1990

A computer model for the management of contaminant plumes and site dewatering.

Robert J. MacLean Hyde
University of Windsor

A Computer Model for the Management of Contaminant Plumes and Site Dewatering

by
Robert J. MacLean Hyde

A Thesis Submitted to
the Faculty of Graduate Studies and Research
through the Department of Geology
in Partial Fulfillment of the Requirements
for the Degree of
Master of Applied Science
at The University of Windsor

Windsor, Ontario, Canada
1990

National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

ISBN 0-315-65183-0

ABSTRACT

The objective of this thesis was to develop an efficient user-friendly computer program that would determine optimum scenarios to dewater complex excavations or to control and capture contaminant plume migration by manipulating groundwater elevations. The computer model presented here utilizes a finite difference approximation of the groundwater flow continuity equation in an aquifer to predict water table elevations and the Todd (1980) equation to calculate velocity dependent capture zones for pumping wells. Most groundwater flow models that presently may be used for dewatering or contaminant plume control applications calculate the potentiometric head or velocity flow paths in an aquifer system as a result of the placement of manually selected pumping wells within a hydrogeologically variable environment. Utilization of the existing models for dewatering or plume control applications would require many simulation iterations accompanied by hand calculations to arrive at the best arrangement of wells. The model presented in this thesis goes one step further by having a series of wells, with variable pumping rates, advance through designated areas of a finite difference grid system discretizing the aquifer. Wells advance through the grid to new nodal configurations and an optimum well configuration solution is found when prescribed conditions are fulfilled. In a dewatering simulation, these conditions are in the form of predesignated

control point potentiometric surface elevations that must be matched. In a plume management scenario, the conditions are fulfilled when a given capture well configuration maximizes the coverage of the contaminant plume. In addition to utilizing pumping and recharge wells to manipulate groundwater elevations, the model is able to simulate recharging ditches, collection ditches, induced infiltration from surface streams and ponds, as well as areas of no flow due to complex geology or foundation structures.

The computer model has been written in Turbo Pascal Version 5.0, and is able to run on an IBM-XT, AT, and 386 personal computer or equivalent clone. It is recommended that the computer system have 2 megabyte s of RAM memory and a math co-processor. While input data preparation time is nominal, dewatering simulation times can take over an hour depending upon the computer system speed and the complexity of the problem. Plume simulation times are usually less than 15 minutes.

The program has been shown to operate with less than 5% error in calculating the groundwater potentiometric surface elevations. The model is applied to two field case studies involving a dewatering and plume capture application. The results of the case study applications indicate that this model is an effective planning tool for the pre-design stages of field work which could save the practicing groundwater engineer valuable time.

ACKNOWLEDGEMENTS

It is necessary, before this report can be considered complete, to acknowledge the role of those who contributed and supported me throughout the various stages of this project. Foremost, I would like to express my sincere gratitude to my thesis advisor, Dr. Michael G. Sklash, whose guidance and much needed patience have been greatly appreciated. I would like to thank Mr. William Lenson who took the time to impress upon me the finer points of creative computer programming and the art of somnambulant modeling. I would like to acknowledge the contributions of Mr. Thomas A. Prickett, who expressed to me that all was not in vain and allowed me to borrow from his wealth of groundwater modelling algorithms. I would like to express my sincerest gratitude to my parents, Mr. and Mrs. Stan &Julia Hyde whose support and love made it all seem worthwhile. Finally, I would like to thank my wife, Kathryn, who turned my insane gibberish into coherent technical prose, and without her love, understanding, and typewriting skills, I would still be trying to put together all of the pieces.

Table of Contents

List of Figures

Page No.

List of Tables

Page No.

List of Appendices

List of Symbols

Aa	Nodal area	(L^2)
AA	Coordinate function matrix	
B	Coordinate function matrix	
BB	Coordinate function matrix	
BOT	Aquifer base elevation	(L)
CC	Coordinate function matrix	
CH	Aquifer top elevation	(L)
D	Present head difference	(L)
DD	Coordinate function matrix	
Delta	Initial time increment	(T)
Dl	Past head difference	(L)
F	Predicted future head difference	(L)
G	Coordinate function matrix	
h	Piezometric head	(L)
ho	Initial piezometric head	(L)
i	Rectangular coordinates	
I	Hydraulic gradient	(Dimensionless)
j	Rectangular coordinates	
K	Hydraulic conductivity	(LT^{-1})
m	Aquitard thickness	(L)

P	Aquitard vertical permeability	(LT^{-1})
PERM	Aquifer permeability	(LT^{-1})
Q	Groundwater discharge or recharge	(L^3T^{-1})
Q_L	Leakage rate	(L^3T^{-1})
Q_I	Induced infiltration rate	(L^3T^{-1})
r	Stagnation distance	(L)
Ra	Aquitard leakage factor	(L^3T^{-1})
RD	Stream bed base elevation	(L)
RH	Source bed head elevation	(L)
Rs	Induced infiltration factor	(L^3T^{-1})
S	Storage coefficient	(Dimensionless)
SF1	Artesian storativity factor	(Dimensionless)
SF2	Water table storativity factor	(Dimensionless)
T	Transmissivity	(L^2T^{-1})
t, time	time increment	(T)
V_d	Darcy velocity	(LT^{-1})
$V_{pumping}$	Velocity towards pumping well	(LT^{-1})
Wc	Capture width	(L)
x	Global rectangular coordinates	
y	Global rectangular coordinates	
Δt	time increment	(T)
Øe	effective porosity	(Dimensionless)
Ω	Time increment multiplier	(Dimensionless)

1.0 INTRODUCTION

During the last decade, water resource research has become almost synonymous with computer modelling (Plate and Duckstein, 1988). The sophistication of groundwater models has increased to such an extent that their development is inhibited only because of our incomprehension of the many flow and transport mechanisms. Groundwater modelers have long since surpassed the difficulties associated with modelling flow in spatially complex geologic conditions as well as quantifying the contaminant transport mechanisms of dispersion, diffusion, and chemical retardation (sorption, precipitation, radioactive decay, and ion exchange). Some of the most complex algorithms currently being developed are used to model the transport and fate of non-point source organic compounds associated with episodes of groundwater contamination by agricultural pesticides.

This direction of modelling will be useful to researchers since it will enable them to develop a better understanding of the processes involved in contaminant transport. Attempts to characterize the observed temporal changes in dispersion for a particular contaminant species by complex algorithms may be useful in explaining the flow and transport mechanisms which may have created the field conditions (Kinzelbach et al., 1988). However, the practicality of these models to a practicing groundwater specialist is questionable since there is a need for validation of the contaminant parameters through large scale field experiments (Woolhiser et al., 1989).

Complex computer models often are limited in scope and are impractical to consulting engineers, who have to balance the benefits of the program as a prediction tool with the projects budget constraints. In practice, many disadvantages that exist when trying to utilize complex computer programs are:

i) In the development of algorithms to successfully match field scenarios, the physical and geochemical parameters governing contaminant migration are poorly understood, and may be used to best advantage in sensitivity analysis in which the uncertain parameters are varied (Frind et al., 1987). The tendency to oversimplify the mechanisms of contaminant transport and flow through the application of 'black box' equations, inhibits our understanding of the processes taking place. This oversimplification also reduces the dynamics of the model to apply to varying field scenarios.

ii) Complex models require empirical parameters which have no physical basis and have to be determined through experimentation. These parameters tend to have unknown probability distribution functions (Kinzelbach et al., 1988).

iii) Complex models require a very complete and thorough data base (Plate and Duckstein, 1988).

iv) The greatest disadvantage of complex field models are their limited applications (Plate and Duckstein, 1988).

v) Preparation of input data for numerical codes can take a long time. Even if one wishes to solve a simple problem which is managable with analytical or semi-analytical methods, far greater time is needed to prepare the input data for a numerical code (Javandel et al., 1984).

1.1 Objectives and Scope

Computer models utilized by practicing groundwater specialists are generally used as pre-design or planning tools. The models should provide several optimum scenarios from which the groundwater specialist can choose the best solution. The main objectives of this thesis are:

i) To develop a practical computer software package that will aid the hydrogeological field practitioner in the pre-design stages of planning a dewatering operation for complex excavations or designing a management system to capture or prohibit contaminant plume migration in a groundwater flow system.

ii) To design a computer program which can be applied to a wide variety of field studies and conditions and achieve its solution through the application of basic principles of groundwater flow.

iii) To produce a user-friendly computer software package that addresses the shortcomings noted in Chapter 1.0 of some existing models, through several means: a minimal data base requirement; the development of a user-friendly graphic menu system; the implementation of a spreadsheet procedure to reduce data management time and facilitate incorporation of digitized ASCII formatted map information; and designing the program to have the ability to create ASCII formatted output files that can be easily contoured by the many software contouring packages in existence.

The program presented in this thesis was designed to produce a solution based upon basic equations of flow to enable the model to be universally applicable to most field problems. Therefore, some field conditions may exist which are best described by more complex algorithms of fluid flow and may only be approximated by this model.

1.2 Thesis Organization

The thesis is organized into eight chapters and eight appendices. The contents of these chapters and appendices are as follows:

- Chapter 1 is an introductory chapter that presents the need for this type of model and outlines the objectives and scope of this thesis
- Chapter 2 describes how this computer simulation enables the user to plan a dewatering system.
- Chapter 3 describes how this computer simulation enables the user to effectively develop a capture and remediation plan for the clean up of contaminated groundwater.
- Chapter 4 provides details on the mathematical background upon which this computer model is based.
- Chapter 5 contains a screen by screen description of the model design and outlines the types of data input requirements.
- Chapter 6 presents three case studies to which this model has been applied and presents the simulations results.
- Chapter 7 discusses the results of the model's applications to the three field studies and comments upon the model's efficiency and usefulness.
- Chapter 8 provides concluding remarks and recommendations for further study.
- Appendix A presents a copy of the computer model's printout.
- Appendix B contains the computer outputs from the finite difference program comparisons discussed in Chapter 6.1.
- Appendix C is a background summary of the dewatering case study discussed in Chapter 6.2.

- Appendix D summarizes the program's input for the dewatering case study.
- Appendix E contains the program's solution summary output for the dewatering case study.
- Appendix F is a background summary of the contaminant plume case study discussed in Section 6.3.
- Appendix G summarizes the program's input for the contaminant plume case study.
- Appendix H is the program's solution summary output for the contaminant plume case study.

2.0 DEWATERING

The design of a well system to lower the groundwater potentiometric surface for construction or development purposes becomes complex when considering the wide variety of stresses upon an aquifer system and the number of controlling water table elevations that may exist. The existence of hydrodynamic stresses such as leaky artesian conditions, induced infiltration from surface water bodies, barriers to flow, and aquifer recharge all affect how an aquifer will respond to a pumping well field (Driscoll,1986). In addition, a number of controlling water table elevations may need to be maintained such as: the maximum allowable water table elevations within an excavation; the minimum allowable water table elevation within the pumping well that must be maintained so as not to burn out the pump, or a minimum water table elevation that may need to be maintained by aquifer recharge so that a nearby water supply or surface stream is not dewatered.

Presently, to design a dewatering study using existing computer models, the practicing groundwater specialist must iteratively run the groundwater model to calculate the groundwater's potentiometric surface for numerous manually selected well locations and pumping rates (Driscol, 1986). The model presented in this thesis takes this process one step further and advances wells through a grid discretizing the aquifer and automatically determines the well configuration that maintains user specified control elevations around the site.

This computer simulation utilizes a modified finite difference approximation of the two dimensional equation of aquifer flow (described in detail in Section 4.1) which can incorporate all of the previously noted complexities that may exist within an aquifer and determines the optimum pumping and recharge well configurations that may exist to maintain prescribed control point water table elevations. The user first characterizes the hydrogeology of the aquifer system, as described in detail in Chapter 5.0, and attaches aquifer characterizing hydrogeological parameters to the nodes of a finite difference grid that discretizes the aquifer system. Pumping wells and recharge wells are specified in addition to a grid bounded search area for each well.

Upon commencement of the simulation, the wells automatically progress through the finite difference grid within their specified grid bounded search areas, in such a manner, so that all possible well configurations have been utilized. For each well configuration, a new water table head map is produced and the desired control elevations are compared to calculated water table elevations at the control point locations. This process continues until a suitable match between the calculated water table elevations and the prescribed control point elevations have been reached. Upon achievement of a successful match, the solution is transferred to an output file. The user then has the option to continue the search for another possible solution, or to quit the program. Subsequent simulations may be run to assess the systems sensitivity to different pumping rates.

3.0 PLUME MANAGEMENT

Presently, practising groundwater specialists must use a combination of hand calculations and computer flow model simulations to design aquifer restoration systems. The groundwater specialist would utilize a flow model to predict the potentiometric surface elevations of the groundwater after simulating the placement of a capture well gallery (Driscoll, 1986). Subsequently, well capture zones would be determined by assessing the drawdown cones of the capture wells determined from the flow model results (Driscoll, 1986), or more appropriately, by solving the Todd equation (Keely and Tsang, 1983) described in more detail in Chapter 4. The model presented in this thesis will allow the user to determine optimum capture well configuration designs with greater flexibility and efficiency. Similar to the grid search procedures described in Chapter 2.0, the plume management portion of this program advances wells through user specified grid bounded search areas to determine the best well configurations to capture a plume. In addition, the program calculates the potentiometric surface of the water table for the best well configuration determined.

As the capture wells are incrementally moved through their grid bounded search areas, the program utilizes a modified Todd equation (see Chapter 4) to determine the velocity governed capture zone for each pumping well. While the program advances the wells through all the possible well configurations, the configuration that captures the largest portion of the most heavily contaminated area of the contaminant plume is determined. This procedure is accomplished by drawing an ellipse around each pumping well at each well location during the search, the radii of which have been calculated to be the capture zone radii (discussed in greater detail in

Chapter 4). Coverage of the plume by the pumping wells is assessed by comparing the areal coverage of each well's capture zone to the contaminant plume characterizing array. The resulting water table elevation distribution is then calculated for the applied capture stresses. After completing the plume capture calculations, a summary of the solution is transferred to an output file where it can be viewed on the screen or sent to a printer.

The program simulation can easily be repeated, subsequent to data manipulation, to allow the following questions to be answered:

i) Would the capture efficiencies of the pumping wells improve if recharge wells were added to the system to create hydraulic divides within the aquifer?

ii) What effect would increasing the pumping rate have upon the well's capture radius?

iii) What pumping rate would maximize the well's capture zone and minimize the wells drawdown cone?

4.0 MATHEMATICAL BACKGROUND

The computer model's output is derived from the solution of two mathematical equations:

i) a finite difference approximation of the nonsteady-state, two-dimensional groundwater flow equation that is used to predict an aquifer's potentiometric surface resulting from an applied stress field, and;

ii) the Todd equation (Keely and Tsang, 1983) that is used to define the areal limits of a pumping wells 'capture zone' or cone of influence so that determinations may be made regarding the ability of a given well configuration to effectively recover a contaminant plume.

The following sections describe the mathematical equations used in the model.

4.1 Theory of Finite Difference Approximations

The partial differential equation governing the nonsteady-state, two-dimensional flow of groundwater in an artesian, nonhomogeneous, isotropic aquifer is (Freeze and Cherry, 1979):

$$\frac{\partial \{T\,\partial h/\partial x\}}{\partial x} + \frac{\partial \{T\,\partial h/\partial y\}}{\partial y} = S\frac{\partial h}{\partial t} + Q \qquad (4.1)$$

where:

T = aquifer transmissivity
h = head
t = time
S = aquifer storage coefficient
Q = net groundwater withdrawal rate per unit area
x,y = rectangular coordinates

The following are assumptions that have been made in the development of equation 4.1 (Konikow and Bredehoeft, 1978):

i) Darcy's law is valid and hydraulic head gradients are the only significant driving mechanism for fluid flow;

ii) The porosity and hydraulic conductivity of the aquifer are constant with time, and porosity is uniform in space;

iii) Gradients of fluid density, viscosity, and temperature do not affect the velocity distribution;

iv) No chemical reactions occur that affect the fluid or aquifer properties;

v) Vertical variations in head are negligible;

There is no exact or mathematical solution to equation 4.1, however an approximate numerical solution can be obtained by substituting finite difference approximations for the equations differential components (Prickett and Lonnquist, 1971). The differentials ∂x and ∂y are approximated by the finite lengths Δx and Δy, the differential ∂t is approximated by a finite time period Δt, and the aquifer is discretized into finite volumes having dimensions $m^* \Delta x * \Delta y$, where m is the aquifer thickness, with T, S and h varying from one finite volume to the next (Walton, 1984).

The application of finite difference flow equations to an aquifer system to study site hydrodynamics is initiated by discretizing the aquifer into finite volumes through the superposition of a rectangular grid system (Figure 4.1) over a schematic of the aquifer. In order to achieve a high degree of accuracy, it is prudent to make the discretizing areas of the square grid ($\Delta x \Delta y$) small in comparison to the areal extent of the groundwater system (Prickett and Lonnquist, 1971). The intersections of two grid lines are called

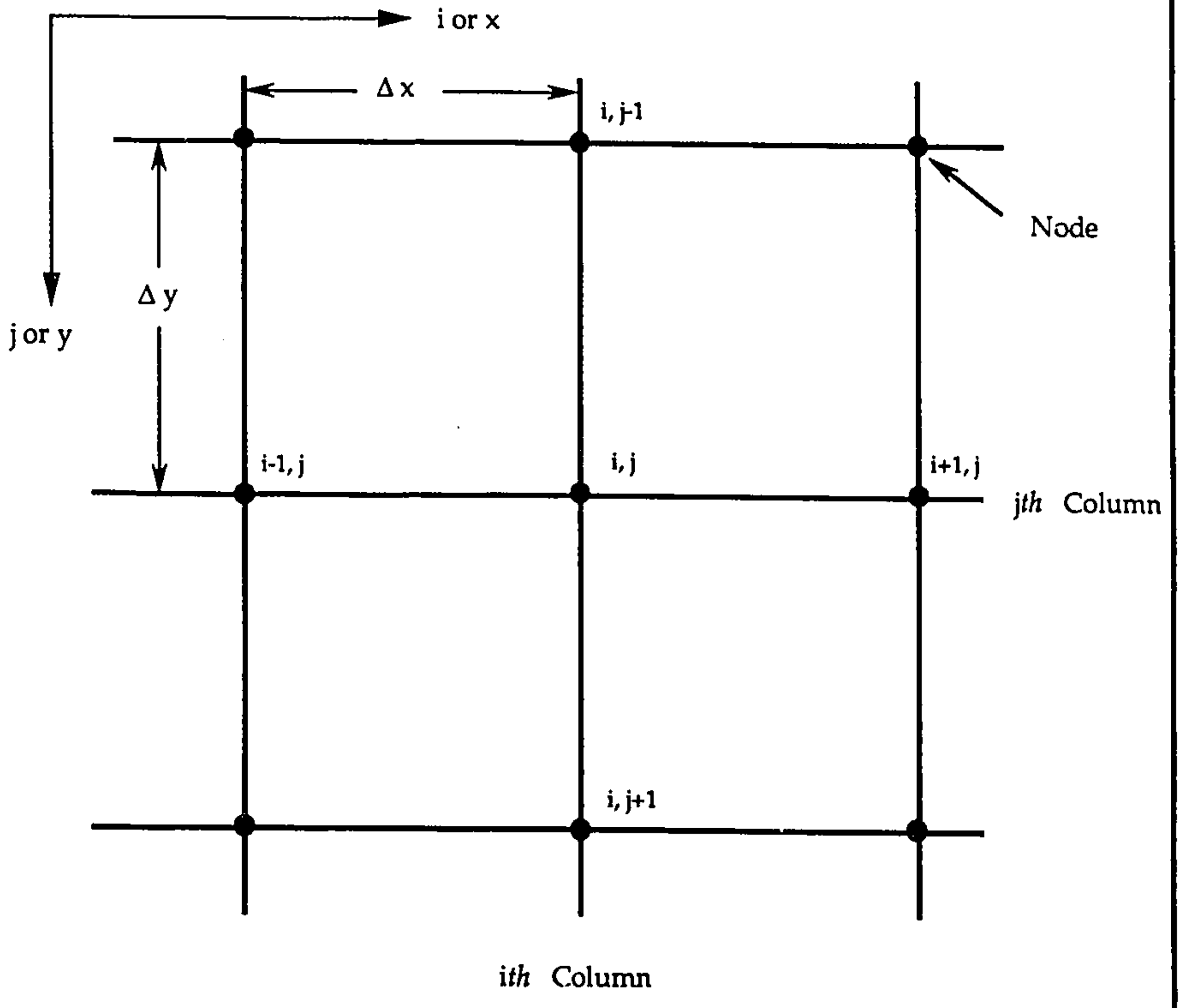


Figure 4.1

Portion of Finite Difference Grid

nodes and have coordinates indexed using the i,j notation such that a column (i) and a row (j) are colinear with the x and y directions, respectively (Prickett and Lonnquist, 1971).

