
                    DAW = AW
```

```
 3              IF (I.EQ.NX) GO TO 4
                    CMATRX(NT,ID) = -AE
                    CCE = CE(IJ)
                    AE=CCE*(AMAX1(HP(IJ),HP(IJ+1))-AMAX1(Z(IJ),Z(IJ+1)))
                    DAE = AE
 4              IF (J.EQ.1) GO TO 5
                    CCN = CS(IJ-NX)
                    AN=CCN*(AMAX1(HIJ,HP(IJ-NX))-AMAX1(Z(IJ),Z(IJ-NX)))
                    CMATRX(NT,IB) = -AN
                    DAN = AN
 5              IF (J.EQ.NY) GO TO 6
                    CCS = CS(IJ)
                    AS=CCS*(AMAX1(HIJ,HP(IJ+NX))-AMAX1(Z(IJ),Z(IJ+NX)))
                    CMATRX(NT,IC) = -AS
                    DAS = AS
 6              DDT = DELTAT
                DEFAR = EFAR(IJ)
                DE = DEFAR/DDT
                DCMAT = DAW + DAN + DAS + DAE + DE
                CMATRX(NT,IM) = DCMAT
C---FOLLOWING 'IF' STATEMENT FOR IMPERMEABLE GRID BLOCKS ONLY.
                IF (EFAR(IJ).EQ.0.0) CMATRX(NT,IM)=1.0
C----UP TO THIS POINT CR HAS THE Q VALUES STORED; IN QFIX, Q SHOULD BE STORED
C----AS CR.
                CR(NT) = -CR(NT)
 100  CONTINUE
C
      IF (IDEBUG.EQ.1)
     1CALL MAP (EFAR,4HEFAR,2.,NX,NY,CUMTIM)
      IF (IDEBUG.EQ.1)
     1WRITE (6,50)
      IF (IDEBUG.EQ.1)
     1PRINT 60,(CMATRX(NT,1),CMATRX(NT,IB),CMATRX(NT,IM),
     2          CMATRX(NT,IC),CMATRX(NT,ID),CR(NT),NT=1,NXNY)
   50 FORMAT (1H1,8X,12HCMATRX(NT,1),5X,13HCMATRX(NT,IB),3X,
     1       13HCMATRX(NT,IM),5X,13HCMATRX(NT,IC),6X,13HCMATRX(NT,ID),
     2       7X,6HCR(NT),5X,8H(TRANSG)//)
   60 FORMAT (6F18.7)
      REWIND 1
      WRITE (1) (CMATRX(I,1),CMATRX(I,IB),CMATRX(I,IC),CMATRX(I,ID),
     1           CMATRX(I,IM),CR(I),I=1,NXNY)
      RETURN
      END
```

```
/*
//GAUS1 EXEC FORTRAN(MAP)
      SUBROUTINE GAUS1 (NX,NY,CMATRX,CR,H,IBC,EFAR,HP,NA,NB,
     1                  HCONV,ITER,IDEBUG,CUMTIM,DELTAT,JAGAIN,ICOUNT)
      DIMENSION CMATRX(NA,NB),CR(1),H(1),IBC(1),EFAR(1),HP(1)
      REAL*8 DH,DCR
C-----THE FOLLOWING EQUATION IS SOLVED
C       AD*H'(IJ)-AN*H'(IJ-NX)-AW*H'(IJ-1)-AE*H'(IJ+1)-AS*H'(IJ+NX) = -R(IJ)
C      IN WHICH R IS A RESIDUAL, AND H' IS A CORRECTION VECTOR, H'=H(T+DT)-H(K)
C
C                                                    (IJ-NX)
C
C                THE SUBSCRIPT NOTATION    (IJ-1)    (IJ)   (IJ+1)
C
C                                                    (IJ+NX)
C
      CALL BSOLVE (CMATRX,NA,NB,CR)
      NXNY = NX*NY
      ITER = 0
      NT = 0
C
      DO 5 I=1,NX
           DO 5 J=1,NY
                IJ = NX*(J-1)+I
                NT = NT+1
                IF (ABS(CR(NT)).GT.HCONV) ITER = 1
                DH = H(IJ)
                IF (IBC(IJ).GE.3) CR(NT)=0.0
                DCR = CR(NT)
                DH = DH + DCR
                H(IJ) = DH
                IF (EFAR(IJ).EQ.0.0) H(IJ) = 0.0
 5    CONTINUE
C
```

```
      IF (ITER.EQ.0) GO TO 10
      IF (IDEBUG.EQ.1) CALL MAP (H,4HH    ,2.,NX,NY,CUMTIM)
      GO TO 20
10    CALL MAP (H,4HH    ,2.,NX,NY,CUMTIM)
      JCOUNT = ICOUNT + 1
      IF (JCOUNT.EQ.1)
     1    WRITE (6,11) JAGAIN,JCOUNT
      IF (JCOUNT.EQ.2)
     1    WRITE (6,12) JAGAIN,JCOUNT
      IF (JCOUNT.EQ.3)
     1    WRITE (6,13) JAGAIN,JCOUNT
      IF (JCOUNT.GE.4)
     1    WRITE (6,6) JAGAIN,JCOUNT
6     FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        13H-TH TIME STEP)
11    FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        12HST TIME STEP)
12    FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        12HND TIME STEP)
13    FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        12HRD TIME STEP)
20    CONTINUE
      RETURN
      END
```

```
/*
//BSOLVE EXEC FORTRAN(MAP)
      SUBROUTINE BSOLVE (C,N,M,V)                                        N    1
C  THIS SUBROUTINE SOLVES THE MATRIX BY GAUSS ELIMINATION.               N    2
      DIMENSION C(N,M), V(N)                                             N    3
      LR=(M-1)/2                                                         N    4
      DO 2 L=1,LR                                                        N    5
      IM=LR-L+1                                                          N    6
      DO 2 I=1,IM                                                        N    7
      DO 1 J=2,M                                                         N    8
1     C(L,J-1)=C(L,J)                                                    N    9
      KN=N-L                                                             N   10
      KM=M-I                                                             N   11
      C(L,M)=0.0                                                         N   12
2     C(KN+1,KM+1)=0.0                                                   N   13
      LR=LR+1                                                            N   14
      IM=N-1                                                             N   15
      DO 10 I=1,IM                                                       N   16
      NPIV=I                                                             N   17
      LS=I+1                                                             N   18
      DO 3 L=LS,LR                                                       N   19
      IF (ABS(C(L,1)).GT.ABS(C(NPIV,1))) NPIV=L                          N   20
3     CONTINUE                                                           N   21
      IF (NPIV.LE.I) GO TO 6                                             N   22
4     DO 5 J=1,M                                                         N   23
      TEMP=C(I,J)                                                        N   24
      C(I,J)=C(NPIV,J)                                                   N   25
5     C(NPIV,J)=TEMP                                                     N   26
      TEMP=V(I)                                                          N   27
      V(I)=V(NPIV)                                                       N   28
      V(NPIV)=TEMP                                                       N   29
6     V(I)=V(I)/C(I,1)                                                   N   30
      DO 7 J=2,M                                                         N   31
7     C(I,J)=C(I,J)/C(I,1)                                               N   32
      DO 9 L=LS,LR                                                       N   33
      TEMP=C(L,1)                                                        N   34
      V(L)=V(L)-TEMP*V(I)                                                N   35
      DO 8 J=2,M                                                         N   36
8     C(L,J-1)=C(L,J)-TEMP*C(I,J)                                        N   37
9     C(L,M)=0.0                                                         N   38
      IF (LR.LT.N) LR=LR+1                                               N   39
10    CONTINUE                                                           N   40
      V(N)=V(N)/C(N,1)                                                   N   41
      JM=2                                                               N   42
      DO 12 I=1,IM                                                       N   43
      L=N-I                                                              N   44
      DO 11 J=2,JM                                                       N   45
      KM=L+J                                                             N   46
11    V(L)=V(L)-C(L,J)*V(KM-1)                                           N   47
      IF (JM.LT.M) JM=JM+1                                               N   48
12    CONTINUE                                                           N   49
      RETURN
      END
```