The finite difference form of Equation 4.1 for a non-leaking artesian aquifer system having a production well, with a constant withdrawal rate, at node i,j (Figure 4.1) may be written as (Pinder and Bredehoeft, 1968):

$$T_{i-1,j,2}\,(h_{i-1,j} - h_{i,j})/\Delta x^2 + T_{i,j,2}\,(h_{i+1,j} - h_{i,j})/\Delta x^2 + T_{i,j,1}\,(h_{i,j+1} - h_{i,j})/\Delta y^2$$
$$+T_{i,j-1,1}\,(h_{i,j-1} - h_{i,j})/\Delta y^2 = S_{i,j}(h_{i,j} - ho_{i,j})/\Delta t + Q_{i,j} / \Delta x\Delta y \qquad (4.2)$$

where:

$S_{i,j}$ = aquifer storativity within the vector volume m$\Delta x\Delta y$ centred at node i,j (dimensionless).

$T_{i-1,j,2}$ = aquifer transmissivity within the vector volume between nodes i,j and i-1,j (L^2T^{-1}).

$T_{i,j,2}$ = aquifer transmissivity within the vector volume between nodes i,j and i+1,j (L^2T^{-1}).

$T_{i,j-1,1}$ = aquifer transmissivity within the vector volume between nodes i,j and i,j-1 (L^2T^{-1}).

$T_{i,j,1}$ = aquifer transmissivity within the vector volume between nodes i,j and i,j+1 (L^2T^{-1}).

hi,j = calculated head at the end of a time increment measured from an arbitrary reference level at node i,j (L).

$h_{i-1,j}$ = calculated head at the end of a time increment measured from an arbitrary reference level at node i-1,j (L).

$h_{i+1,j}$ = calculated head at the end of a time increment measured from an arbitrary reference level at node i+1,j (L).

$h_{i,j+1}$ = calculated head at the end of a time increment measured from an arbitrary reference level at node i,j+1 (L).

$h_{i,j-1}$ = calculated head at the end of a time increment measured from an arbitrary reference level at node i,j-1 (L).

$ho_{i,j}$ = calculated head at node i,j at the end of the previous time increment Δt (L).

Qi,j = constant withdrawal rate (L^3T^{-1}).

Δt = time increment elapsed since last calculation of heads (T).

The principle unknown of Equation 4.2 is $h_{i,j}$, the calculated head. Equation 4.2 will calculate the hydrostatic heads in the aquifer system after the application of any hydrodynamic stress. Hydrodynamic nodal stresses such as leakage through an aquitard, leakage from a stream or lake bed, evapotranspiration, and production well withdrawal are all handled through the flux term $Q_{i,j} / \Delta x \Delta y$.

Every node of the finite-difference grid has an equation similar to Equation 4.2. In order to solve the large number of simultaneous algebraic equations efficiently, numerical techniques have been developed for the digital computer. The methodology utilized to solve the simultaneous equations in this model is known as the modified iterative alternating direction implici' method (MIADIM), and was developed by Prickett and

Lonnquist (1971). This simultaneous equation solution technique optimizes the works of past modelers and mathematicians and has proven successful in groundwater flow applications (Prickett and Lonnquist, 1971). The technique is based on a combination of the alternating direction implicit method of Peaceman and Rachfford (1955), the Gauss-Seidel iterative method (James et al., 1977) and a special predictor-corrector method (Prickett and Lonnquist, 1971).

In brief, the MIADIM first reduces a large set of simultaneous equations down to a number of smaller sets for a given time increment. This is done by solving the node equations by Gauss elimination of an individual column of the grid, while all terms related to the nodes in adjacent columns are held constant. The set of column equations is then implicit in the direction along the column and explicit in the direction orthogonal to the column alignment (Peaceman and Rachford, 1955). A solution to the set of column equations is then determined by the methodology explained in the following sections.

After all column equations have been processed column by column, the node equations are solved again by Gauss elimination of an individual row while all terms related to adjacent rows are held constant. When all equations have been solved row by row, an 'iteration' has been completed and an initial head matrix produced. The above process is repeated for a number of iterations. Subsequent to each iteration, the head matrix is compared to the head matrix of the previous iteration. When the difference between the two matrices is within an acceptable tolerance of error, convergence has been achieved and this completes the calculations for the given time step or increment. The successfully converged head values are then utilized as the initial conditions of the next time step and this process repeats until the head values have been calculated for every time step of the simulated pumping period. The next section presents a more detailed summary of the MIADIM solution operations (Prickett and Lonnquist, 1971).

4.1.1 MIADIM Solution Operations

Equation 4.2 is modified and rearranged to facilitate node equation solving by column and rows:

$$AA_j * h_{i,j-1} + BB_j * h_{i,j} + CC_j * h_{i,j+1} = DD_j \qquad (4.3)$$

$$AA_i * h_{i-1,j} + BB_i * h_{i,j} + CC_i * h_{i+1,j} = DD_i \qquad (4.4)$$

Equations 4.3 and 4.4 are used to solve calculations by columns and rows, respectively. In the case of Equation 4.3, the constant terms AA, BB, CC and DD are:

$$AA_j = -T_{i,j-1,1} \qquad (4.3a)$$

$$BB_j = T_{i-1,j,2} + T_{i,j,2} + T_{i,j,1} + T_{i,j-1,1} + S_{i,j}\,\Delta x \Delta y/\Delta t \qquad (4.3b)$$

$$CC_j = -T_{i,j,1} \qquad (4.3c)$$

$$DD_j = (S_{i,j}\,\Delta x \Delta y/\Delta t) * ho_{i,j} - Q_{i,j} + T_{i-1,j,2}\, h_{i-1,j} + T_{i,j,2} h_{i+1,j} \qquad (4.3d)$$

In the case of Equation 4.4, the constant terms AA, BB, CC and DD are:

$$AA_i = -T_{i-1,j,2} \qquad (4.4a)$$

$$BB_i = T_{i-1,j,2} + T_{i,j,2} + T_{i,j,1} + T_{i,j-1,1} + S_{i,j}\,\Delta x \Delta y/\Delta t \qquad (4.4b)$$

$$CC_i = -T_{i,j,2} \qquad (4.4c)$$

$$DD_i = (S_{i,j}\,\Delta x \Delta y/\Delta t) * ho_{i,j} - Q_{i,j} + T_{i,j-1,1}\, h_{i,j-1} + T_{i,j,1} h_{i,j+1} \qquad (4.4d)$$

When isolating a column or row for solution (Figure 4.1), there are three unknowns, ho $_{i,j}$, $h_{i-1,j}$, and $h_{i+1,j}$ for each node along a column (Equation 4.3), or three unknowns, ho $_{i,j}$, $h_{i,j-1}$, and $h_{i,j+1}$ for each node in a row (Equation 4.4). The solution of a set of column or row head equations is accomplished by Gauss elimination incorporating the Thomas algorithm (Prickett, 1975). The solution technique utilizes the B and G arrays of Peaceman and Rachford(1955). The method for calculating heads along rows or columns involves first computing values of G_i (see Equation 4.5) and B_i (see Equation 4.6) or G_j (see Equation 4.8) and B_j (see Equation 4.9) for the nodes of a row or column in order of increasing i or j, respectively. After completing this, the head at the last node of a row or column is found. All other heads in the row or column are then computed in order of decreasing i or j, respectively. After completing the calculation of heads in an individual row or column, the computer proceeds to the next row or column until all have been processed satisfactorily (Walton, 1985). To complete the row calculations (Prickett, 1975):

$$G_i = (DD_i - AA_i G_{i-1})/(BB_i - AA_i B_{i-1}) \tag{4.5}$$

$$B_i = CC_i/(BB_i - AA_i B_{i-1}) \tag{4.6}$$

where: AA_i (Equation 4.4a) is set equal to zero for the first node of a row and CC_i (Equation 4.4c) is set equal to zero for the last node of a row and (Prickett, 1975):

$$h_{i,j} = G_i - B_i h_{i+1,j} \tag{4.7}$$

To complete the column calculations (Prickett, 1975):

$$G_j = (DD_j - AA_j G_{j-1})/(BB_j - AA_j B_{j-1}) \tag{4.8}$$

$$B_j = CC_j/(BB_j - AA_j B_{j-1}) \tag{4.9}$$

where: AA_j (Equation 4.3a) is set equal to zero for the first node of a column and CC_j (Equation 4.3c) is set equal to zero for the last node of a column and (Prickett, 1975):

$$h_{i,j} = G_j - B_j h_{i,j+1} \tag{4.10}$$

Iterations are terminated and the values of head converge when the sum or absolute value of the difference in head from the present iteration compared to the last iteration has not changed more than the tolerance of error.

The rate of convergence is increased by inclusion with each time increment of a preliminary head predictor (Walton, 1985). The preliminary head predictor assumes that the ratio of the present to past differences in head will equal the ratio of the future to present differeııces in head, or (Prickett and Lonnquist, 1971):

$$F = D/DL_{i,j} = \text{predicted future difference in head}/D \tag{4.11}$$

where:

D = present difference in head

$DL_{i,j}$ = past difference in head

When F is undefined because $DL_{i,j}$ is zero, as in the first two time steps, F is set equal to 1.0.

4.1.2 Boundary Conditions

The two types of boundaries incorporated into this computer model are constant-flux and constant-head boundaries. They can be used to simulate real boundaries in the aquifer system or to represent artificial boundaries.

The constant-flux boundary can be used to represent well withdrawals, well injections, no-flow boundaries, or drainage ditches. Well withdrawals or injections have been historically designated by stipulating a positive flux or negative flux rate respectively, for the appropriate node. No-flow boundaries simulate natural limits or barriers to groundwater flow. The no-flow boundaries are designated by setting the transmissivity equal to zero at the appropriate nodes, and thereby restricting the flow of water across the boundaries of the cell containing that node (Konikow and Bredehoeft, 1978). The borders of the finite difference grid are represented as no-flow boundaries in this model.

A constant-head boundary can be used to simulate parts of the aquifer where the head will not change with time, such as recharge galleries or in areas beyond the influence of hydraulic stresses. The constant-head boundaries are designated by setting the infiltration term equal to a sufficiently high value (1.0E+21 (Konikow and Bredehoeft, 1978)) to allow the head in the aquifer at a node to be implicitly computed as a value that is equal to the value of the desired constant-head elevation (Prickett and Lonnquist, 1971). The resulting rate of leakage into, or out of, the designated constant-head cell would equal the flux required to maintain the head in the aquifer at the specified constant-head elevation (Konikow and Bredehoeft, 1978).

4.1.3 Non-Uniform Time Increments

Subsequent to the application of a hydraulic stress to an aquifer system, water levels will change rapidly at first (when a well first starts to pump) and as time goes on, the rate of change declines. During the initial stage of rapidly fluctuating water levels, small time increments should be used to ensure accurate head calculations. As water levels stabilize , progressively larger time increments should be used to optimize calculation time without loss of accuracy.

This computer model applies a geometric progression to the time increment. An initial small time increment is chosen by the user. Successive time increments are progressed by the relationship:

$$\text{Time} = \text{Delta} * \Omega \tag{4.12}$$

where:

$$\Omega = 1 + 1.2^1 + 1.2^2 + 1.2^3 + \ldots\ldots\ldots\ldots\ldots\ldots + 1.2^{(\text{istep-1})} \tag{4.13}$$

Delta = initial time increment

istep = time increment number.

The factor 1.2 has been found to produce satisfactory results for most applications (Prickett and Lonnquist, 1971) but may need to be adjusted to fit individual aquifer conditions.

4.1.4 Leaky Artesian Conditions

Leaky artesian conditions are simulated by assuming the rate of flow through an aquitard into an aquifer is directly proportional to the horizontal projected aquitard area, the vertical permeability of the aquitard, and the head difference between that in the aquifer and the source bed above the aquifer, and inversely proportional to the aquitard thickness (Walton, 1985). Assuming there is no water release from storage within the aquitard, a recharge factor is assigned to each node where the aquitard exists with the following equation (Prickett and Lonnquist, 1971):

$$R_{a\,i,j} = (P_{i,j} / m_{i,j})\, A_{a\,i,j} \tag{4.14}$$

where:

$R_{a\,i,j}$ = aquitard leakage (recharge) factor

$P_{i,j}$ = aquitard vertical permeability

$m_{i,j}$ = aquitard thickness

$A_{a\ i,j}$ = nodal area through which aquitard leakage occurs

Figure 4.2 depicts a parameter schematic for the leaky artesian scenario. When aquifer heads are above the aquitard base, the leakage rate is given by the following equation (Prickett and Lonnquist, 1971):

$$Q_{L\ i,j} = (RH_{i,j} - h_{i,j})\, R_{A\ i,j} \tag{4.15}$$

where:

$Q_{L\ i,j}$ = leakage (recharge) rate

$RH_{i,j}$ = head elevation in source bed

$h_{i,j}$ = head elevation in aquifer

The vector volumes related to recharge extend the full aquitard thickness and have an area that encompasses half the grid spacing from the node of interest to the for surrounding nodes.

4.1.5 Induced Infiltration

Induced infiltration from a stream into an aquifer system is simulated by assuming that the effects of partial penetration of the stream are negligible; the head in the stream remains constant; and the rate of flow through a streambed is directly proportional to the streambed area, streambed permeability, and the aquifer-stream head difference, while being inversely proportional to the streambed thickness (Walton, 1985). Figure 4.3 presents a schematic of the induced infiltration scenario. An aquifer infiltration recharge factor is defined and assigned to each streambed node with the following equation (Prickett and Lonnquist, 1971):

$$R_{s\ i,j} = (P_{i,j} / m_{s\ i,j})\, A_{s\ i,j} \tag{4.16}$$

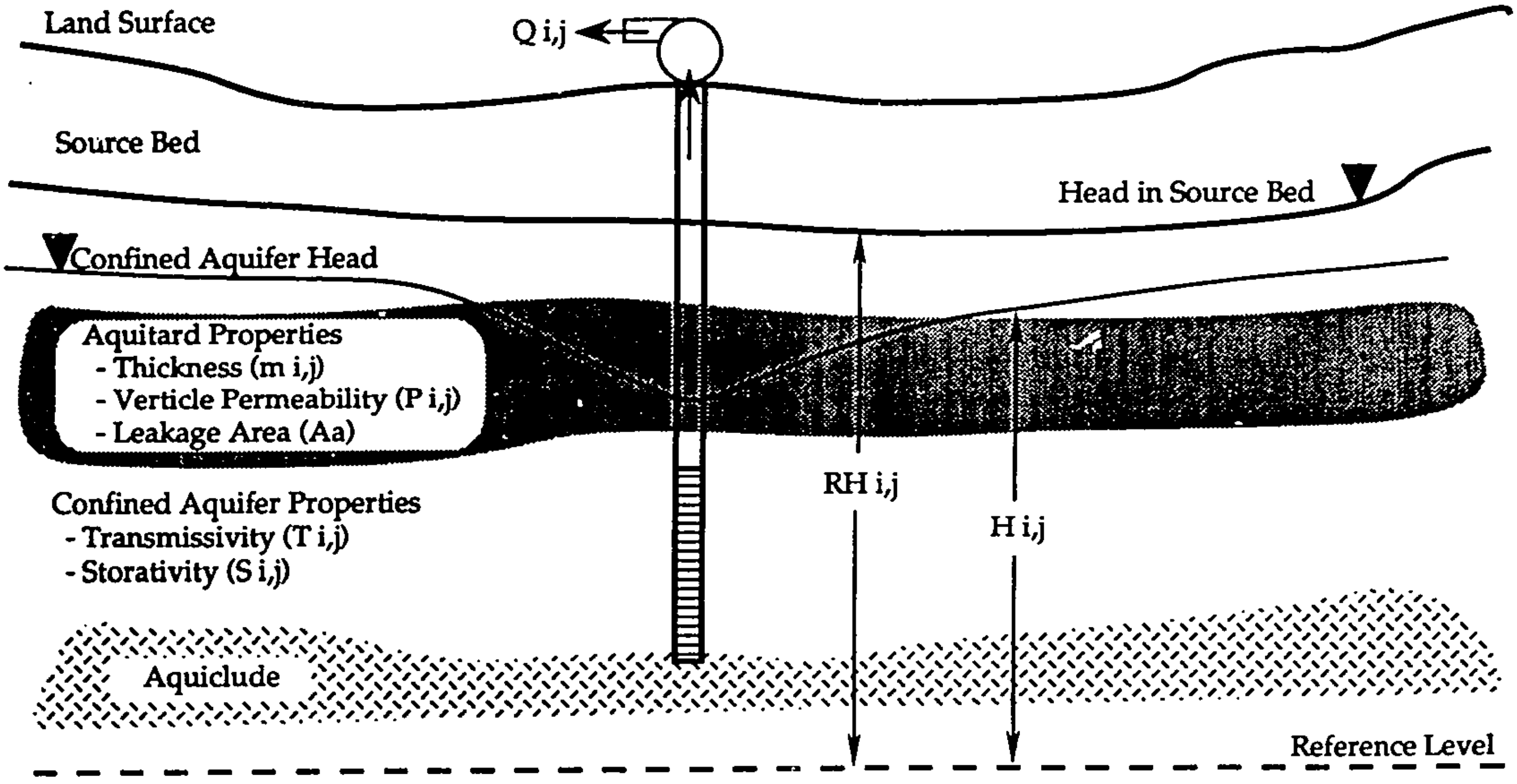


Figure 4.2

Parameters for Leaky Artesian Conditions

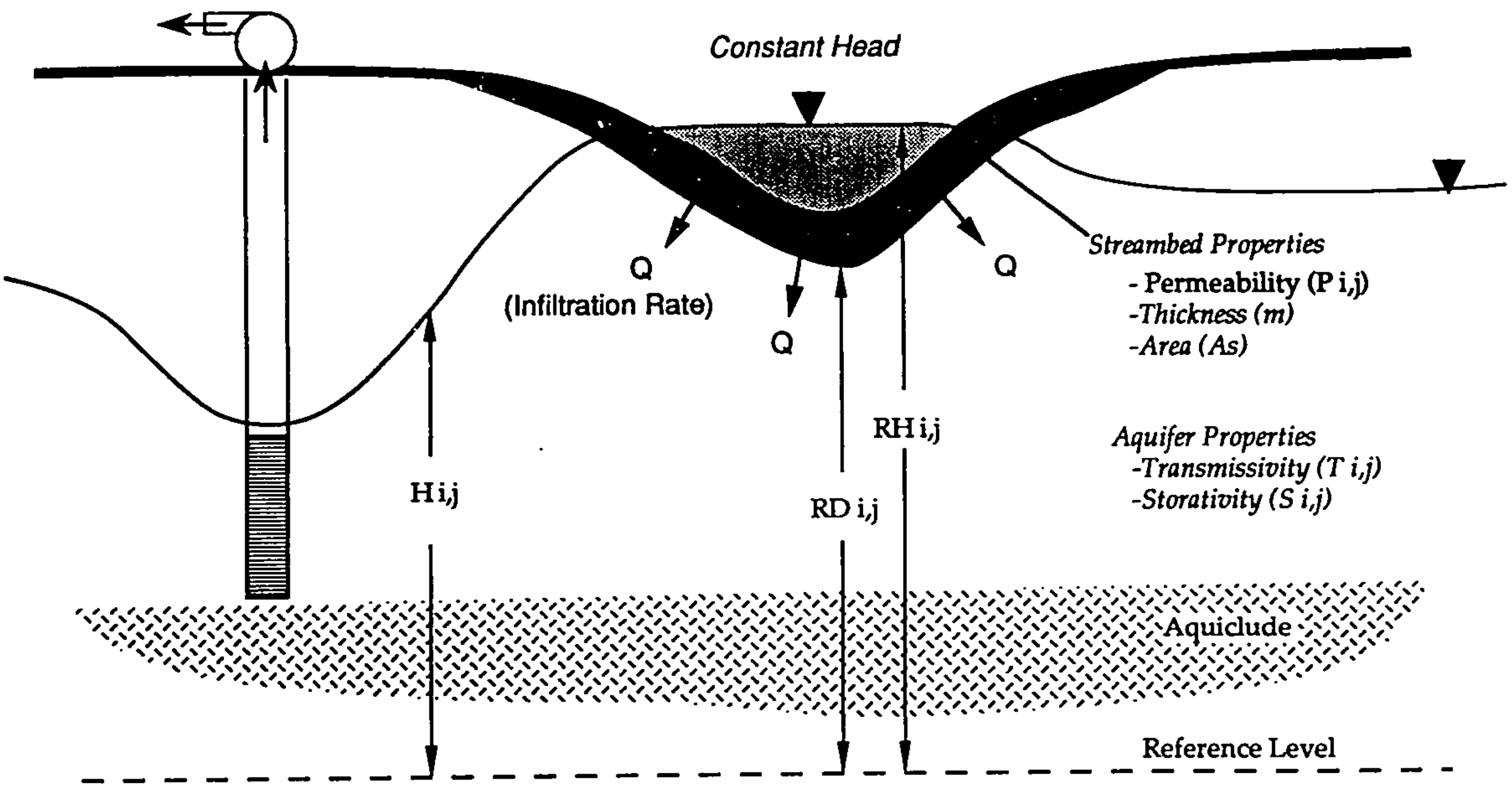


Figure 4.3

Parameters for Induced Infiltration

where:

$R_{s\ i,j}$ = induced infiltration (recharge) factor

$P_{i,j}$ = streambed vertical permeability

$m_{i,j}$ = streambed thickness

$A_{a\ i,j}$ = nodal area of streambed through which infiltration occurs.

Figure 4.4 presents how the streambed nodal areas are derived. When the aquifer heads are above the streambed base, the induced infiltration rate is given by the following equation (Prickett and Lonnquist, 1971):

$$Q_{I\ i,j} = (RH_{i,j} - h_{i,j})\, R_{S\ i,j} \tag{4.17}$$

where:

$Q_{I\ i,j}$ = induced infiltration (recharge) rate

$RH_{i,j}$ = stream surface elevation

$h_{i,j}$ = head elevation in aquifer

When the aquifer heads are below the streambed base, the induced infiltration rate is given by the following equation (Prickett and Lonnquist, 1971):

$$Q_{I\ i,j} = (RH_{i,j} - RD_{i,j})\, R_{S\ i,j} \tag{4.18}$$

where:

$RD_{i,j}$ = streambed base elevation

4.1.6 Water Table Conditions

In a water table aquifer, the amount of dewatering taking place as a result of pumping is small compared to the total aquifer saturated thickness (Walton, 1985). Water table conditions are simulated by adjusting the values of the aquifer's storativity and transmissivity in Equation 4.2 to accommodate

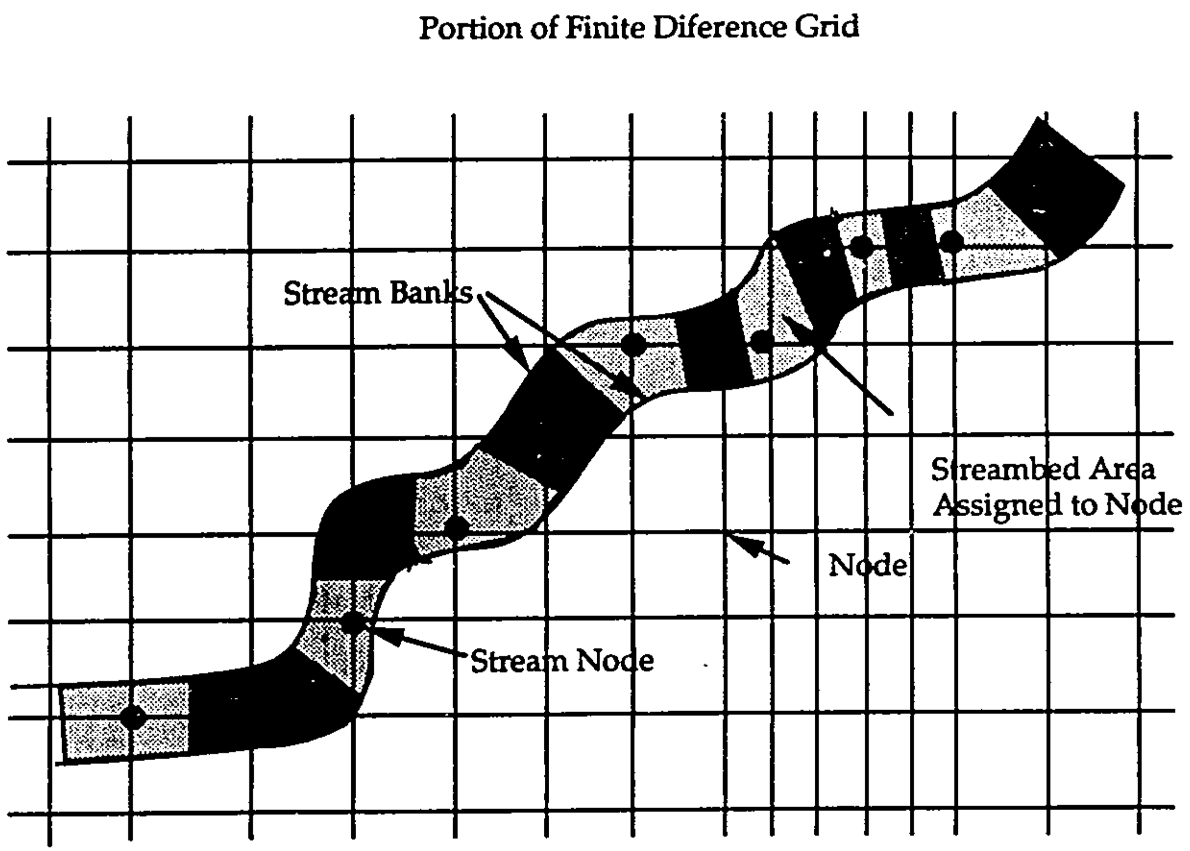


Figure 4.4

Streambed Infiltration Vector Areas

the above noted effects. The following equations are used when storativity changes because of heavy withdrawals (Prickett and Lonnquist, 1971):

when $h_{i,j}$ is above or equal to $CH_{i,j}$ (Figure 4.5):

$$S_{i,j} = SF1_{i,j} \qquad (4.19)$$

when $h_{i,j}$ is below $CH_{i,j}$:

$$S_{i,j} = SF2_{i,j} \qquad (4.20)$$

where as noted in Figure 4.5:

$CH_{i,j}$ = elevation of aquifer top

$SF1_{i,j}$ = artesian storativity factor

$SF2_{i,j}$ = water table storativity factor

$S_{i,j}$ = storativity facιᴗr used in computations

Transmissivity is the product of an aquifer's hydraulic conductivity and its saturated thickness. Under water table conditions, the gravity drainage of interstices decreases the aquifer's saturated thickness and therefore the transmissivity must be adjusted accordingly. Assuming slow gravity drainage effects are negligible, simulation of a water tᵣᵇle aquifer's transmissivity is accomplished by applying the following equations (Prickett and Lonnquist, 1971):

$$T_{i,j,1} = PERM_{i,j,1} \left[(h_{i,j} - BOT_{i,j})(h_{i,j+1} - BOT_{i,j+1}) \right]^{1/2} \qquad (4.21)$$

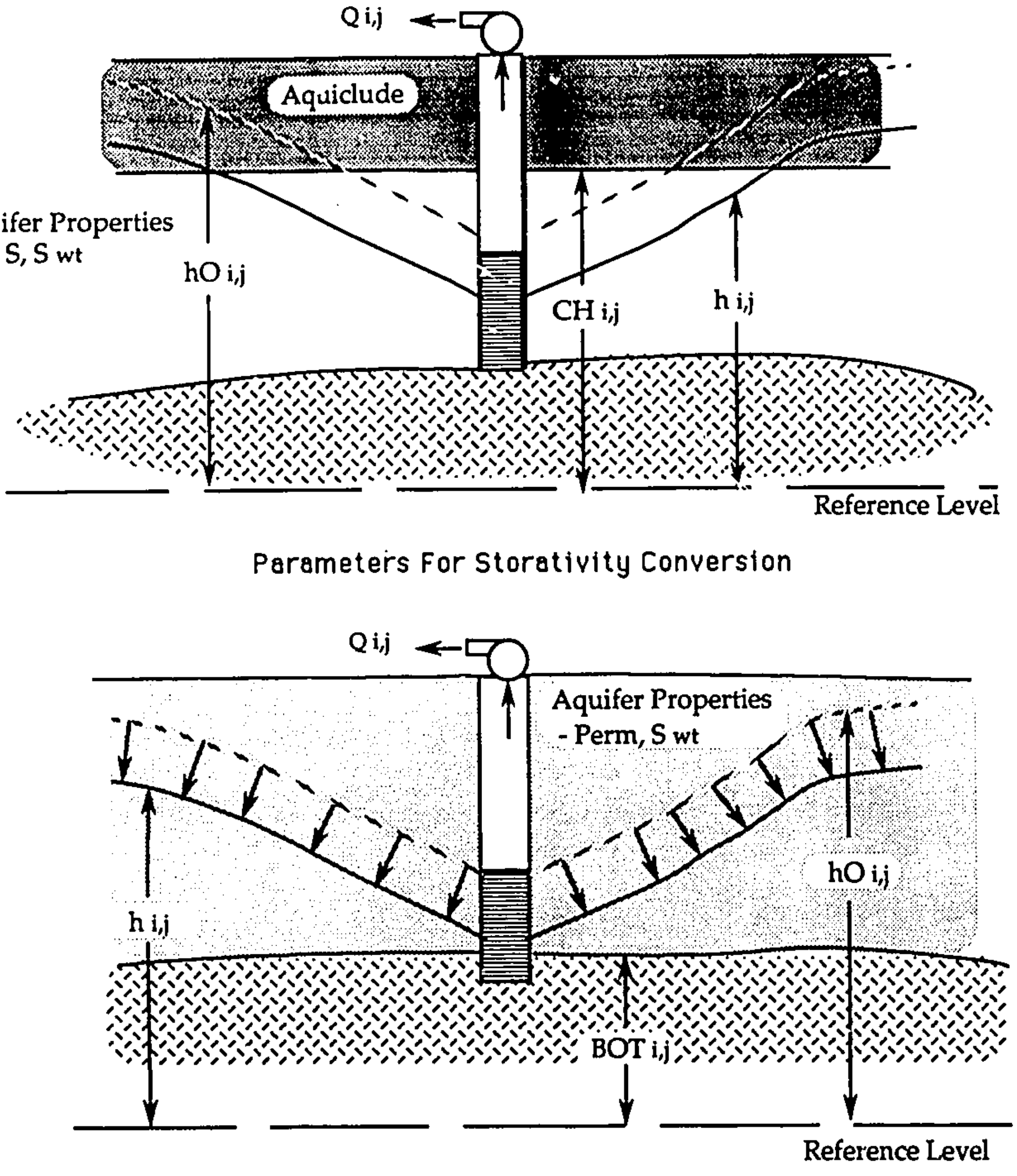
$$T_{i,j,2} = PERM_{i,j,2} \left[(h_{i,j} - BOT_{i,j})(h_{i+1,j} - BOT_{i+1,j}) \right]^{1/2} \qquad (4.22)$$

where as noted in Figure 4.5:

$T_{i,j,1}$ = equivalent geometric mean aquifer transmissivity with dewatering between i,j and i,j+1

$T_{i,j,2}$ = equivalent geometric mean aquifer transmissivity with dewatering between i,j and i+1,j

$PERM_{i,j,1}$ = aquifer permeability between i,j and i, j+1



Parameters For Storativity Conversion

Parameters For Transmissivity Conversion

Figure 4.5
Parameters For Water Table Conversions

$PERM_{i,j,2}$ = aquifer permeability between i,j and i+1, j

$BOT_{i,j}$ = aquifer base elevation at node i,j

$BOT_{i,j+1}$ = aquifer base elevation at node i,j+1

$BOT_{i+1,j}$ = aquifer base elevation at node i+1,j

When heads decline below the aquifer base (Walton, 1985), they are set equal to the aquifer base elevation plus 0.01 (ft or m) .

4.2 Theory of Well Capture Zone Determination

The velocity of flow through an aquifer can be represented by rearranging and modifying Darcy's law (Freeze and Cherry, 1979):

$$Q = KIA \tag{4.23}$$

where:

Q : is the volumetric flow rate in cubic metres per day;

K: is the hydraulic conductivity in metres/day;

I: is the hydraulic gradient (dimensionless);

A: is the cross-sectional area through which flow occurs in square meters.

By rearrangement, the 'Darcy velocity' (V_d) can be obtained by (Freeze and Cherry, 1979):

$$V_d = Q/A = KI \tag{4.24}$$

Since the flow occurs only through the pores, rather than through the entire cross-sectional area (A), the true pore velocity or interstitial velocity (V_i) can be obtained by (Freeze and Cherry, 1979):

$$V_i = Q/A\varnothing_e = KI/\varnothing_e \tag{4.25}$$

where:

$Ø_e$: is the aquifer effective porosity.

The centre term of the above equation ($Q/AØ_e$) can be used to compute the velocity of waters moving towards a pumping well because Q is usually known for the well and A is readily estimated (Keely and Tsang, 1983). Assuming uniformly radial flow towards the well, the cross-sectional area through which flow must pass to reach the well is equal to the area of the curved face of an imaginary cylinder of radius r, where r is chosen as the distance from the well where the velocity effect is of interest to the investigator. The area of the curved face of the imaginary cylinder at the radial distance is (Keely and Tsang, 1983):

$$A = 2\Pi\, rh \qquad (4.26)$$

where:

h: is the effective saturated thickness of the aquifer zone yielding water to the well.

Naturally, this implies that there is a distribution of velocities surrounding the pumping well which increase in magnitude as one gets closer to the well. By substitution of Equation 4.26 into Equation 4.25, an expression for velocity towards the pumping well may be obtained:

$$V_{pumping} = Q/(2\Pi rhØ_e) \qquad (4.27)$$

The right hand term of Equation 4.25 ($V_i = KI/Ø_e$) is generally employed for estimation of the aquifer's natural flow velocity. This is because the average hydraulic conductivity (K) and hydraulic gradient (I) are usually known or easily estimated, whereas the aquifer's bulk flow (Q) and cross-sectional area (A) are not easily estimated.

Manual plots of the velocity distribution surrounding a pumping well, in the presence of a real natural flow rate and direction, can be constructed from Equations 4.27 and the right hand term of Equation 4.25 (Keely and

Tsang, 1983). The natural flow system moves water toward the well on the well's upgradient side, but tries to move water away from the well on the downgradient side. Therefore, at distances upgradient from a pumping well, the velocities from natural flow and pumpage are added together to yield net velocities, whereas, at distances downgradient from the pumping well, the velocities from natural flow and pumpage are subtracted to yield net velocities (Keely and Tsang, 1983).

At some distance downgradient, the pull of waters back toward the well by pumping is exactly countered by the flow away from the well due to the natural flow velocity. This location is called the stagnation point (Todd, 1980) or velocity divide (Keely and Tsang, 1983). It is important to note that the stagnation point occurs well within the cone of depression caused by pumping. This relationship is such that the greater the pumping stress, the farther downgradient the stagnation point occurs, or conversely, the greater the natural flow velocity, the closer the stagnation point will be to the pumping well (Keely and Tsang, 1983).

The distance to the stagnation point downgradient of an extraction well can be calculated directly by setting the expression for Vpumping (Equation 4.27) equal to the value of Vnatural (the right hand term of Equation 4.25) and rearranging to solve directly for r:

$$r = Q/(2\prod h\varnothing_e V_{natural}) \qquad (4.28)$$

where:

r : stagnation distance

which can be rewritten as:

$$r = Q/(2\prod TI) \qquad (4.29)$$

where:

Q : well pumping rate;

T : aquifer's transmissivity;

I : aquifer's average gradient before pumping began.

The width of the capture area parallel to the extraction well is then estimated to be (Todd, 1980):

$Wc = \Pi r$

where:

Wc : the wells capture width;

r : is the distance to the stagnation point (Equation 4.29).

An important point that must be noted is that only in extremely rare cases of zero natural flow velocity are the areal boundaries of a wells capture zone and the cone of depression identical everywhere (Keely and Tsang, 1983). In the past, the standard method for developing capture zones was to overlap the cones of depression of a line of pumping wells. Therefore, in an aquifer with a large natural flow velocity, the use of the well's cone of depression as the well's capture zone could lead to potential failure in containing the plume, despite the fact that their adjacent cones of depression overlap (Keely and Tsang, 1983).