```
/*
//RESIDW EXEC FORTRAN(MAP)
      SUBROUTINE RESIDW (NX,NY,RHS,H,IBC,EFAR,JAGAIN,IDEBUG,DELTAT,
     1                   AN,AS,AE,AW,AO,R,HP,CUMTIM)
      DIMENSION RHS(1),H(1),IBC(1),EFAR(1),AN(1),AS(1),AE(1),AO(1),R(1),
     1          AW(1),HP(1)
      REAL*8 DH1,DHIC,DHIB,DHID,DHIM,DHPIM,DDT,DEFAR,DRHS,DAN,DAS,
     1       DAE,DAW,DAO,DR,DE
      DO 10 I=1,NX
         DO 10 J=1,NY
            IJ = NX*(J-1)+I
            IF (IBC(IJ).GE.3) GO TO 5
             DAN = 0.0
             DAW = 0.0
             DAS = 0.0
             DAE = 0.0
             DH1  = 0.0
             DHIC = 0.0
             DHIB = 0.0
             DHID = 0.0
             DHIM = H(IJ)
            IF (I.EQ.1) GO TO 11
             DH1  = H(IJ-1)
             DH1  = DHIM - DH1
 11          IF (J.EQ.NY) GO TO 12
             DHIC = H(IJ+NX)
             DHIC = DHIM - DHIC
 12          IF (J.EQ.1) GO TO 13
             DHIB = H(IJ-NX)
      DHIB=DHIM-DHIB
   13 IF(I.EQ.NX)GO TO 14
      DHID=H(IJ+1)
      DHID=DHID-DHIM
   14 DHPIM=HP(IJ)
      DDT=DELTAT
      DEFAR=EFAR(IJ)
      DRHS=RHS(IJ)
      DRHS=DRHS*43560.
      DE=(DEFAR/DDT)*(DHIM-DHPIM)
      DRHS=DRHS+DE
      DAN=AN(IJ)
      DAS=AS(IJ)
      DAE=AE(IJ)
      DAW=AW(IJ)
      DR=DAW*DH1+DAS*DHIC+DAN*DHIB+DAE*DHID+DRHS
      R(IJ)=-DR
             IF (EFAR(IJ).EQ.0.0) R(IJ)  = 0.0
             IF (EFAR(IJ).EQ.0.0) AO(IJ) = 1.0
             GO TO 10
 5           AW(IJ) = 0.0
             AS(IJ) = 0.0
             AN(IJ) = 0.0
             AE(IJ) = 0.0
             AO(IJ) = 1.0
             R(IJ)  = 0.0
 10   CONTINUE
      IF (IDEBUG.NE.1) GO TO 20
      CALL MAP (AO,4HAO   ,2.,NX,NY,CUMTIM)
      CALL MAP (AW,4HAW   ,2.,NX,NY,CUMTIM)
      CALL MAP (AE,4HAE   ,2.,NX,NY,CUMTIM)
      CALL MAP (AN,4HAN   ,2.,NX,NY,CUMTIM)
      CALL MAP (AS,4HAS   ,2.,NX,NY,CUMTIM)
      CALL MAP (R,4HR    ,2.,NX,NY,CUMTIM)
 20   CONTINUE
      RETURN
      END
```

```
/*
//TRANSW EXEC FORTRAN(MAP)
      SUBROUTINE TRANSW (RHS,EFAR,HP,Z,CE,CS,IBC,NX,NY,DELTAT,AN,AS,
     1                   AE,AW,AO)
      DIMENSION RHS(1),EFAR(1),HP(1),Z(1),CE(1),CS(1),IBC(1),AW(1),
     1          AE(1),AN(1),AS(1),AO(1)
      REAL*8 DAW,DAE,DAN,DAS,DAO,DE,DEFAR,DDT
      DO 100 I=1,NX
          DO 100 J=1,NY
                IJ = NX*(J-1)+I
                   DAW = 0.0
                   DAE = 0.0
                   DAN = 0.0
                   DAS = 0.0
                   AW(IJ) = 0.0
                   AN(IJ) = 0.0
                   AE(IJ) = 0.0
                   AS(IJ) = 0.0
                IF (I.EQ.1) GO TO 3
                   CCW = CE(IJ-1)
                   AW(IJ) = CCW*(AMAX1(HP(IJ),HP(IJ-1))-AMAX1(Z(IJ),
     1                          Z(IJ-1)))
                   DAW = AW(IJ)
 3              IF (I.EQ.NX) GO TO 4
                   CCE = CE(IJ)
                   AE(IJ) = CCE*(AMAX1(HP(IJ),HP(IJ+1))-AMAX1(Z(IJ),
     1                          Z(IJ+1)))
                   DAE = AE(IJ)
 4              IF (J.EQ.1) GO TO 5
                   CCN = CS(IJ-NX)
                   AN(IJ) = CCN*(AMAX1(HP(IJ),HP(IJ-NX))-AMAX1(Z(IJ),
     1                          Z(IJ-NX)))
                   DAN = AN(IJ)
 5              IF (J.EQ.NY) GO TO 6
                   CCS = CS(IJ)
                   AS(IJ) = CCS*(AMAX1(HP(IJ),HP(IJ+NX))-AMAX1(Z(IJ),
     1                          Z(IJ+NX)))
                   DAS = AS(IJ)
 6              DDT = DELTAT
                DEFAR = EFAR(IJ)
                DE = DEFAR/DDT
                DAO = DAW + DAN + DAS + DAE + DE
                AO(IJ) = DAO
C------Q IS OVERREAD BY RHS
                RHS(IJ) = -RHS(IJ)
 100  CONTINUE
      RETURN
      END
```