5.0 PROGRAM DESCRIPTION

The computer software package developed in this thesis has been designed to provide the user with optimum solutions to dewater complex excavations or to control contaminant plume migration by controlling groundwater elevations. The program allows considerable flexibility in data input so that a wide range of field situations may be addressed. Most groundwater flow models presently in use, calculate the potentiometric head or velocity flow paths of an aquifer system as a result of the placement of pumping wells within a hydrogeologically variable environment. The model presented in this thesis goes one step further by having a series of wells, with various pumping rates, adv :nce through designated areas of the finite difference grid system discretizing the aquifer. As the wells advance to each new nodal configuration and the spatial orientation of the wells within the hydrogeologically variable flow field change, the optimum well configuration solution is determined when user specified conditions are fulfilled . In a dewatering simulation, these conditions are in the form of predesignated control point potentiometric surface elevations that must be matched by the program calculated elevations. In a plume management scenario, they are fulfilled when a given well configuration maximizes contaminant plume coverage within the well's capture zones. In addition to utilizing pumping and recharge wells to manipulate groundwater elevations, the model is able to simulate recharging ditches, collection ditches, induced infiltration from surface streams and ponds, as well as areas of no flow due to complex geology or foundation structures. Appendix A contains a copy of the program listing.

The computer model has been written in Turbo Pascal Version 5.0, and is able to run an IBM-XT, AT, and 386 personal computer or equivalent clone.

It is recommended that the computer system have 2 megabyte's of RAM memory and a math co-processor.

The solution to an application of this program has three phases prior to simulation initiation:

i) data collection;

ii) data input, and;

iii) program sensitization.

These phases are discussed in greater detail in the following sections.

5.1 Data Collection

To solve a dewatering problem, the user must either have basic knowledge of the aquifer's hydraulic properties as determined from a well pump test or the user can estimate the properties if familiar with the site's hydrogeology. Areal distributions of the aquifer's potentiometric surface, transmissivity, storativity, the locations of key controlling water table elevations such as a nearby stream or a neighbor's well that should not be dewatered, and the desired dewatered potentiometric elevations, must be known by the user before initiating any simulations with this model.

To capture a contaminant plume, the user must have the same basic information of the aquifer's hydraulics as required in the dewatering scenario, plus the areal distribution of the indicator contaminant in the aquifer. It is safe to assume that the user is familiar with the indicator contaminants distribution within the aquifer if they are implementing a study to design a well capture and waste water treatment system.

5.2 Data Input

The program is driven by a menu system that allows the user to quickly tour through the model and make data changes efficiently. In addition to the menu prompts for data input, a spreadsheet has been developed to allow more efficient manipulations of the many aquifer characterizing matrices. The spread sheet may also read ASCII formatted files derived from digitized maps and incorporate the information into the data set for a particular simulation.

5.2.1 Menu System

The menu system consists of a series of interactive windows. The windows have a menu title followed either by a series of submenu titles or interactive parameter statements. At the base of the window is a dialogue box directing the user through the program's menu system. Figure 5.1 presents a program menu map.

5.2.2 Menu System Mobility

The user may advance through the window system by pressing "return" and retreat to the previous menu system by pressing "escape". The interactive parameter windows require the user to enter the appropriate data. Within a window, the various parameters are selected by highlighting a parameter using the "up" and "down" arrow keys. Data are entered manually into the space highlighted. The "left" and "right" arrow keys allow the user to move horizontally within the highlighted parameter space. The "delete"

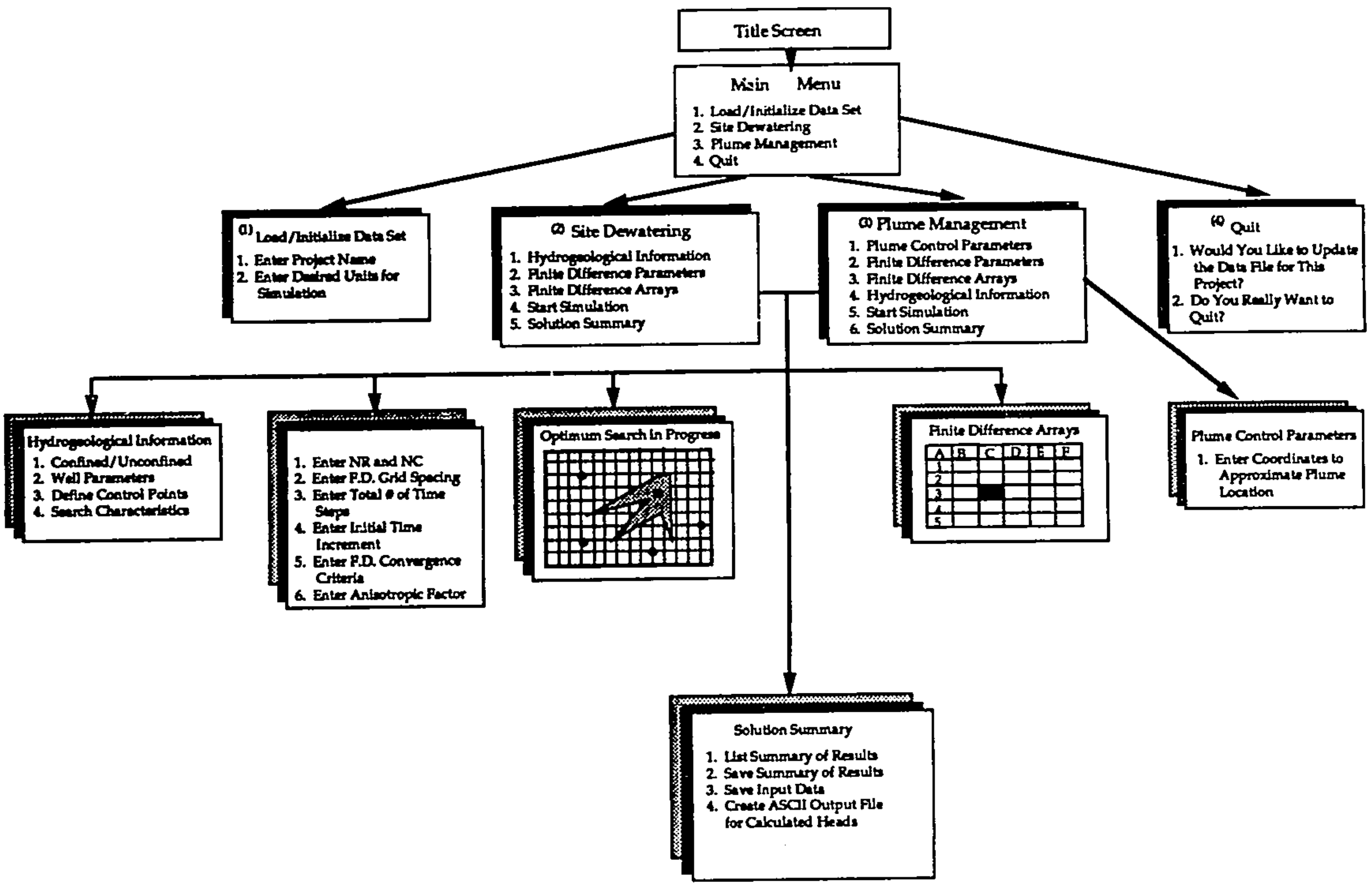

Title Screen
Main Menu
1. Load/Initialize Data Set
2. Site Dewatering
3. Plume Management
4. Quit
(1) Load/Initialize Data Set
1. Enter Project Name
2. Enter Desired Units for Simulation
(2) Site Dewatering
1. Hydrogeological Information
2. Finite Difference Parameters
3. Finite Difference Arrays
4. Start Simulation
5. Solution Summary
(3) Plume Management
1. Plume Control Parameters
2. Finite Difference Parameters
3. Finite Difference Arrays
4. Hydrogeological Information
5. Start Simulation
6. Solution Summary
(4) Quit
1. Would You Like to Update the Data File for This Project?
2. Do You Really Want to Quit?
Hydrogeological Information
1. Confined/Unconfined
2. Well Parameters
3. Define Control Points
4. Search Characteristics
1. Enter NR and NC
2. Enter F.D. Grid Spacing
3. Enter Total # of Time Steps
4. Enter Initial Time Increment
5. Enter F.D. Convergence Criteria
6. Enter Anisotropic Factor
Optimum Search in Progress
Finite Difference Arrays
A
B
C
D
E
F
1
2
3
4
5
Plume Control Parameters
1. Enter Coordinates to Approximate Plume Location
Solution Summary
1. List Summary of Results
2. Save Summary of Results
3. Save Input Data
4. Create ASCII Output File for Calculated Heads

Figure 5.1

Program Menu Map

or "backspace" key allows data to be erased left of the cursor position within the highlighted parameter space.

5.2.3 Main Menu

Upon invoking the executable program command, the program title screen appears outlining the program's name, author, and copyright statement as well as directions of how to continue. Proceeding to the **Main Menu**, there are four sub-menus listed: **Load/Initialize Data Set, Site Dewatering, Plume Management**, and **Quit**.

5.2.4 Load Initialize Data Set

The first selection must always be the **Load/Initialize Data Set** submenu. This menu asks the user to input the job name or number. If the job is new, all parameters are initialized, otherwise the most updated data set for the identified job is loaded into memory. The menu then requests that you enter the units that will be used throughout the simulations. When this information has been entered, the dialogue box directs the user back to the main menu.

5.2.5 Site Dewatering

To utilize the program to produce optimum dewatering scenarios, the user would select the second listed sub-menu of the **Main Menu** entitled **Site Dewatering**. As depicted in Figure 5.1, the **Site Dewatering** window has five sub-menus. Selection of a sub-menu would lead to a series of interactive screens that will allow data to be input. The five sub-menus are **Hydrogeological Information, Finite Difference Parameters, Finite Difference Arrays, Start Simulation, and Solution Summary.**

5.2.5.1 Hydrogeological Information

If the sub-menu **Hydrogeological Information** is selected, the following information is requested in a series of interactive parameter windows: designation of the aquifer as confined or unconfined, number of site dewatering control points, number of pumping or recharge wells, location of the site dewatering control points, control point groundwater elevations, acceptable control point elevation ranges, maximum allowable well drawdown, and the finite difference grid coordinates of the well search areas.

5.2.5.2 Finite Difference Parameters

If the sub-menu **Finite Difference Parameters** is selected, the following finite difference approximation parameters are requested in a series of interactive windows: number of columns and rows in the finite difference grid, default value of spacing between gridlines, total number of simulation time steps, initial time increment, error of tolerance for finite difference convergence, and finally an anisotropic factor may be stipulated if warranted.

5.2.5.3 Data Manipulation by Spread Sheet

If **Finite Difference Arrays** is selected, a spreadsheet is opened. The various aquifer characterizing arrays such as initial head, transmissivity, storativity, infiltration, recharge, and aquifer thickness may be edited efficiently within the spreadsheet. The areal distribution of recharge and collection ditches may be simulated here. As well, the discretized nodal values for river bed properties and confining layer properties are addressed.

5.2.5.4 Dewatering Simulation

If the sub-menu **Start Simulation** is selected, a graphic display appears and presents the finite difference grid which the user watches as the search for an optimum well configuration to dewater the site progresses. The recharge wells and pumping wells move through their search areas incrementally, so that all possible well configurations have been addressed, to find the successful configuration that will satisfy the control point criteria. If a successful configuration is found, a window opens on the screen summarizing the successful configuration's specifications as well as writing the successful solution to an output file. The user may continue to locate additional successful configurations or can quit the simulation and return to the previous menu.

If during the simulation, the finite difference approximations do not successfully converge, an error message appears upon the screen requesting the user to try different values of the time increment or error convergence criteria.

5.2.5.5 Solution Summary

Subsequent to the search simulation, the **Site Dewatering** menu appears. At this time you may request the sub-menu **Solution Summary**.

Within the **Solution Summary** sub-menu, the user is able to see a tabulation of all successful solutions and their relevant information that has been saved in an output file and has the ability to send the output file to a printer.

5.2.6 Plume Management

If the user had selected **Plume Management** instead of **Site Dewatering**, the series of subsequent sub-menus would have been identical to those described above for **Site Dewatering** . In addition, the user would be requested to locate the plume within the finite difference grid system and to characterize the contaminant plume's areal concentration distribution within the aquifer with the spreadsheet.

5.2.6.1 Plume Control Parameters

In order to accommodate complex plume shapes within a rectangular grid system, the user is asked to enter the number of rectangular shapes that could be combined to represent the plume shape in the field. A series of interactive windows allows the grid coordinates of these shapes to be input.

5.2.6.2 Plume Characterizing Array

A plume characterizing array must be created by the user. The simulation will determine the well field configuration whose capture zones optimally cover the contaminant plume, represented by the plume characterizing array. The user discretizes the areal distribution of the plume characterizing parameters concentration, preferably utilizing the most mobile contaminant of concern. This information is input as an array in the program spreadsheet.

5.2.6.3 Plume Capture Simulation

The plume management portion of this program moves wells through prescribed grid bounded search areas to determine the best well configurations to capture the plume. At each node point within the grid representing a capture well for a particular well configuration, the capture zone is drawn on the screen. When the best solution well configuration is determined, the model will next calculate the aquifer's potentiometric surface as a result of the applied capture stresses.

5.2.7 Data Preparation Time

It is anticipated that data preparation input time for both dewatering and plume capture applications will be less than one hour, once the user has become familiar with the model. The interactive parameter menu design and the spreadsheet facilitate quick data file manipulations.

5.3 Program Sensitization

Before the model can be utilized to design a plume management or site dewatering scheme, a sensitivity analysis should be performed to match the aquifer's simulated flow field with the site's observed flow field. Differences may exist between the simulated and observed flow fields due to the heterogeneous nature of the actual site. This can be compensated for by varying the nodal parameter input values within reasonable limits. Only after a sensitivity analysis is performed, can the user feel confident that the model will produce reasonably accurate results when subsequently applied to a dewatering or plume management problem.

6.0 CASE STUDY APPLICATIONS

To validate the computer program's solutions and to demonstrate that the model is an effective planning tool, for solving complex dewatering and plume capture problems, the program has been applied to three independent case studies.

The first case study is a simple comparison of the finite difference output of the program presented in this thesis and the output of a well documented and successfully proven flow model produced by Thomas A. Prickett & Associates known as PLASMER4 (Pricket, 1975). The PLASMER4 model determines the potentiometric surface of the water table subsequent to applying a stress field. The purpose of this comparison is to ensure that the results of the finite difference calculations, used to determine the potentiometric surface of the watertable in the program, are reasonably accurate. The solution of the finite difference approximation of the flow equation constitutes the framework to solving the dewatering and plume capture algorithms utilized in the computer program presented in this thesis.

The second case study is a comparison between a documented hydrogeological consultant's design to depressurize a bedrock aquifer for the Welland Canal realignment project that commenced in southwestern Ontario in 1969 and a dewatering design determined by the thesis program. The dewatering solution produced by the program is compared to the solution derived by the consulting groundwater specialists to assess the usefulness of the program as a time saving, dewatering planning tool.

The final case study presents a United States Environmental Protection Agency (US EPA) documented field investigation that utilized pumping wells to capture and extract contaminated groundwater from an industrial property in Florida. The plume capture solution produced by the thesis program is compared to the solution derived by the consulting engineers responsible for the site's remediation. The purpose of the comparison is to assess the usefulness of the program as a time saving, planning tool for the pre-design of contaminant plume capture well systems.

The aquifer parameter values used in the case study comparisons are all expressed in United States (US) and Imperial units for the following reasons:

i) The flow model PLASMER 4 was designed in the United States which necessitated the use of US units to comply with constants built into the program.

ii) Both the dewatering and plume control case studies were documented in Imperial and US units and during the thesis simulations, the units were kept in these units to maintain consistency with documentation and for ease of comparison.

It should be noted, however, that the model developed for this thesis performs equally well using parameters expressed in either metric, imperial, or U.S. units.

6.1 Simple Case to Assess Finite Difference Algorithm Accuracy

As noted previously, the finite difference approximation solution of the flow equation (Equation 4.1) constitutes the framework for solving a dewatering or plume capture problem in this program. To assess the accuracy of the finite difference approximations, the program presented in this thesis and the successfully proven program PLASMER4 (Prickett, 1975) were applied to a simple field scenario and, subsequently, their results compared. The thesis program was utilized in this comparison as a simple flow model by having the designated well search area limited to one node of the finite difference grid and subsequently altering the program source code to print out the calculated potentiometric heads of the aquifer for every time step of the simulation.

The simple field scenario is described as follows:

- A confined aquifer is discretized by a finite difference grid (Figure 4.1) consisting of ten rows and ten columns. The grid spacing is 100 feet and is constant over the aquifer.

- The aquifer hydrogeologic properties which are constant throughout the aquifer are :

Transmissivity	= 500 gal/day/ft
Storativity	= 0.05
Initial Potentiometric Elevation	= 100 ft

- There is one pumping well located at the exact centre of the aquifer study area at node (5,5). The well's pumping rate is 2000 gal/day.

Both computer programs were run for a time period simulating approximately 270 days of pumping. Appendix B contains the program outputs at various simulated time intervals. Tables 6.1, 6.2, and 6.3 present a

Table 6.1

**Comparison of Potentiometric Surfaces
Calculated by PLASMER4 and Thesis Model**

PLASMER4 head calculations for a simulated pumping Period of 10.3 days*
Thesis Head calculations for a simulated pumping Period of 9.9 days*

Rows					*Columns*					
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0**	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0.01	0	0	0	0	0
4	0	0	0	0.01	0.02	0.01	0	0	0	0
5	0	0	0.01	0.02	0.03	0.02	0.01	0	0	0
6	0	0	0	0.01	0.02	0.01	0	0	0	0
7	0	0	0	0	0.01	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0

*The calculated differences between the PLASMER4 solution and the thesis model solution can be partially attributed to the fact that the results of the thesis model represent elevations after 9.9 days of pumping and not 10.3 days as simulated by PLASMER4 .

** Difference (feet) = PLASMER4 output value - thesis model output value.

Table 6.2

Comparison of Potentiometric Surfaces
Calculated by PLASMER4 and Thesis Model

PLASMER4 head calculations for a simulated pumping Period of 186.9 days*
Thesis Head calculations for a simulated pumping Period of 186.7 days*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0**	0.01	0	0	0	0.01	0	0.01	0	0
2	0.01	0	0.01	0	0.01	0	0	0	0	0
3	0	0.01	0	0	0	0	0	0	0	0
4	0.01	0	0	0	0	0	0	0	0.01	0
5	0	0.01	0	0	0.01	0	0	0.01	0	0
6	0	0	0	0	0	0	0	0	0	0.01
7	0	0	0	0	0	0	0	0	0	0
8	0.01	0	0	0	0.01	0	0	0.01	0	0.01
9	0	0	0	0.01	0	0	0	0	0.01	0.01
10	0	0	0	0	0	0.01	0	0.01	0	0.01

*The calculated differences between the PLASMER4 solution and the thesis model solution can be partially attributed to the fact that the results of the thesis model represent elevations after 186.7 days of pumping and not 186.9 days as simulated by PLASMER4 .