```
/*
//LSOR EXEC FORTRAN(MAP)
      SUBROUTINE LSOR (AN,AS,AE,AW,AO,HST,R,A,B,C,D,NX,NY,OMEGA,IDEBUG,
     1                 CUMTIM,H,IBC,SEAR,HP,HCONV,ITER,DELTAT,JAGAIN,
     2                 ICOUNT)
      DIMENSION AN(1),AS(1),AE(1),AW(1),AO(1),HST(1),A(1),B(1),C(1),
     1          D(1),H(1),IBC(1),SEAR(1),HP(1),R(1)
      REAL*8 DH,DHST
      TOL = 0.0001
      TOL = 0.01
      TOL = 0.001
      TOL=0.00001
      NXNY = NX*NY
      LS = 0
      DO 1 K=1,NXNY
1        HST(K) = 0.0
2     ITER = 0
      LS = LS + 1
      DO 8 J=1,NY
           DO 5 I=1,NX
                HSTE = 0.0
                HSTW = 0.0
                HSTS = 0.0
                HSTN = 0.0
                IJ = NX*(J-1) + I
                IF (J.NE.1 ) HSTN = HST(IJ-NX)
                IF (J.NE.NY) HSTS = HST(IJ+NX)
                IF (I.NE.1 ) HSTW = HST(IJ-1 )
                IF (I.NE.NX) HSTE = HST(IJ+1 )
                A(I) = AN(IJ)
                B(I) =-AO(IJ)
                C(I) = AE(IJ)
5               D(I) = OMEGA *(-R(IJ)-AN(IJ)*HSTN - AS(IJ)*HSTS)
     1                 + (OMEGA-1.0)*(AO(IJ)*HST(IJ)-AE(IJ)*HSTE-AW(IJ)
     2                 *HSTW)
                CALL THOMAS (D,A,B,C,D,NX)
C---------THE VECTOR D NOW CONTAINS THE NEW HST VALUES.
           DO 55 I=1,NX
                IJ = NX*(J-1) + I
                IF (ABS(D(I)-HST(IJ)).GT.TOL) ITER = 1
55              HST(IJ) = D(I)
8     CONTINUE
      IF (ITER.EQ.1) GO TO 2
      ITER = 0
      DO 500 IJ=1,NXNY
           IF (ABS(HST(IJ)).GT.HCONV) ITER = 1
           DH = H(IJ)
           IF (IBC(IJ).GE.3) HST(IJ) = 0.0
           DHST = HST(IJ)
           DH = DH + DHST
           H(IJ) = DH
           IF (SEAR(IJ).EQ.0.0)  H(IJ) = 0.0
500   CONTINUE
      PRINT 9,LS
9     FORMAT (10X,15HLSOR ITERATIONS,I3)
      IF (ITER.EQ.0) GO TO 10
      IF (IDEBUG.EQ.1) CALL MAP (H,4HH    ,2.,NX,NY,CUMTIM)
      GO TO 20
10    CALL MAP (H,4HH    ,2.,NX,NY,CUMTIM)
      JCOUNT = ICOUNT + 1
      IF (JCOUNT.EQ.1)
     1    WRITE (6,11) JAGAIN,JCOUNT
      IF (JCOUNT.EQ.2)
     1    WRITE (6,12) JAGAIN,JCOUNT
      IF (JCOUNT.EQ.3)
     1    WRITE (6,13) JAGAIN,JCOUNT
      IF (JCOUNT.GE.4)
     1    WRITE (6,6) JAGAIN,JCOUNT
6     FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        13H-TH TIME STEP)
11    FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        12HST TIME STEP)
12    FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        12HND TIME STEP)
13    FORMAT (1HO,5X,22HNUMBER OF ITERATIONS =,I3,2X,5H  FOR,I3,
     1        12HRD TIME STEP)
20    CONTINUE
      RETURN
      END
```

```
/*
//THOMAS EXEC FORTRAN(MAP)
      SUBROUTINE THOMAS(U,A,B,C,D,NN)
      DIMENSION A(NN),B(NN),C(NN),D(NN),U(NN),W(200),G(200)
C-----THIS SOLVES A TRI-DIAGONAL MATRIX BY THE THOMAS ALGORITHM.
C               TRI-DIAGONAL EQUA IS
CC              A*U(I-1) + B*U(I) + C*U(I+1) = D
C
      W(1) = B(1)
      G(1) = D(1)/W(1)
      DO 60 I=2,NN
      W(I) = B(I) - (A(I)*C(I-1))/W(I-1)
      G(I) = (D(I)-A(I)*G(I-1))/W(I)
   60 CONTINUE
      U(NN) = G(NN)
      DO 70 I=2,NN
      J = NN - I + 1
      U(J) = G(J) - (C(J)*U(J+1))/W(J)
   70 CONTINUE
      RETURN
      END



/*
//BALCOM EXEC FORTRAN(MAP)
      SUBROUTINE BALCOM (NX,NY,H,HP,EFAR,IBC,Z,G,CE,CS,DELTAT,
     1                   CUMTIM,CUMMB,SQVOL,ICOUNT)
      COMMON/STOR/TSTOR,BIN,BOUT
      DIMENSION H(1),HP(1),EFAR(1),IBC(1),Z(1),G(1),CE(1),CS(1)
C-----THIS SUBROUTINE COMPUTES THE MATERIAL BALANCE ERROR
C                    GLOSSARY (OUTFLOW + ,INFLOW - )
C                    -------------------------------
C               QRIV       RIVER OR LAKE (OUTFLOW OF AQUIFER OR INFLOW OF AQUIFER)
C                                          (INFLOW)                (OUTFLOW)
C               BINFL      BOUNDARY INFLOW
C               BOUTFL     BOUNDARY OUTFLOW
C               PTSTOR     TOTAL STORAGE (PREVIOUS TIME LEVEL)
C               TSTOR      TOTAL STORAGE (NEW TIME LEVEL)
C               CSTOR      CHANGE IN STORAGE DURING DELTA-T
C               SQVOL      TOTAL WATER APPLIED (NET WITHDRAWAL)
C               SMB        MATERIAL BALANCE
C               CUMMB      CUMULATIVE MATERIAL BALANCE ERROR
C               CUMPC      CUMULATIVE MATERIAL BALANCE ERROR IN PERCENTAGE
C
      CALL BNDFLO (QRIV,BINFL,BOUTFL,H,HP,CE,CS,IBC,Z,NX,NY)
      PTSTOR = TSTOR
      TSTOR  = 0.0
      SMB    = 0.0
C
      DO 1 I=1,NX
           DO 1 J=1,NY
                IJ = NX*(J-1) + I
                IF (IBC(IJ).GE.3) GO TO 1
                IF (ICOUNT.EQ.0)
     1          PTSTOR = PTSTOR + EFAR(IJ)*(HP(IJ)-Z(IJ))
                TSTOR  = TSTOR  + EFAR(IJ)*( H(IJ)-Z(IJ))
 1    CONTINUE
C
C-----ALL VALUES CONVERTED INTO ACRE-FT FOR DELTAT PERIOD
           IF (ICOUNT.EQ.0) PTSTOR = PTSTOR/43560.
           BIN    = -BIN * DELTAT
           BOUT   = -BOUT* DELTAT
           SQVOL  = -SQVOL * DELTAT
           TSTOR  = TSTOR/43560.
           CSTOR  = TSTOR - PTSTOR
           BOUTFL = BOUTFL * DELTAT /43560.
           BOUTFL = BOUTFL + BOUT
           QRIV   = QRIV   * DELTAT /43560.
           BINFL  = BINFL  * DELTAT /43560.
           BINFL  = BINFL  + BIN
      SMB = SMB + QRIV + BINFL + BOUTFL + CSTOR + SQVOL
      CUMMB = CUMMB + SMB
      CUMPC = (CUMMB/TSTOR) * 100.0
C
      WRITE (6,11) CUMTIM,DELTAT
      WRITE (6,12) QRIV
      WRITE (6,13) BINFL
      WRITE (6,14) BOUTFL
      WRITE (6,15) SQVOL
      WRITE (6,16) PTSTOR
```