** Difference (feet) = PLASMER4 output value - thesis model output value.

Table 6.3

Comparison of Potentiometric Surfaces Calculated by PLASMER4 and Thesis Model

PLASMER4 head calculations for a simulated pumping Period of 270 days*
Thesis Head calculations for a simulated pumping Period of 271 days*

Rows	*Columns* 1	2	3	4	5	6	7	8	9	10
1	0.01**	0.01	0.01	0.01	0	0	0.01	0.01	0	0
2	0.01	0.01	0.01	0.01	0	0.01	0	0	0.01	0
3	0.01	0.01	0	0.01	0.01	0	0.01	0.01	0	0.01
4	0.01	0.01	0.01	0	0.01	0	0	0.01	0	0.01
5	0	0	0.01	0.01	0.01	0.01	0.01	0	0	0
6	0	0.01	0	0	0.01	0.01	0	0	0	0
7	0.01	0	0.01	0	0.01	0	0	0	0	0
8	0.01	0	0.01	0.01	0	0	0	0	0	0
9	0	0.01	0	0	0	0	0	0	0	0
10	0	0	0.01	0.01	0	0	0	0	0	0

*The calculated differences between the PLASMER4 solution and the thesis model solution can be partially attributed to the fact that the results of the thesis model represent elevations after 271 days of pumping and not 270 days as simulated by PLASMER4 .

** Difference (feet) = PLASMER4 output value - thesis model output value.

comparison between the potentiometric surfaces calculated by PLASMER4 and the program presented in this thesis.

As presented in Table 6.1, 6.2 and 6.3, there is very little difference between the solution of the program presented in this thesis and the PLASMER4 output. After a simulated pumping period of approximately 10 days, the highest difference between the two program's outputs is 0.03 feet (Table 6.3). This difference is attributed to the fact that the combined solution represents the flow field after a simulated pumping period of 10.3 days and the thesis model's output represents the flow field after a simulated pumping period of 9.9 days. In the initial stages of pumping an aquifer, the applied stresses near the well cause rapid changes of the water table surface in a short period of time. Eventually, steady state conditions are approached and these changes become less noticeable. Therefore, the slight differences in simulating pumping periods is enough to create an output difference of 0.03 feet. As steady state conditions are approached in the aquifer, the program's output differences presented in Table's 6.2 and 6.3 indicate that the maximum difference between the two program's outputs has been reduced to 0.01 ft. This difference can also be attributed to the time difference between the simulated pumping periods of the two programs. Chapter 7 discusses the relevancy of the thesis model's results.

6.2 Dewatering Case Study

The realignment of the Welland Canal between Port Robinson and Port Colborne, Ontario and the building of canal underpass structures commenced in 1969 and necessitated both temporary and permanent dewatering operations of the underlying artesian bedrock aquifer (Farvolden and Nunan, 1970). The thesis program is applied to the Welland Canal Project to assess the usefulness of the model as an effective planning tool. Appendix C presents a summary of the hydrogeological background of the dewatering project (Farvolden and Nunan, 1970) in addition to a description of a computer model (Frind, 1970) that was developed for the site to predict regional dewatering impacts upon nearby potable water supplies. The emphasis of these previous studies was to predict the site's dewatering requirements and to assess the regional impact of these dewatering operations.

Four pumping centres were established to dewater the excavations for the Welland Canal project (Farvolden and Nunan, 1970). Figure 6.1 presents a schematic of the region delineating the bedrock aquifer's original potentiometric surface (Farvolden and Nunan, 1970). Figure 6.2 depicts the locations of the four pumping centres and presents the bedrock aquifer's drawdown cone observed on June 30, 1969 (Farvolden and Nunan, 1970). The observed drawdown matched closely with the potentiometric surface predicted by Frind's computer simulation (Frind, 1970). It should be noted on Figure 6.2 that potable water supplies in nearby Fonthill were impacted by a regional aquifer lowering of as much as 20 ft.

In applying the thesis model to the Welland Canal project, the regional scale of the compiled field information and the lack of detailed hydrogeologic data in the vicinity of the excavation site did not provide much freedom in having pumping wells advance through a finite difference grid to find an optimum dewatering configuration. Hence, this provided an opportunity to present another aspect of the thesis model's versatility in addition to dewatering applications. The model could be used to determine optimum recharge well configurations. Therefore, for the Welland Canal project, the

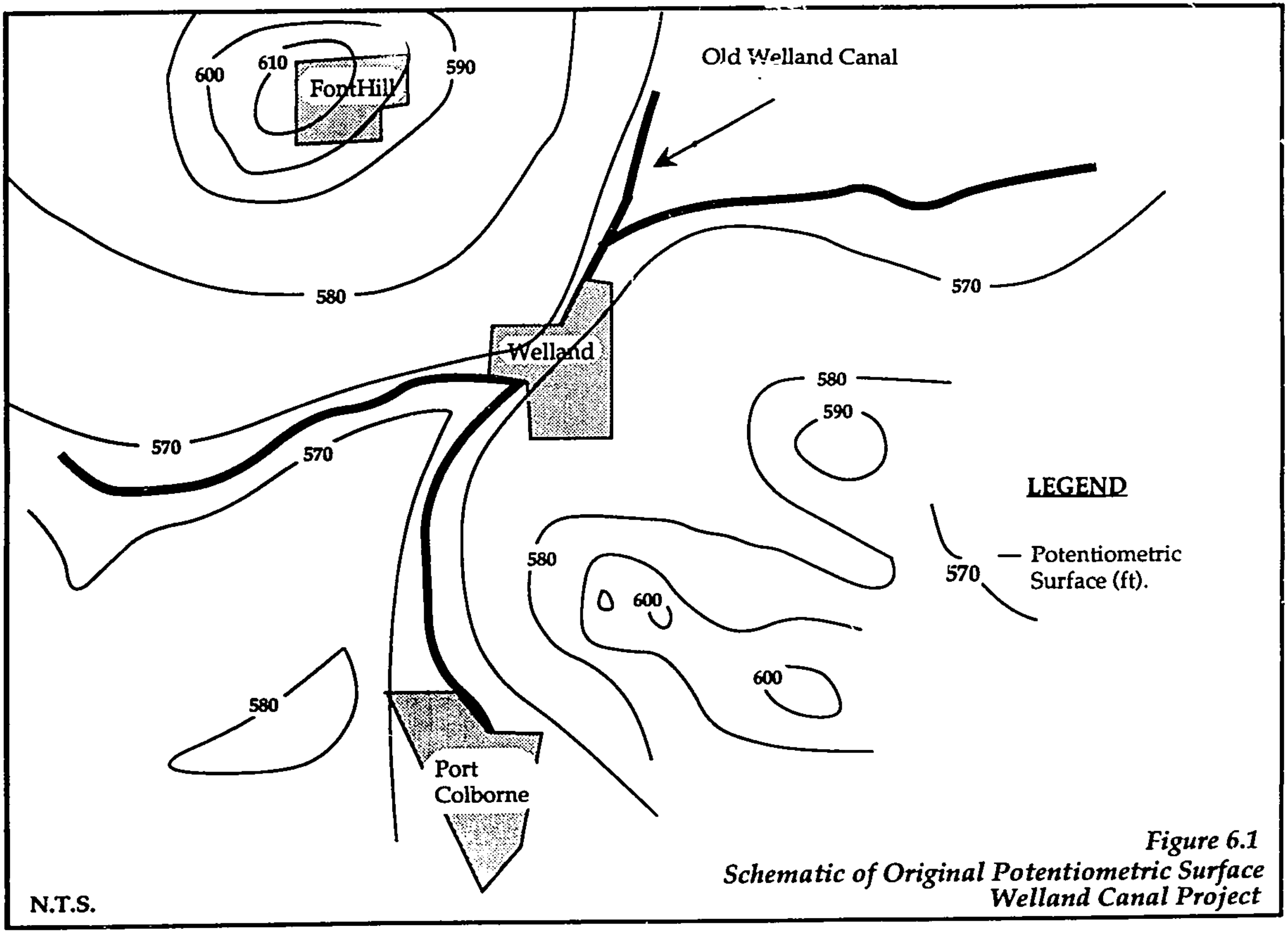


Figure 6.1
Schematic of Original Potentiometric Surface
Welland Canal Project

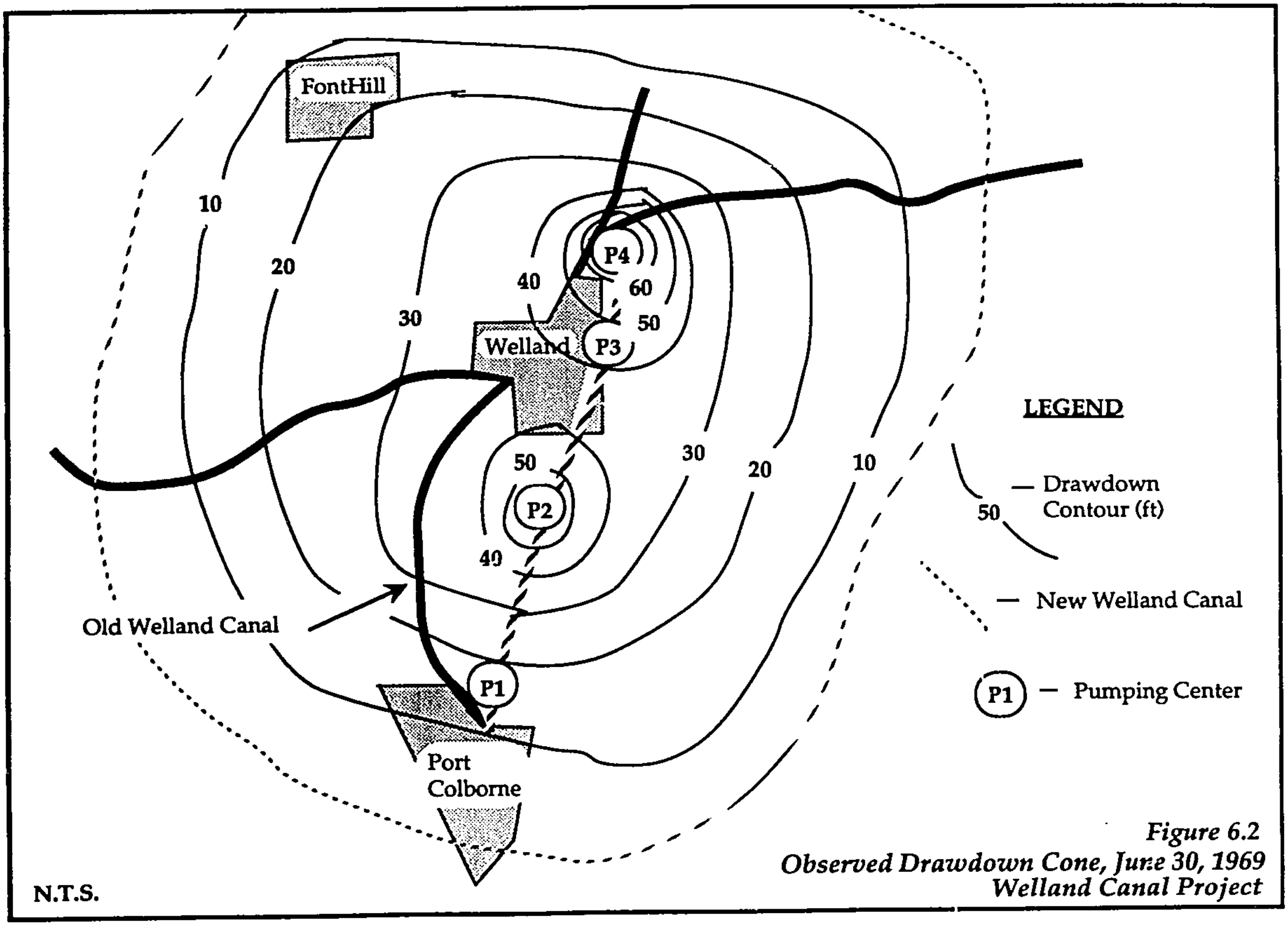


Figure 6.2
Observed Drawdown Cone, June 30, 1969
Welland Canal Project

thesis model was utilized to predict the pumping well requirements to dewater the site and to determine an optimum recharge well configuration that would limit the regional impact of the site's dewatering operations.

To accomplish this, the aquifer was discretized into a grid of ten rows and columns. Appendix D presents the model's input data that were utilized to produce the solution. After calibrating the model to accurately simulate field conditions, the model was run to determine both the site's dewatering flux requirements and an optimum recharge well configuration that would decrease the regional impact of the dewatering wells.

Figure 6.3 presents the potentiometric surface created by the optimum dewatering and recharge well configuration predicted by the thesis model. Similar to the field studies, the model utilized four pumping centres to dewater the construction site of the new Welland Canal. In addition, three recharge wells were required to minimize the the regional impacts of the dewatering operations. Figure 6.4 presents a schematic of the computer screens simulation graphics portraying the optimum recharge well configuration and pumping well requirements as determined by the thesis model. Also presented in Figure 6.4 are the user designated control points. As described in Chapters 2 and 5, the solution of the thesis model is achieved when the predicted potentiometric surface of a particular well configuration matches the designated groundwater elevations of user defined control nodes within the finite difference grid. The controlling groundwater elevations for the control points were derived from Figure 6.2. Control points 6, 7, 8, &9, the dewatering control points, are located in the vicinity of the deepest excavations for the new canal. The Control points 1, 2, 3, 4, & 5, the recharge control points, were located near the Old Welland Canal. The basis for the control point configuration was to ensure proper dewatering took place for

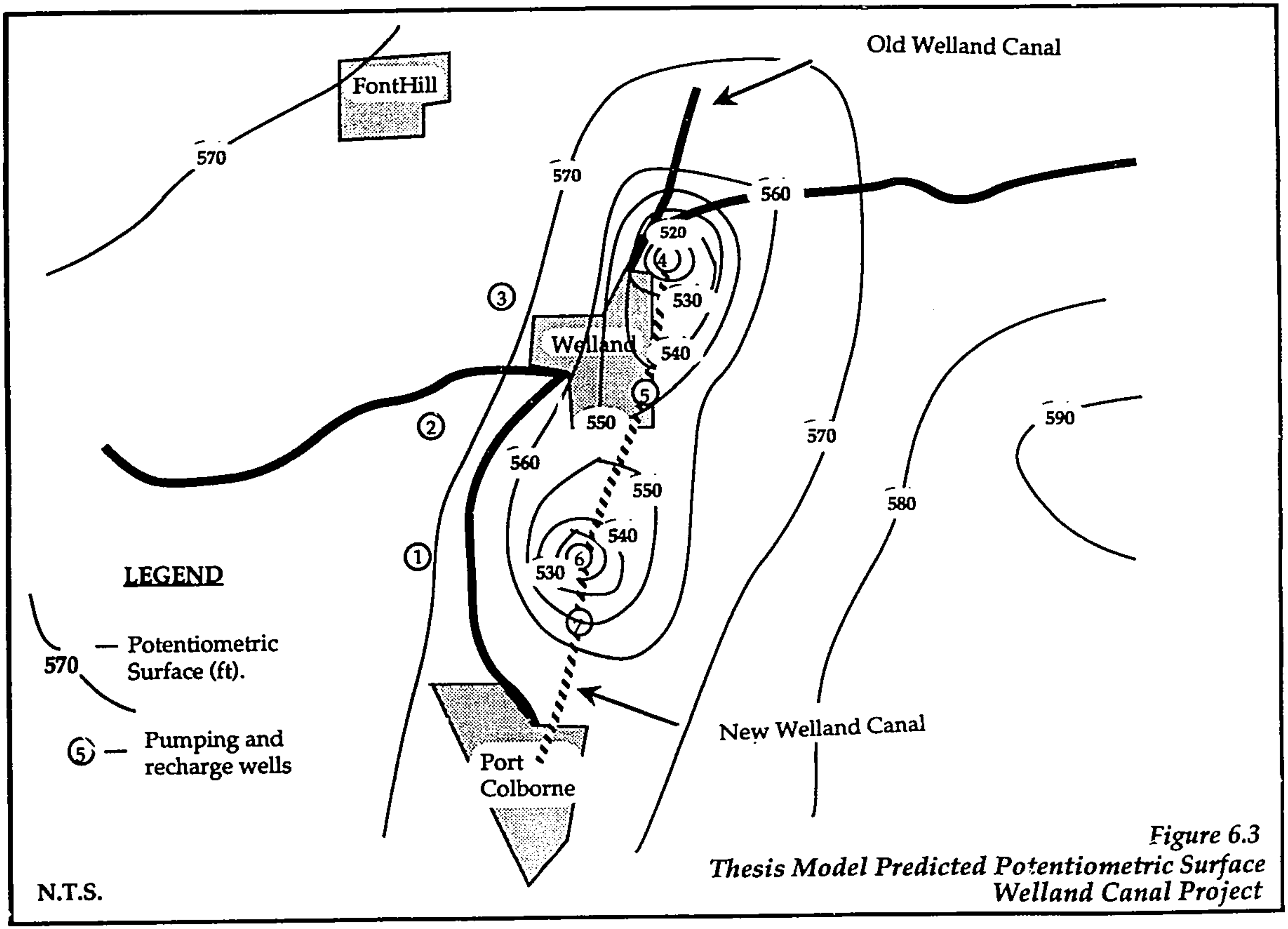


Figure 6.3
Thesis Model Predicted Potentiometric Surface Welland Canal Project

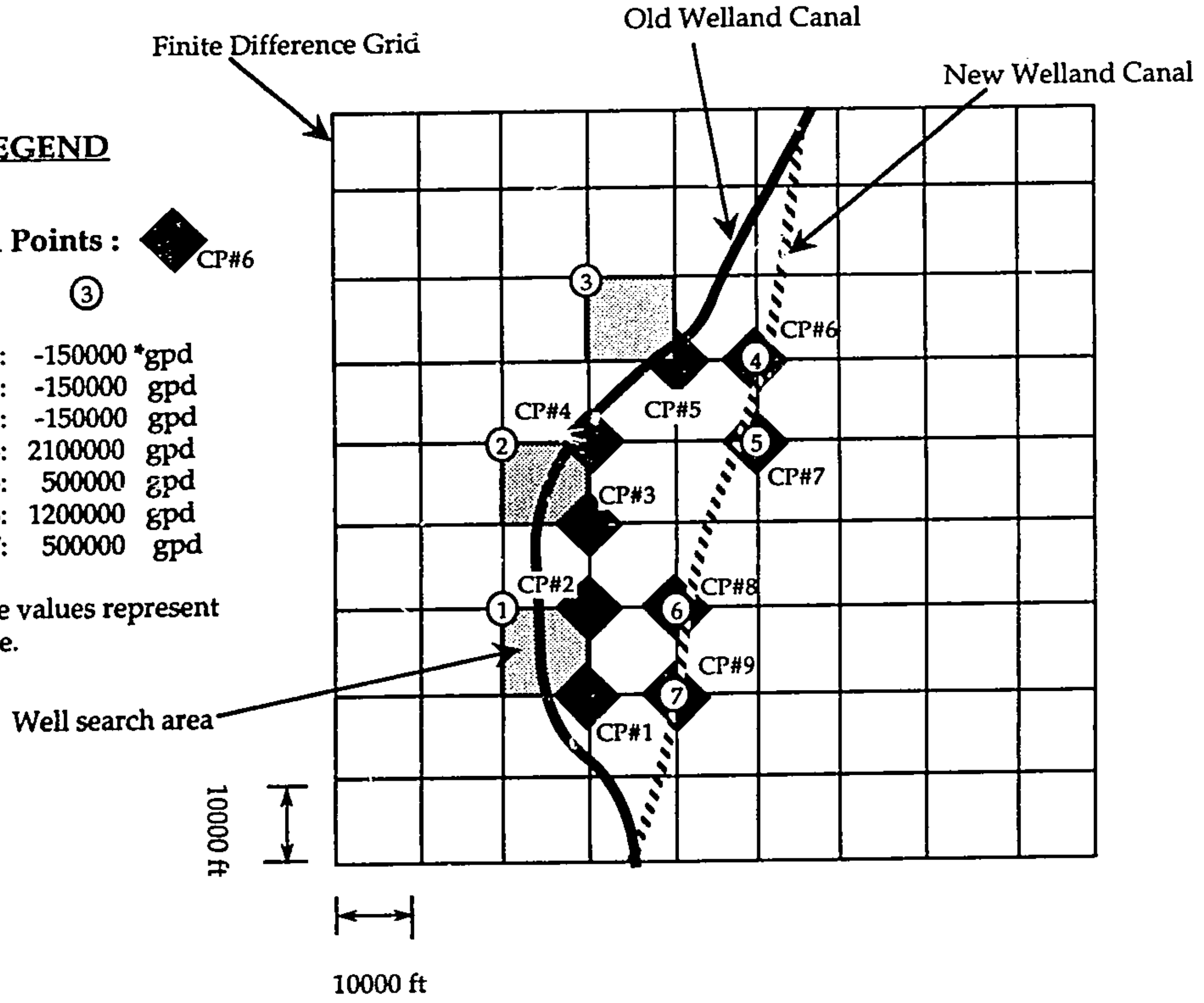
LEGEND

Control Points : ◆ CP#6

Wells : ③

Well No. 1:	-150000 *gpd
Well No. 2:	-150000 gpd
Well No. 3:	-150000 gpd
Well No. 4:	2100000 gpd
Well No. 5:	500000 gpd
Well No. 6:	1200000 gpd
Well No. 7:	500000 gpd

*- Negative values represent recharge.