```
      WRITE (6,17) TSTOR
      WRITE (6,18) CSTOR
      WRITE (6,19) SMB
      WRITE (6,20) CUMPC
11    FORMAT (1H0,29X,31HMATERIAL BALANCE AT THE END OF ,F6.1,18H DAYS(T
     1IME STEP = ,F5.1,7H  DAYS)/30X,66H==========
     2==========)
12    FORMAT (1H0,15X,72HFLOW FROM RIVER OR LAKE INTO(-)OR OUT OF(+) AQU
     1IFER(AC-FT PER TIME STEP),3X,F12.2)
14    FORMAT (1H0,15X,57HFLOW OUT OF AQUIFER ALONG BOUNDARIES(AC-FT PER
     1TIME STEP),18X,F12.2)
13    FORMAT (1H0,15X,55HFLOW INTO AQUIFER ALONG BOUNDARIES(AC-FT PER TI
     1ME STEP),20X,F12.2)
15    FORMAT (1H0,15X,69HNET WITHDRAWAL(PUMPING,PRECIPITATION,PHREATOPHY
     1TE LOSS,LEAKAGE,ETC.),7X,F12.2)
16    FORMAT (1H0,15X,45HTOTAL STORAGE UP TO PREVIOUS TIME STEP(AC-FT),
     130X,F12.2)
17    FORMAT (1H0,15X,32HTOTAL STORAGE UP TO DATE (AC-FT),43X,F12.2)
18    FORMAT (1H0,15X,38HCHANGE IN STORAGE(AC-FT PER TIME STEP),37X,
     1F12.2/91X,12H============)
19    FORMAT (1H0,15X,24HMATERIAL BALANCE (AC-FT),51X,F12.2)
20    FORMAT (1H0,15X,26HCUMULATIVE ERROR (PERCENT),49X,F12.2,1H%)
      DO 8 I=1,NX
          DO 8 J=1,NY
               IJ = NX*(J-1)+I
               HP(IJ) = H(IJ)
8     CONTINUE
      RETURN
      END


/*
//BNDFLO EXEC FORTRAN(MAP)
      SUBROUTINE BNDFLO (QRIV,BINFL,BOUTFL,H,HP,CF,CS,IBC,Z,NX,NY)
      DIMENSION H(1),HP(1),CF(1),CS(1),IBC(1),Z(1)
C
      QRIV   = 0.0
      BINFL  = 0.0
      BOUTFL = 0.0
C
      DO 100 I=1,NX
          DO 100 J=1,NY
               IJ = NX*(J-1) + I
               IF (IBC(IJ).LT.3) GO TO 100
                    QW = 0.0
                    QE = 0.0
                    QN = 0.0
                    QS = 0.0
                    AW = 0.0
                    AE = 0.0
                    AN = 0.0
                    AS = 0.0
               HIJ = HP(IJ)
               IF (I.EQ.1) GO TO 3
                    CCW = CF(IJ-1)
                    AW=CCW*(AMAX1(HP(IJ),HP(IJ-1))-AMAX1(Z(IJ),Z(IJ-1)))
                    QW = AW*(H(IJ)-H(IJ-1))
3              IF (I.EQ.NX) GO TO 4
                    CCE = CF(IJ)
                    AE=CCE*(AMAX1(HP(IJ),HP(IJ+1))-AMAX1(Z(IJ),Z(IJ+1)))
                    QE = AE*(H(IJ)-H(IJ+1))
4              IF (J.EQ.1) GO TO 5
                    CCN = CS(IJ-NX)
                    AN=CCN*(AMAX1(HIJ,HP(IJ-NX))-AMAX1(Z(IJ),Z(IJ-NX)))
                    QN = AN*(H(IJ)-H(IJ-NX))
5              IF (J.EQ.NY) GO TO 6
                    CCS = CS(IJ)
                    AS=CCS*(AMAX1(HIJ,HP(IJ+NX))-AMAX1(Z(IJ),Z(IJ+NX)))
                    QS = AS*(H(IJ)-H(IJ+NX))
6              IF (IBC(IJ).EQ.4) GO TO 7
               IF (QE.GT.0.0) BINFL  = BINFL  + QE
               IF (QN.GT.0.0) BINFL  = BINFL  + QN
               IF (QW.GT.0.0) BINFL  = BINFL  + QW
               IF (QS.GT.0.0) BINFL  = BINFL  + QS
               IF (QW.LT.0.0) BOUTFL = BOUTFL + QW
               IF (QE.LT.0.0) BOUTFL = BOUTFL + QE
               IF (QN.LT.0.0) BOUTFL = BOUTFL + QN
               IF (QS.LT.0.0) BOUTFL = BOUTFL + QS
               GO TO 100
7              QRIV = QRIV -(QW + QE + QN + QS)
```