N.T.S.

Figure 6.4

Optimum Dewatering Solution Schematic

the project and the aquifer regional dewatering was kept to a minimum west of the old canal.

To compare the differences in regional dewatering trends predicted by the thesis program and the observed drawdown cone as a result of actual dewatering operations, a loose comparison can be made between the predicted potentiometric surface of the thesis model's optimum solution presented in Figure 6.3 and the observed drawdown cone portrayed in Figure 6.2. Comparing Figure 6.2 and 6.3, it is evident that the thesis program's solution closely simulated the earlier described dewatering requirements of the site, while reducing the regional dewatering of the aquifer, notably for the Town of Fonthill. Table 6.4 summarizes the thesis model's output for the dewatering application by comparing the required groundwater elevations for the eight control points with the programs predicted elevations for the solution well configuration. Based upon the results presented in Table 6.4, it is evident that the model's solution well configuration would create a potentiometric surface that fell within the user defined tolerance of error for the required groundwater elevations of the control points as well as limiting the amount of regional dewatering.

The thesis program output summary presenting the model's solution is presented in Appendix E. The program output gives the coordinates of a model determined optimum dewatering well configuration within the finite difference grid, the site's control point's desired and calculated potentiometric elevations, and the predicted potentiometric surface of the flow field after being stressed by the well field. Chapter 7 discusses the usefulness of the model presented in this thesis in light of the dewatering application solution.

Table 6.4

Comparison of Predicted Potentiometric Surface and Required Control Point Elevations*
Welland Canal Project

Control Point Number	*Control Point Coordinates*	*Control Point Desired Elevation***	*Control Point Tolerance of Error***	*Thesis Model's Predicted Elevation***
1	(4,8)	570	+/-10	566.09
2	(4,7)	570	+/-10	563.32
3	(4,6)	570	+/-10	568.63
4	(4,5)	570	+/-10	570.28
5	(5,4)	560	+/-5	557.27
6	(6,4)	515	+/-5	519.72
7	(6,5)	540	+/-5	544.83
8	(5,7)	525	+/-5	525.1
9	(5,8)	555	+/-5	559.62

*Taken from the thesis model's dewatering output contained in Appendix E
**Units in feet.

6.3 Plume Control Case Study

To assess the usefulness of the program presented in this thesis as an effective planning tool to help control contaminated plume migration and begin aquifer restoration, the model is applied to a US EPA documented case study. The site is known as the Emerson Electric Company site and is located in Altamonte Springs, Florida (US EPA, 1989). The site remediation is regulated by the Florida State Environmental Protection Office. At the site, organic and heavy metal contaminants have entered into a sandy water table aquifer as a result of discharge from the industrial plant to a septic tile system without pretreatment (US EPA, 1989). Extraction wells were implemented and have effectively restored the aquifer to less than detectable limits for the contaminants observed (US EPA, 1989). Appendix F presents a more detailed hydrogeologic description of the site and the designed capture well system that was implemented to restore the aquifer (US EPA, 1989).

As noted in Appendix F, the consultants hired by the Emerson Electric Company designed an aquifer restoration system consisting of five pumping wells, pumping at a combined rate of 43 200 US gal/day, concentrated in an area having anomalously high conductivity readings (US EPA, 1989). Figure 6.5 presents a schematic of the site and the position of the consultant's purge wells. As a result of this design, the water table was lowered by almost 20 ft. in the wells' vicinity. As noted in Figure 6.5, the contaminated water table aquifer is only 20 ft thick, hence the consultants dewatering scheme created a large unsaturated zone that could not be purged efficiently. Figure 6.6 presents the potentiometric surface created by the consultant's five well purge system.

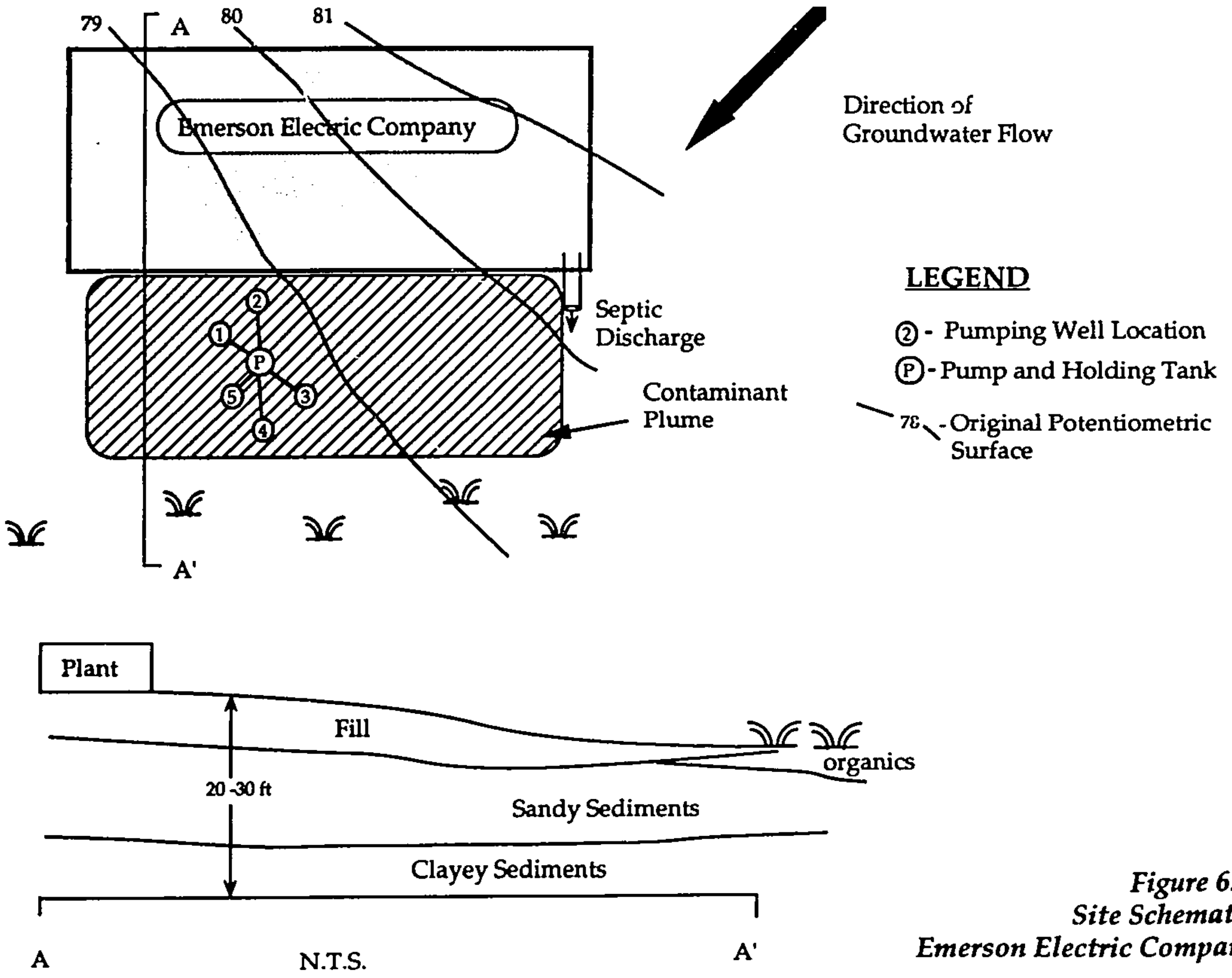
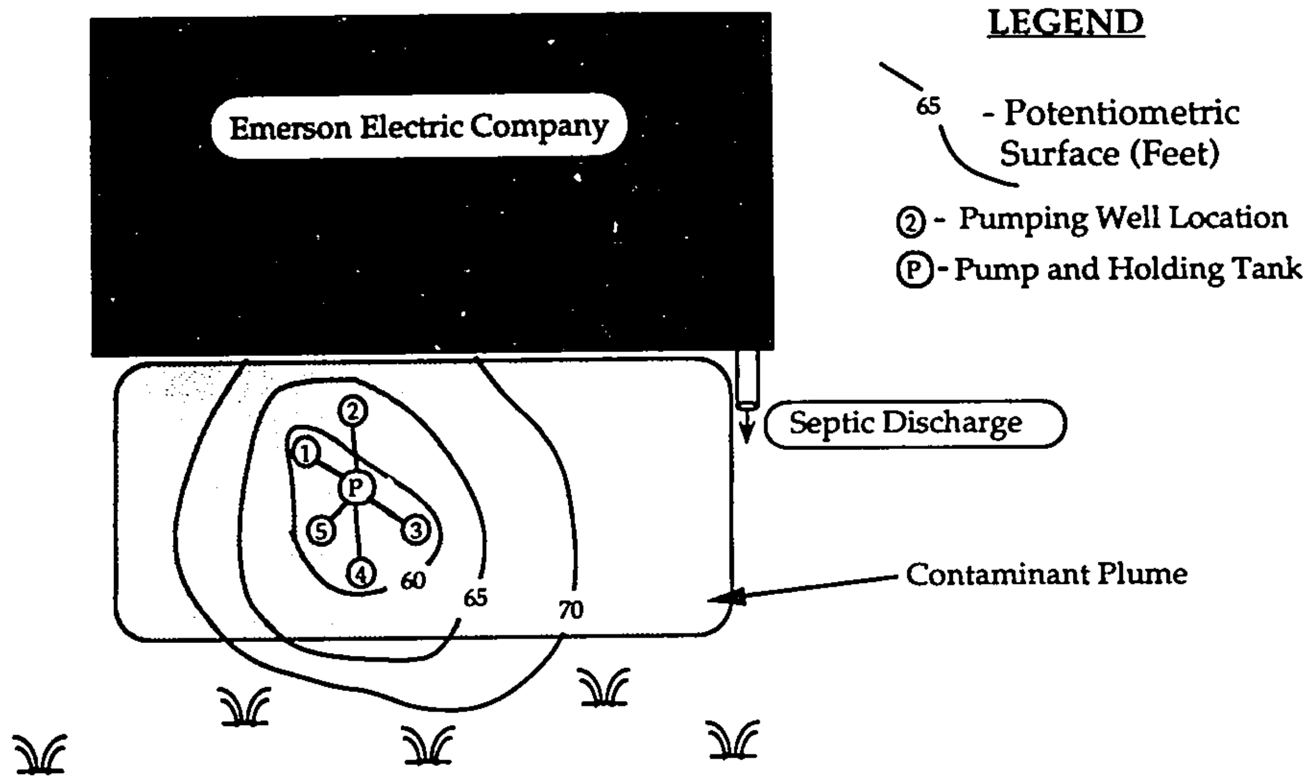


Figure 6.5
Site Schematic
Emerson Electric Company


Emerson Electric Company
LEGEND
65 - Potentiometric Surface (Feet)
2 - Pumping Well Location
P - Pump and Holding Tank
Septic Discharge
Contaminant Plume
1
2
3
4
5
P
60
65
70

N.T.S.

Figure 6.6
Observed Potentiometric Surface Schematic
Emerson Electric Company

To apply the thesis model to this plume capture case study, the aquifer was discretized into a grid of ten rows and columns. Appendix G presents the model's input data that was utilized to produce the solution. After calibrating the model to the Emerson Electric Company property, a solution was derived for the application and is presented in Appendix H.

As noted in Chapter 3, the strategy to designing an efficient aquifer restoration system is to create as large a capture zone as possible to contain the contamination, with a system that will minimally dewater the site, so that the soil-water matrix can be effectively purged. Figure 6.7 presents a schematic of the computer's optimum well configuration solution depicting a system of two pumping wells. As noted from Figure 6.7, most of the contaminated areas on the property are enclosed within the solution well's capture zones. To maximize capture areal coverage and minimize dewatering it was found that the solution required a combined pumping rate of 14 000 US gal/day. Figure 6.8 presents the predicted potentiometric surface of the thesis model's solution.

In addition to providing good site coverage, a comparison between Figure's 6.6 and 6.8 indicates that the thesis model's solution well configuration would be a more efficient aquifer purging system. The consultant's solution dewaters the property approximately 15 ft or almost one third of the aquifer's thickness more than the thesis model's solution.

As noted in Appendix H, the model's solution output contains the optimum well placement configurations to capture the plume, the capture width of each well, as well as the resultant potentiometric surface produced as a result of the capture wells spatial configuration. Chapter 7 discusses the accuracy and suitability of the thesis model's solution to the plume capture application.

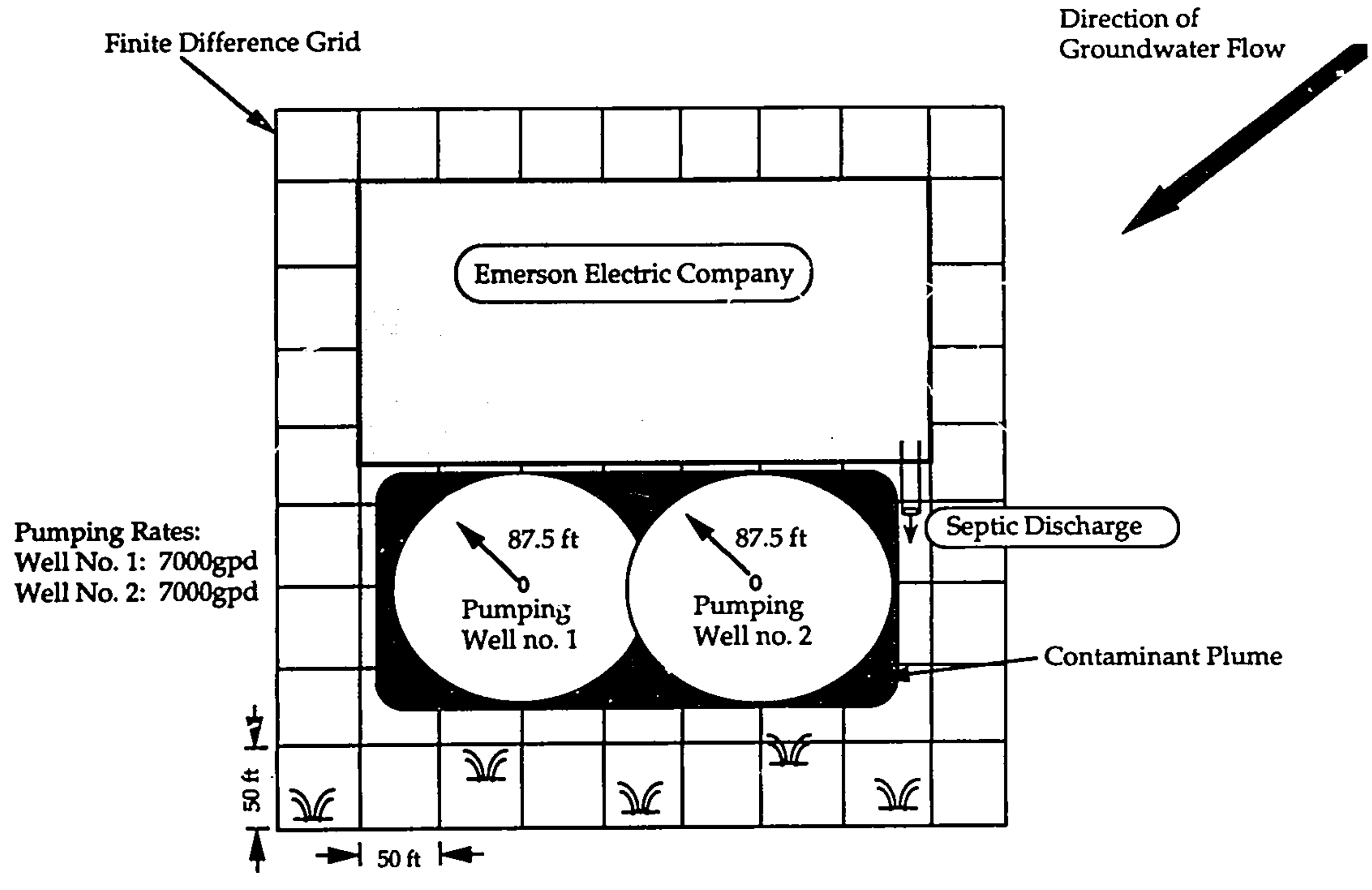
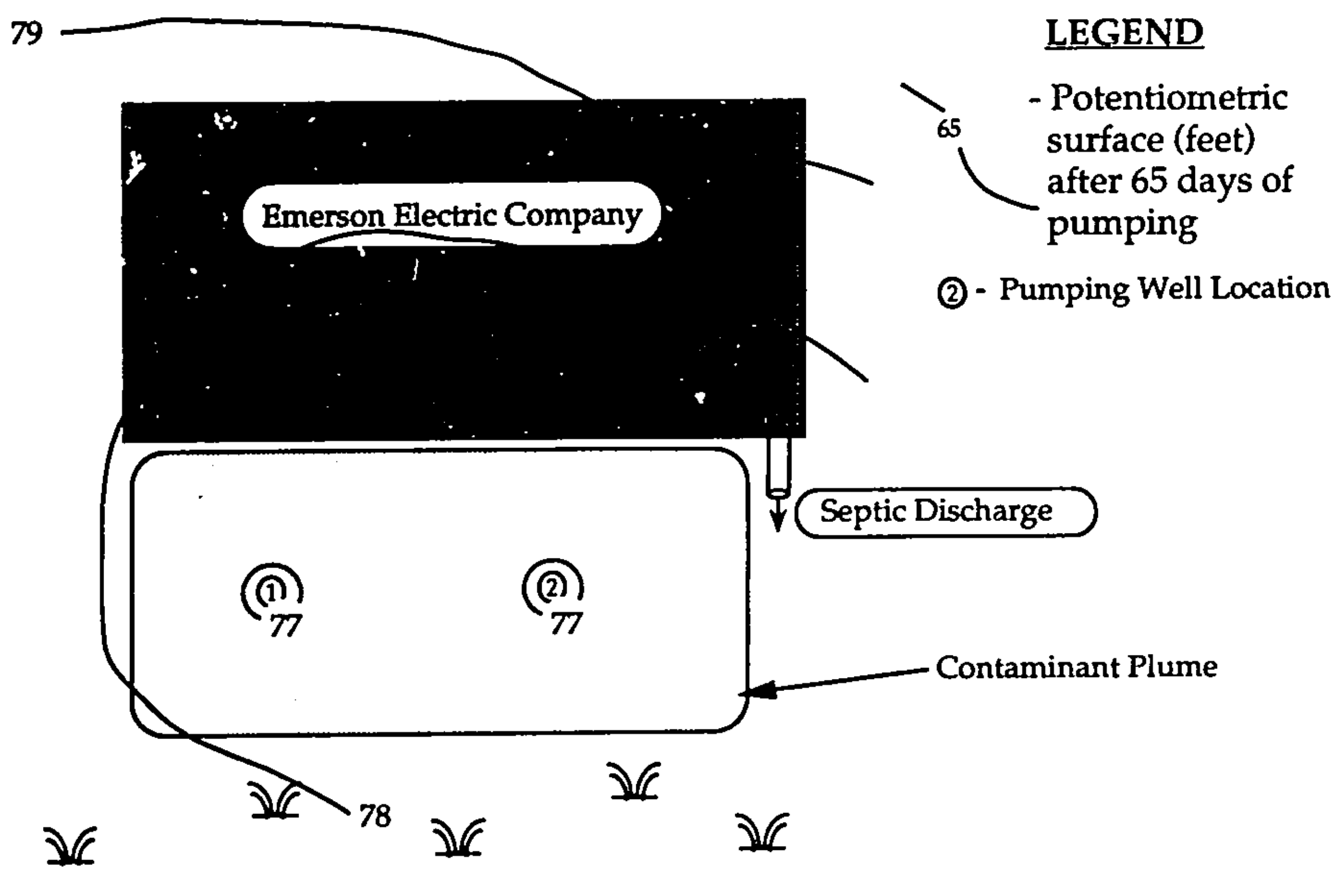


Figure 6.7

Plume Solution Schematic


79
Emerson Electric Company
LEGEND
65
- Potentiometric surface (feet) after 65 days of pumping
② - Pumping Well Location
Septic Discharge
①
77
②
77
Contaminant Plume
78

N.T.S

Figure 6.8
Thesis Model Predicted Potentiometric Surface Schematic
Emerson Electric Company

7.0 DISCUSSIONS

To assess the thesis program's suitability and efficiency as a practical design tool for site dewatering and plume management applications, the model was applied to three case studies described in detail in Chapter 6. The following sections discuss the observations that were made concerning the model based upon the results of the three case study applications.

7.1 Finite Difference Approximations

The comparison case presented in Section 6.1 indicates that the thesis model can simulate the potentiometric surface of a groundwater flow field with a high degree of accuracy. The solution to the finite difference approximations of the groundwater flow equation (Equation 4.1) forms the basis for the thesis dewatering model. The program's plume management model utilizes the solution to the flow equation to determine the aquifer's potentiometric surface resulting from the well configuration that the model has determined would best capture or manage the contaminant plume. Therefore, if the model has been properly sensitized to the site, it should provide reasonably accurate results for most applications.

7.2 Dewatering Case Study

The thesis model's solution to the Welland Canal case study is presented in Figure 6.1 and Appendix E. The solution was found to adequately dewater the excavation while eliminating regional dewatering impacts upon water supplies of rural residents west of the old Welland Canal.

The dewatering requirements and effects determined by the thesis model closely matches the requirements predicted by the consulting groundwater specialists who studied the site. In addition, the configuration of recharge wells determined by the program would appear to greatly reduce the regional impact from the dewatering operations. The computer derived recharge well configuration is practical, in that the wells are located adjacent to the old Welland Canal, a potential water supply for the recharge operations. The furthest distance that water would need to be transported from the canal would be approximately 12000 ft.

Based upon the results of the dewatering case study comparison, it is evident that the program presented in this thesis is a practical and effective tool for the pre-design stages of planning a dewatering management system. The total time taken to complete this exercise subsequent to collection of the background hydrogeological field data was approximately seven (7) hours utilizing an IBM-XT operating at 4.7 MH with a math coprocessor installed. This time period consisted of one (1) hour of data input time, two (2) hours of sensitizing the model to the site's field observations, and four (4) hours simulation time. The simulation time entailed running the model (approximately fifteen minutes (15) minutes for each run), interpretation and assessment of the program's output, followed by further simulations after adjusting various input parameters.

7.3 Plume Control Case Study

The thesis model's optimum plume management solution to the Emerson Electric Company case study is presented in Appendix H. Comparing the thesis model solution with the consulting engineer's solution presented in Appendix F, the following differences are noted:

i) The consulting engineers designed an aquifer remediation system that included five (5) wells pumping at a combined rate of 43 200 US gal/day. The thesis program determined that the optimum aquifer remediation system consisted of two (2) wells pumping at a combined rate of 14 000 US gal/day.

ii) The consulting engineers at the site concentrated all five (5) wells in the vicinity of the highest observed contaminant concentrations and predicted that the water table would be lowered by approximately 20 ft. The thesis program determined that to optimally capture the on-site contaminant plume, two (2) wells were required and should be positioned as shown in Figure 6.2. As a result of this well configuration, the thesis model predicted that the water table would be lowered by approximately five (5) ft.

It would appear that the consulting engineers responsible for designing the aquifer restoration system at the Emerson Electric Plant had over designed their system, and that the thesis program has determined a better plume management system for the site. Reasons for this conclusion are:

i) The consulting engineers remediation design lowers the water table fifteen (15) ft further than the thesis program's design. This means that the capture zone created by the concentrated well field designed by the site engineers is not flushing any contamination, adhered to the soil matrix or present within undrained pore spaces, within the dewatered fifteen (15) ft. The thesis program's solution has optimized the largest possible capture zone while minimally dewatering the subsurface.

ii) Due to the long periods of time that the aquifer restoration pumping system has to be maintained, unnecessary site dewatering, present in the site engineer's design, could lead to differential settlement and foundation failure of the Emerson Electric Company's plant. Settlement could have the potential for releasing further contamination into the environment through cracking of subsurface drainage systems or storage tanks.

8.0 CONCLUSIONS AND RECOMMENDATIONS

The program presented in this thesis is a good pre-design tool to plan a dewatering or plume capture system. Based upon the application of this program to the case studies presented in Chapter 6, the following conclusions are made:

i) The program successfully has been applied to a dewatering and plume management case study and proven useful.

ii) The program simulates a wide variety of observed field conditions accurately, and has compared well with the solutions of existing flow models. In utilizing basic equations of groundwater flow, the model's applications are not restricted or require extensive data bases, and the use of empirical parameters, that are determined through experimentation to describe 'black box' processes, are not required.

ii) The program is design to be user friendly which greatly reduces the user time required for data input and manipulation. Attempts have been made to make the program as visually pleasing and as graphically oriented as possible for the user. The program's menu system allows the user to advance quickly to various parts of the program and enter or change parameter values. The development and incorporation of a spreadsheet greatly reduces the manipulation time required when managing the many site characterizing matrices used for finite difference solutions.

Further studies that may improve the computer model are:

i) Development and incorporation of a two dimensional contaminant transport procedure that would simulate contaminant travel through the aquifer from the contaminant source to the capture wells. This would provide invaluable information to the user regarding predicting aquifer restoration pumping times, predicting the water quality changes with time of collected waste waters, and predicting contaminant concentrations remaining within the aquifer.

References

Driscoll, Fletcher D., 1986. Groundwater and Wells. Johnson Division, St. Paul, Minnesota, 1089 p.

Farvolden, R.N., and Nunan, J.P., 1970. Hydrogeologic aspects of dewatering at Welland. Canadian Geotechnical Journal, 7, 194-204.

Freeze, R.A., and Cherry, J.A., 1979. Groundwater. Prentice-Hall, Englewood Cliffs, N.J., 523 p.

Frind, E.O., 1970. Theoretical analysis of aquifer response due to dewatering at Welland. Canadian Geotechnical Journal, **7**, 205-216.

Frind, E.O., Sudicky, E.A., Schellenberg, S.L., 1987. Micro-scale modelling in the study of plume evolution in heterogeneous media. Stochastic Hydrology and Hydraulics, **1(4)**, 263-279.

James, M.L., Smith, G.M., and Wolford, J.C., 1977. Applied numerical methods for digital computation with Fortran and CSMP, second edition. Harper & Row, New York, 687p.

Javandel, I., Doughty, C., and Tsang, C.F., 1984. Groundwater transport: handbook of mathematical models. Water Resources Monograph Series 10, American Geophysical Union, Washington, D.C., 228p.

Keely, J.F., and Tsang, C.F., 1983. Velocity plots and capture zones of pumping centers for groundwater investigations. Groundwater. **21(6)**, 701-714.

Kinzelbach, W.K., Dillon, P., and Jenson, K.H., 1988. State of the art of existing groundwater quality models of the saturated zone and experience with their application in agricultural problems. International Symmposium on Water Quality Modeling of Agricultural Non-Point Sources, Utah State University, Logan, Utah, 24p.

Konikow, L.F., and Bredehoeft, J.D., 1978. Computer model of two-dimensional solute transport and dispersion in groundwater. Techniques of Water Resource Investigations of the U.S. Geological Survey, Chapter No. C2, Scientific Publications, Washington, D.C. 80 p.

Peaceman, D.W., and Rachford, Jr., H.H., 1955. The numerical solution of parabolic and elliptical differential equations. Jour. Soc. Ind. Appl. Math. **5(2)**.

Pinder, G.F., and Bredehoeft, J.D., 1968. Application of the digital computer for aquifer evaluation. Water Resources Research, **4(5)**, 1069-1093.

Plate, E.J., and Duckstein, L., 1988. Stochastic aspects of water quality modeling for non-point sources. NWWA Conference, 35p.

Prickett, T.A., 1975. Modelling techniques for groundwater evaluation. Advances in Hydroscience, Academic Press, New York, **10** .

Prickett, T.A., and Lonnquist, C.G., 1971. Selected digital computer techniques for groundwater resource evaluation. Illinois State Water Survey Bulletin 55, 63p.

Todd, D.K., 1980. Groundwater Hydrology. 2nd edition. John Wiley and Sons, New York, NY. 325 p.

Walton, W.C., 1984. Practical aspects of groundwater modeling. National Water Well Association, Ohio, 588p.

Woolhiser, D.A., Jury, W.A., Neilsen, D.R., 1989. DRAFT-Effects of spatial and temporal variability on water quality model development. NWWA Conference, 21p.

US Environmental Protection Agency, 1989. Evaluation of ground-water extraction remedies. Office of Emergency and Remedial Response, Washington, DC., EPA/540/2-89/054, **2(2).**