```
  100  CONTINUE
C
      BINFL  = -BINFL
      BOUTFL = -BOUTFL
      RETURN
      END



/*
//MAP EXEC FORTRAN(MAP)
      SUBROUTINE MAP (ARRAY,NAME,FORM,NX,NY,DAYS)
      DIMENSION ARRAY(2),LABEL(20)
C
      DATA KCS /'CS  '/,KCE /'CE  '/,KH  /'H   '/,KPPT/'PPT '/,
     1     KPUM/'PUM '/,KERE/'EREA'/,KLEA/'LEAK'/,KSEE/'SEEP'/,
     2     KQBC/'QBC '/,KZ  /'Z   '/,KG  /'G   '/,KFKX/'FKX '/,
     3     KFKY/'FKY '/,KDX /'DX  '/,KDY /'DY  '/,KPHI/'PHI '/,
     4     KEFA/'EFAR'/,KHP /'HP  '/
C
      LMAX = 20
      IF (FORM.EQ.2) LMAX = 10
      N1 = 1
      LENGTH = NX
      IF (NAME.EQ.KCE ) PRINT  10,DAYS
      IF (NAME.EQ.KCS ) PRINT  20,DAYS
      IF (NAME.EQ.KPPT) PRINT  30,DAYS
      IF (NAME.EQ.KPUM) PRINT  40,DAYS
      IF (NAME.EQ.KERE) PRINT  50,DAYS
      IF (NAME.EQ.KLEA) PRINT  60,DAYS
      IF (NAME.EQ.KSEE) PRINT  70,DAYS
      IF (NAME.EQ.KQBC) PRINT  80,DAYS
      IF (NAME.EQ.KFKX) PRINT  90,DAYS
      IF (NAME.EQ.KFKY) PRINT 110,DAYS
      IF (NAME.EQ.KDX ) PRINT 120,DAYS
      IF (NAME.EQ.KDY ) PRINT 130,DAYS
      IF (NAME.EQ.KPHI) PRINT 140,DAYS
      IF (NAME.EQ.KG  ) PRINT 150,DAYS
      IF (NAME.EQ.KZ  ) PRINT 160,DAYS
      IF (NAME.EQ.KH  ) PRINT 170,DAYS
      IF (NAME.EQ.KHP ) PRINT 180,DAYS
      IF (NAME.EQ.KEFA) PRINT 190,DAYS
 10   FORMAT (1H0/40X,21HCE  MAP                    ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 20   FORMAT (1H0/40X,21HCS  MAP                    ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 30   FORMAT (1H0/40X,21HPRECIPITATION  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 40   FORMAT (1H0/40X,21HPUMPING RATE  MAP          ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 50   FORMAT (1H0/40X,21HREPEAT. WATER  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 60   FORMAT (1H0/40X,21HWATER LEAKING  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 70   FORMAT (1H0/40X,21HWATER SEEPING  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 80   FORMAT (1H0/40X,21HINFLOW (OUTFLOW)  MAP,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
 90   FORMAT (1H0/40X,21HHYDRO. CONDUC.-X  MAP,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
110   FORMAT (1H0/40X,21HHYDRO. CONDUC.-Y  MAP,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
120   FORMAT (1H0/40X,21HX-INCREMENT  MAP           ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
130   FORMAT (1H0/40X,21HY-INCREMENT  MAP           ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
140   FORMAT (1H0/40X,21HEFF. POROSITY  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
150   FORMAT (1H0/40X,21HGROUND ELEV.  MAP          ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
160   FORMAT (1H0/40X,21HBEDROCK ELEV.  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
170   FORMAT (1H1/40X,21HWATER-TABLE  MAP           ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
180   FORMAT (1H0/40X,21HPREVIOUS W.T.  MAP         ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
190   FORMAT (1H0/40X,21HEFFECTIVE AREA  MAP        ,14X,7HTIME = ,
     1        F10.4,2X,4HDAYS/)
    2 CONTINUE
      IF (LENGTH.GT.LMAX) LENGTH = LMAX
      DO 3 I=1,LENGTH
```

```
  3 LABEL(I) = N1-1+I
    IF (FORM.EQ.1) PRINT 11,(LABEL(I),I=1,LENGTH)
    IF (FORM.EQ.2) PRINT 12,(LABEL(I),I=1,LENGTH)
    IF (FORM.EQ.3) PRINT 11,(LABEL(I),I=1,LENGTH)
 11 FORMAT (1X,1X,20I6)
 12 FORMAT (1X,I9,9I12)
    N3 = N1 - NX
    DO 100 J=1,NY
    N3 = N3 + NX
    N4 = N3+LENGTH-1
    IF (FORM.EQ.1) PRINT 101,J,(ARRAY(IJ),IJ=N3,N4)
    IF (FORM.EQ.2) PRINT 102,J,(ARRAY(IJ),IJ=N3,N4)
    IF (FORM.EQ.3) PRINT 103,J,(ARRAY(IJ),IJ=N3,N4)
101 FORMAT (1X,I2,20F6.0)
102 FORMAT (1X,I2,10E12.5)
103 FORMAT (1X,I2,2X,20F6.3)
100 CONTINUE
    LENGTH = NX - N1 - LENGTH + 1
    IF (LENGTH.LE.0) GO TO 99
    N1 = N1 + LMAX
999 FORMAT (1X,///)
    GO TO 2
 99 RETURN
    END


/*
//SYSA92 ACCESS SDSPHS
// EXEC LNKEDT(MAP)
      PHASE PHASECOM,ROOT
      INCLUDE MAIN,L
      INCLUDE CLAY1,L
      INCLUDE SINGLE,L
      INCLUDE ARRAYS,L
      INCLUDE VALUES,L
      INCLUDE COEF,L
      INCLUDE DISTRI,L
      INCLUDE QFIX,L
      INCLUDE BALCOM,L
      INCLUDE BNDFLD,L
      INCLUDE MAP,L
      PHASE PHASEGAU,*
      INCLUDE TRANSG,L
      INCLUDE RESIDG,L
      INCLUDE GAUS1,L
      INCLUDE BSOLVE,L
      PHASE PHASEMAT,PHASEGAU
      INCLUDE TRANSW,L
      INCLUDE RESIDW,L
      INCLUDE LSQR,L
      INCLUDE THOMAS,L
/*
// EXEC
CMNT
NCOM                  44
OMEGA      1.5
IWAT
IGAU
HCONV      0.01
DEBUG
DELTAT     91.25
NX                    15
NY                    26
ENDS
CMNT
CMNT
CMNT
CMNT
CMNT
H                      4000.
FREA
QBC
SEEP
LEAK
Z                      3900.
FKX                    40.
FKY                    40.
PHI                    0.15
G                      4100.
PPT
DY         1.
```

31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.	31680.	31680.
63360.	63360.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	31680.		

DX	1.					
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						
31680.	31680.	31680.	31680.	31680.	31680.	31680.
31680.	31680.	31680.	31680.	63360.	63360.	63360.
95040.						

```
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
31680.   31680.   31680.   31680.   31680.   31680.   31680.
31680.   31680.   31680.   31680.   63360.   63360.   63360.
95040.
IBC       1.
         1        1        1        1        1        1        1
         1        1        1        1        1        1        1
         1
         1

         1
         1

         1
         1

         1
         1

         1
         1

         1
         1

         1
         1
```

```
              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1

              1
              1       1       1       1       1       1       1
              1       1       1       1       1       1       1
              1
ENDA
YPT
                                                                   YPT     1
                                                                   YPT     2
YPM
0.0833333 0.0833333 0.0833333 0.0833333 0.0833333 0.0833333 0.0833333 YPM     1
0.0833333 0.0833333 0.0833333 0.0833333 0.0833333                     YPM     2
YPR
                                                                   YPR     1
                                                                   YPR     2
YHH
                                                                   YHH     1
                                                                   YHH     2
CPT
CPM        1.
ENDV
```