Appendix A

Computer Program Listing

```
{$M 65520,0,400000}
Program menusys;
Uses Crt,Dos,Graph,globals,hydrosys,spredsht,solutsys;


procedure Load_spread(s:string80; var a:matrixtype);
var f:text;
    x,y : integer;
    ts : string80;
begin
   assign(f,s);
   reset(f);
   for x :=1 to matrixsize do begin
      for y := 1 to matrixsize do begin
         readln(f,ts);
         a^[x,y] := rvalue(ts);
      end;
   end;
   close(f);
end;


function exist(s:string80):boolean;
var t : text;
    b : boolean;
begin
   {$I-}
   assign(t,s);
   reset(t);
   b := (ioresult = 0);
   close(t);
   exist := b;
end;


procedure Load_spreads;
begin
   if exist(sfilename+'.H') then begin
      load_spread(sfilename+'.H',H);
   end;
   if exist(sfilename+'.T') then load_spread(sfilename+'.T',T);
   if exist(sfilename+'.S') then load_spread(sfilename+'.S',Stor);
   if exist(sfilename+'.L') then load_spread(sfilename+'.L',Leakance);
   if datarec.plumeprob then begin
      if exist(sfilename+'.CON') then load_spread(sfilename+'.CON',Contaminant_C
oncentration);
   end;
   if datarec.leaky_artesian_conditions then begin
      if exist(sfilename+'.ADK') then load_spread(sfilename+'.ADK',aqtdperm);
      if exist(sfilename+'.ADS') then load_spread(sfilename+'.ADS',aqtdthickness
);
```

```
      if exist(sfilename+'.SUH') then load_spread(sfilename+'.SUH',aqtdthickness
);
   end;
   if not datarec.confined then begin
      if exist(sfilename+'.AQB') then load_spread(sfilename+'.AQB',aquiferbase);
      if exist(sfilename+'.AQK') then load_spread(sfilename+'.AQK',aqfrperm);
   end;
   if datarec.induced_infiltration then begin
      if exist(sfilename+'.WBK') then load_spread(sfilename+'.WBK',waterbodybedp
erm);
      if exist(sfilename+'.WBS') then load_spread(sfilename+'.WBS',waterbodybedt
hk);
      if exist(sfilename+'.WBA') then load_spread(sfilename+'.WBA',waterbodyarea
);
      if exist(sfilename+'.WBB') then load_spread(sfilename+'.WBB',waterbodybede
lev);
      if exist(sfilename+'.WBE') then load_spread(sfilename+'.WBE',waterbodysurf
elev);
   end;
end;


Procedure Plume_Characteristics;
Const
   dwidth : integer = 10;
   bh     : integer = 24;
   bw     : integer = 78;
   sr     : integer = 2;
   sc     : integer = 2;
   title  : string[30] = 'Plume Characteristics';
   c      : integer = 63;
var
   ch           : char;
   ppos         : integer;
   maxnumparams : integer;
   diff         : integer;
   x            : integer;
   exit_window  : boolean;
   maxwindow    : integer;
   exit_menu    : boolean;
   exit         : boolean;
   level        : integer;
   exitlev      : boolean;
begin
   clrscr;
   param_box(sr,sc,bw,bh);
   s := 'Enter the number of rectangular areas that';
   putat((80-length(s))shr 1, sr+7,s);
   s := 'will best outline the plume:';
   putat((80-length(s))shr 1, sr+8,s);
   textcolor(0);
   textbackground(1);
   exit_menu := false;
   exit := false;
   diff:= 0;
   ppos:= sr+8;
   maxnumparams:=1;
   level := 1;
   exitlev := false;
   repeat
      case level of
         1 : begin
                repeat
                   gotoxy(57,ppos);
                   inum := datarec.nplume;
                   s := istringequiv(inum);
                   len:=length(s);
                   lendiff:=2-len;
                   inverseon;
                   for i:=1 to lendiff do begin
                     write(' ');
                   end;
                   write(s);
                   gotoxy(57,ppos);
                   s := getnum(2,s);
                until (ivalue(s) in [1..10]);
                exitlev := true;
                inverseoff;
                parameter[1] := s;
             end;
      end;
   until (exitlev);
   datarec.nplume := ivalue(parameter[1]);
   if datarec.no_pumping_wells<>0 then begin
      exit_window:= false;
      mrxwindow:= datarec.nplume;
      window:= 1;
      repeat
         clrscr;
         param_box(sc,sr,bw,bh);
         putat((72-length(title))shr 1,sr+1,title);
         putat(wherex-length(title), sr+3,
                    'Plume characterizing Rectangle #');
         write(window);
         putat(sc+2,sr+6,
'Enter x coordinate for top left of plume characterizing rectangle: ');
         putat(sc+2,sr+9,
'Enter y coordinate for top left of plume characterizing rectangle: ');
         putat(sc+2,sr+12,
'Enter x coordinate for bottom right of plume characterizing rectangle: ');
         putat(sc+2,sr+15,
'Enter y coordinate for bottom right of plume characterizing rectangle: ');
         diff:= 4;
         ppos:= sr+6;
         level := 1;
         exitlev := false;
         repeat
            case level of
               1 : begin
                      repeat
```

```
                          gotoxy(74,ppos);
                          inum := datarec.plume[window].ptlx;
                          s := istringequiv(inum);
                          len:=length(s);
                          lendiff:=2-len;
                          inverseon;
                          for i:=1 to lendiff do begin
                            write(' ');
                          end;
                          write(s);
                          gotoxy(75,ppos);
                          s := getnum(2,s);
                       until (ivalue(s) in [2..datarec.nc-2]) or (lastchar = #72)
;
                       if lastch = #72 then level := level-1;
                       inverseoff;
                       parameter[1] := s;
                   end;
               2 : begin
                       repeat
                          gotoxy(74,ppos+3);
                          inum := datarec.plume[window].ptly;
                          s := istringequiv(inum);
                          len:=length(s);
                          lendiff:=2-len;
                          inverseon;
                          for i:=1 to lendiff do begin
                            write(' ');
                          end;
                          write(s);
                          gotoxy(75,ppos+3);
                          s := getnum(2,s);
                       until (ivalue(s) in [2..datarec.nr-2]) or (lastchar = #72)
;
                       if lastch = #72 then level := level-2;
                       inverseoff;
                       parameter[2] := s;
                   end;
               3 : begin
                       repeat
                          gotoxy(75,ppos+6);
                          inum := datarec.plume[window].pbrx;
                          s := istringequiv(inum);
                          len:=length(s);
                          lendiff:=2-len;
                          inverseon;
                          for i:=1 to lendiff do begin
                            write(' ');
                          end;
                          write(s);
                          gotoxy(76,ppos+6);
                          s := getnum(2,s);
                       until (ivalue(s) in [2..datarec.nc-2]) or (lastchar = #72)
;
                       if lastch = #72 then level := level-2;
```

```
                     inverseoff;
                     parameter[3] := s;
                  end;
               4 : begin
                     repeat
                        gotoxy(75,ppos+9);
                        inum := datarec.plume[window].pbry;
                        s := istringequiv(inum);
                        len:=length(s);
                        lendiff:=2-len;
                        inverseon;
                        for i:=1 to lendiff do begin
                          write(' ');
                        end;
                        write(s);
                        gotoxy(76,ppos+9);
                        s := getnum(2,s);
                     until (ivalue(s) in [2..datarec.nr-2]) or (lastchar = #72)
;
                     if lastch = #72 then level := level-2
                                     else exitlev := true;
                     inverseoff;
                     parameter[4] := s;
                  end;
            end;
            level := level + 1;
         until (exitlev);
         with datarec.plume[window] do begin
            ptlx:= ivalue(parameter [1]);
            ptly:= ivalue(parameter [2]);
            pbrx:= ivalue(parameter [3]);
            pbry:= ivalue(parameter [4]);
            ptrx:= pbrx;
            ptry:= ptly;
            pblx:= ptlx;
            pbly:= pbry;
         end;
         window:= window + 1;
         if window > maxwindow then exit_window := true;
      until exit_window;
   end;
end;


Procedure Contaminant_Param;
Const
   Pwidth : integer =10;
   bh     : integer=24;
   bw     : integer=78;
   sr     : integer=2;
   sc     : integer=2;
   title  : string[30]='Contaminant Arrival Parameters';
   c      : integer=63;
var
```

```
   ch           : char;
   exit         : boolean;
   ppos         : integer;
   maxnumparams : integer;
   diff         : integer;
   x            : integer;
   exit_windows : boolean;
   maxwindow    : integer;
   window       : integer;
   exit_menu    : boolean;
   level        : integer;
   exitlev      : boolean;
begin
   clrscr;
   param_box(sr,sc,bw,bh);
   putat((80-length(title))shr 1,sr+1,title);
   putat(sc+3,sr+4,'Enter average bulk mass soil density: ');
   putat(sc+3,sr+6,'Enter average aquifer porosity: ');
   putat(sc+3,sr+8,'Enter contaminants soil/water partition coef. (Kd): ');
   putat(sc+3,sr+10,'Enter dispersivity value as percentage of travel distance:
');
   putat(sc+3,sr+12,'Enter contaminants diffusion coefficient in soil: ');
   putat(sc+3,sr+14,'Enter time increment for arrival concentrations (days): ');
   putat(sc+3,sr+16,'Enter total time for arrivals to be calculated (days): ');
   textcolor(0);
   textbackground(1);
   exit_menu:=false;
   exit:=false;
   diff:= 2;
   ppos:= sr+4;
   level := 1;
   exitlev := false;
   repeat
      case level of
         1 : begin
                repeat
                   gotoxy(sc+42,ppos);
                   rnum := datarec.bulk_density;
                   s := rstringequiv(rnum,6);
                   len:=length(s);
                   lendiff:=6-len;
                   inverseon;
                   for i:=1 to lendiff do begin
                     write(' ');
                   end;
                   write(s);
                   gotoxy(sc+42,ppos);
                   s := getnum(6,s);
                until ((rvalue(s) >=0.0) and (rvalue(s)<=999999)) or
                      (lastchar = #72);
                if lastch = #72 then level := level-1;
                inverseoff;
                parameter[1] := s;
             end;
         2 : begin
```

```
       repeat
          gotoxy(sc+42,ppos+2);
          rnum := datarec.porosity;
          s := rstringequiv(rnum,4);
          len:=length(s);
          lendiff:=4-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+42,ppos+2);
          s := getnum(4,s);
       until ((rvalue(s) >=0.00) and (rvalue(s)<=1.0)) or
             (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[2] := s;
    end;
3 : begin
       repeat
          gotoxy(sc+56,ppos+4);
          rnum := datarec.Kd;
          s := rstringequiv(rnum,9);
          len:=length(s);
          lendiff:=9-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+56,ppos+4);
          s := getnum(9,s);
       until ((rvalue(s) >=0.0) and (rvalue(s)<=999999999)) or
             (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[3] := s;
    end;
4 : begin
       repeat
          gotoxy(sc+64,ppos+6);
          rnum := datarec.dispersivity;
          s := rstringequiv(rnum,4);
          len:=length(s);
          lendiff:=4-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+65,ppos+6);
          s := getnum(5,s);
       until ((rvalue(s) >=0.0) and (rvalue(s)<=1.0)) or
             (lastchar = #72);
```

```
         if lastch = #72 then level := level-2;
         inverseoff;
         parameter[4] := s;
      end;
   5 : begin
         repeat
            gotoxy(sc+54,ppos+8);
            rnum := datarec.diffusion;
            s := rstringequiv(rnum,9);
            len:=length(s);
            lendiff:=9-len;
            inverseon;
            for i:=1 to lendiff do begin
              write(' ');
            end;
            write(s);
            gotoxy(sc+54,ppos+8);
            s := getnum(9,s);
         until ((rvalue(s))>=0.0) and (rvalue(s)<=999999999))
            or (lastchar = #72);
         if lastch = #72 then level := level-2;
         inverseoff;
         parameter[5] := s;
      end;
   6 : begin
         repeat
            gotoxy(sc+61,ppos+10);
            inum := datarec.timeincrement;
            s := istringequiv(inum);
            len:=length(s);
            lendiff:=6-len;
            inverseon;
            for i:=1 to lendiff do begin
              write(' ');
            end;
            write(s);
            gotoxy(sc+66,ppos+10);
            s := getnum(11,s);
         until ((ivalue(s) >= 0) and (ivalue(s)<=999999))
               or (lastchar = #72);
         if lastch = #72 then level := level-2;
         inverseoff;
         parameter[6] := s;
      end;
   7 : begin
         repeat
            gotoxy(sc+61,ppos+12);
            inum := datarec.totaltime;
            s := istringequiv(inum);
            len:=length(s);
            lendiff:=9-len;
            inverseon;
            for i:=1 to lendiff do begin
              write(' ');
            end;
```

```
                  write(s);
                  gotoxy(sc+67,ppos+12);
                  s := getnum(15,s);
               until ((ivalue(s) >= 0) and (ivalue(s)<=999999999))
                     or (lastchar = #72);
               if lastch = #72 then level := level-2
                                else exitlev := true;
               inverseoff;
               parameter[7] := s;
            end;
      end;
      level := level+1;
   until (exitlev);
   datarec.Bulk_Density := rvalue(parameter[1]);
   datarec.Porosity     := rvalue(parameter[2]);
   datarec.Kd           := rvalue(parameter[3]);
   datarec.Dispersivity := rvalue(parameter[4]);
   datarec.Diffusion    := rvalue(parameter[5]);
   datarec.Timeincrement:= ivalue(parameter[6]);
   datarec.TotalTime    := ivalue(parameter[7]);
end;


Procedure Hydrogeo_Plume_param;
const
   c      : integer=63;
   title  : string[30] = 'Hydrolgeological Parameters';
   title1 : string[30] = 'Pumping Well Characteristics';
   sc     : integer=5;
   sr     : integer=2;
   bw     : integer=70;
   bh     : integer=22;
   pwidth : integer=10;
var
   ch         : char;
   level : integer;
   exitlev : boolean;
   ppos       : integer;
   exit_menu : boolean;
   window     : integer;
   maxwindow : integer;
   x          : integer;
   exit       : boolean;
   parameters: parameterstype;
   diff       : integer;
   maxnumparams    : integer;
   exit_window : boolean;
begin
   clrscr;
   param_box(sr,sc,bw,bh);
   putat((80-length(title)) shr 1, sr+1,title);
   putat(sc+4,sr+5,'Is the aquifer confined or unconfined (C/U)? ');
   putat(sc+4,sr+7,'Do you have leaky artesian conditions (Y/N): ');
   putat(sc+4,sr+9,'Do you have induced infiltration from a stream or pond (Y/N)
? ');
   putat(sc+4,sr+11,'Enter number of pumping wells utilized to capture plume: ')
;
   putat(sc+4,sr+13,'Enter average site hydraulic gradient: ');
   putat(sc+4,sr+15,'Enter average aquifer thickness: ');
   textcolor(0);
   textbackground(1);
   level := 1;
   ppos:= sr+5;
   exitlev := false;
   repeat
      case level of
         1 : begin
                gotoxy(sc+c-14,ppos);
                s := datarec.aquifer;
                len:=length(s);
                lendiff:=1-len;
                inverseon;
                for i:=1 to lendiff do begin
                  write(' ');
                end;
                write(s);
                repeat
                   gotoxy(sc+c-14,ppos);
                   s := getstr_alphanum(1,s);
                until (((length(s)=1) and (upcase(s[1]) in ['C','U'])) or
                (lastchar = #72));
                if lastch = #72 then level := level - 1;
                inverseoff;
                parameter[1] := upcase(s[1]);
             end;
         2 : begin
                gotoxy(sc+50,ppos+2);
                s:=datarec.leakage;
                len:=length(s);
                lendiff:=1-len;
                inverseon;
                for i:=1 to lendiff do begin
                  write(' ');
                end;
                write(s);
                repeat
                   gotoxy(sc+50,ppos+2);
                   s := getstr_alphanum(1,s);
                until (((length(s)=1) and (upcase(s[1]) in ['Y','N'])) or
                      (lastchar = #72));
                if lastch = #72 then level := level-2;
                inverseoff;
                parameter[2] := upcase(s[1]);
             end;
         3 : begin
                gotoxy(sc+66,ppos+4);
                s := datarec.Infiltration;
                len:=length(s);
                lendiff:=1-len;
```

```
       inverseon;
       for i:=1 to lendiff do begin
         write(' ');
       end;
       write(s);
       repeat
          gotoxy(sc+66,ppos+4);
          s := getstr_alphanum(1,s);
       until (((length(s)=1) and (upcase(s[1]) in ['Y','N']))
             or (lastchar = #72));
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[3] := upcase(s[1]);
    end;
4 : begin
       repeat
          gotoxy(sc+c+1,ppos+6);
          inum := datarec.no_pumping_wells;
          s := istringequiv(inum);
          len:=length(s);
          lendiff:=2-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+c+2,ppos+6);
          s := getnum(3,s);
       until (ivalue(s) in [1..10]) or (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[4] := s;
    end;
5 : begin
      repeat
        gotoxy(sc+50,ppos+8);
        rnum := datarec.average_site_gradient;
        s := rstringequiv(rnum,5);
        len:=length(s);
        lendiff:=5-len;
        inverseon;
        for i:=1 to lendiff do begin
          write(' ');
        end;
        write(s);
        gotoxy(sc+50,ppos+8);
        s := getnum(5,s);
        until ((rvalue(s) >= 0.0) and (rvalue(s) <= 99999.0))
               or (lastchar = #72);
        if lastch = #72 then level := level-2;
        inverseoff;
        parameter[5] :=s;
   end;

   6 : begin
```

```
                 repeat
                   gotoxy(sc+50,ppos+10);
                   rnum := datarec.average_aquifer_thickness;
                   s := rstringequiv(rnum,5);
                   len:=length(s);
                   lendiff:=5-len;
                   inverseon;
                   for i:=1 to lendiff do
                      write(' ');
                   write(s);
                   gotoxy(sc+50,ppos+10);
                   s := getnum(5,s);
                 until ((rvalue(s) >= 0.0) and (rvalue(s) <= 99999.0))
                                                  or (lastchar = #72);
                 if lastch = #72 then level := level-2
                                  else exitlev := true;
                 inverseoff;
                 parameter[6] :=s;
            end;
     end;
     level := level+1;
  until (exitlev);
  datarec.aquifer                   := parameter[1];
  datarec.Leakage                   := parameter[2];
  datarec.Infiltration              := parameter[3];
  datarec.no_pumping_wells          := ivalue(parameter[4]);
  datarec.Average_Site_Gradient     := rvalue(parameter[5]);
  datarec.average_aquifer_thickness := rvalue(parameter[6]);
  Case datarec.aquifer[1] of
     'C','c' : datarec.Confined:=true;
     'U','u' : datarec.Confined:=false;
  end;
  Case datarec.Leakage[1] of
     'Y','y' : datarec.Leaky_Artesian_Conditions:=true;
     'N','n' : datarec.Leaky_Artesian_Conditions:=false;
  end;
  Case datarec.Infiltration[1] of
     'Y','y' : datarec.Induced_Infiltration:=true;
     'N','n' : datarec.Induced_Infiltration:=false;
  end;
  clrscr;

  if datarec.no_pumping_wells <> 0 then begin
     exit_window:=  false;
     maxwindow:= datarec.no_pumping_wells;
     window:= 1;
     repeat
        clrscr;
        param_box(sr,sc,bw,bh+2);
        putat((80-length(title1)) shr 1 , sr+1,title1);
        putat(wherex-length(title1), sr+3,'Pumping Well #');
        write(window);
        putat(sc+2,sr+7,'Pumping Rate: ');
        putat(sc+2,sr+9,'Enter Maximum Percent Dewatering of Available Yield: '
);
```

```
        putat(sc+2,sr+11,'Enter x coord for top left corner of search area: ');
        putat(sc+2,sr+13,'Enter y coord for top left corner of search area: ');
        putat(sc+2,sr+15,'Enter x coord for bottom right corner of search area
: ');
        putat(sc+2,sr+17,'Enter y coord for bottom right corner of search area
: ');
        diff:= 2;
        ppos:= sr + 7;
        level := 1;
        exitlev := false;
        repeat
           case level of
              1 : begin
                     repeat
                        gotoxy(sc+21,ppos);
                        rnum := datarec.wellno[window].well_pump_rate;
                        s := rstringequiv(rnum,6);
                        len:=length(s);
                        lendiff:=6-len;
                        inverseon;
                        for i:=1 to lendiff do begin
                          write(' ');
                        end;
                        write(s);
                        gotoxy(sc+21,ppos);
                        s := getnum(6,s);
                     until ((rvalue(s) >= -9999999.99) and (rvalue(s) <= 999999
9.99))
                        or (lastchar = #72);
                     if lastch = #72 then level := level-1;
                     inverseoff;
                     parameter[1] := s;
                  end;
              2 : begin
                     repeat
                        gotoxy(sc+57,ppos+2);
                        rnum := datarec.wellno[window].percent_dewater;
                        s := rstringequiv(rnum,6);
                        len:=length(s);
                        lendiff:=6-len;
                        inverseon;
                        for i:=1 to lendiff do begin
                          write(' ');
                        end;
                        write(s);
                        gotoxy(sc+57,ppos+2);
                        s := getnum(6,s);
                     until ((rvalue(s) >= 0.0) and (rvalue(s) <= 1.0))
                        or (lastchar = #72);
                     if lastch = #72 then level := level-2;
                     inverseoff;
                     parameter[2] := s;
                  end;
              3 : begin
                     repeat
```

```
                          gotoxy(sc+61,ppos+4);
                          inum := datarec.wellno[window].tlx;
                          s := istringequiv(inum);
                          len:=length(s);
                          lendiff:=2-len;
                          inverseon;
                          for i:=1 to lendiff do begin
                            write(' ');
                          end;
                          write(s);
                          gotoxy(sc+62,ppos+4);
                          s := getnum(3,s);
                       until ((rvalue(s) >= 2.0) and (trunc(rvalue(s)) <= datarec
.nc-2))
                          or (lastchar = #72);
                       if lastch = #72 then level := level-2;
                       inverseoff;
                       parameter[3] := s;
                    end;
                4 : begin
                       repeat
                          gotoxy(sc+61,ppos+6);
                          inum := datarec.wellno[window].tly;
                          s := istringequiv(inum);
                          len:=length(s);
                          lendiff:=2-len;
                          inverseon;
                          for i:=1 to lendiff do begin
                            write(' ');
                          end;
                          write(s);
                          gotoxy(sc+62,ppos+6);
                          s := getnum(3,s);
                       until ((rvalue(s) >= 2.0) and (trunc(rvalue(s)) <= datarec
.nr-2))
                          or (lastchar = #72);
                       if lastch = #72 then level := level-2;
                       inverseoff;
                       parameter[4] := s;
                    end;
                5 : begin
                       repeat
                          gotoxy(sc+61,ppos+8);
                          inum := datarec.wellno[window].brx;
                          s := istringequiv(inum);
                          len:=length(s);
                          lendiff:=2-len;
                          inverseon;
                          for i:=1 to lendiff do begin
                            write(' ');
                          end;
                          write(s);
                          gotoxy(sc+62,ppos+8);
                          s := getnum(3,s);
```

```
                      until ((rvalue(s) >= 2.0) and (trunc(rvalue(s)) <= datarec
.nc-2))
                         or (lastchar = #72);
                      if lastch = #72 then level := level-2;
                      inverseoff;
                      parameter[5] := s;
                   end;
               6 : begin
                      repeat
                         gotoxy(sc+61,ppos+10);
                         inum := datarec.wellno[window].bry;
                         s := istringequiv(inum);
                         len:=length(s);
                         lendiff:=2-len;
                         inverseon;
                         for i:=1 to lendiff do begin
                           write(' ');
                         end;
                         write(s);
                         gotoxy(sc+62,ppos+10);
                         s := getnum(3,s);
                      until ((rvalue(s) >= 2.0) and (trunc(rvalue(s)) <= datarec
.nr-2))
                         or (lastchar = #72);
                      if lastch = #72 then level := level-2
                                          else exitlev := true;
                      inverseoff;
                      parameter[6] := s;
                   end;
            end;
            level := level+1;
         until (exitlev);
         with datarec.wellno[window]  do begin
            well_pump_rate:= rvalue(parameter[1]);
            percent_dewater := rvalue(parameter[2]);
            tlx:= ivalue(parameter[3]);
            tly:= ivalue(parameter[4]);
            brx:= ivalue(parameter[5]);
            bry:= ivalue(parameter[6]);
            trx := brx;
            try := tly;
            blx := tlx;
            bly := bry;
         end;
         window:= window + 1;
         if window > maxwindow then exit_window := true;
      until exit_window;
   end;
end;


Procedure Finite_Param;
Const
   Pwidth : integer =10;
   bh     : integer=22;
   bw     : integer=72;
   sr     : integer=2;
   sc     : integer=5;
   title  : string[30]='Finite Difference Parameters';
   c      : integer=63;
var
   ch           : char;
   exit         : boolean;
   ppos         : integer;
   maxnumparams : integer;
   diff         : integer;
   x            : integer;
   exit_windows : boolean;
   maxwindow    : integer;
   window       : integer;
   exit_menu    : boolean;
   level        : integer;
   exitlev      : boolean;
begin
   clrscr;
   param_box(sr,sc,bw,bh);
   sr := sr + 1;
   putat((80-length(title))shr 1,sr+1,title);
   putat(sc+4,sr+3,'Enter total number of nodes in x-direction: ');
   putat(sc+4,sr+5,'Enter total number of nodes in y-direction: ');
   putat(sc+4,sr+7,'Enter number of time steps in simulation: ');
   putat(sc+4,sr+9,'Enter initial time increment (days): ');
   putat(sc+4,sr+11,'Enter tolerance for F.D. convergence criteria (1-0.001): ')
;
   putat(sc+4,sr+13,'Enter anisotropy factor (ty/tx): ');
   putat(sc+4,sr+15,'Enter distance of grid spacing: ');
   textcolor(0);
   textbackground(1);
   exit_menu:=false;
   exit:=false;
   diff:= 2;
   ppos:= sr+3;
   level := 1;
   exitlev := false;
   repeat
      case level of
         1 : begin
                repeat
                   gotoxy(sc+48,ppos);
                   inum := datarec.nc;
                   s := istringequiv(inum);
                   len:=length(s);
                   lendiff:=2-len;
                   inverseon;
                   for i:=1 to lendiff do begin
                     write(' ');
                   end;
                   write(s);
                   gotoxy(sc+48,ppos);
                   s := getnum(2,s);
```

```
     until (ivalue(s) in [1..50]) or (lastchar = #72);
     if lastch = #72 then level := level-1;
     inverseoff;
     parameter[1] := s;
   end;
2 : begin
     repeat
       gotoxy(sc+48,ppos+2);
       inum := datarec.nr;
       s := istringequiv(inum);
       len:=length(s);
       lendiff:=2-len;
       inverseon;
       for i:=1 to lendiff do begin
         write(' ');
       end;
       write(s);
       gotoxy(sc+48,ppos+2);
       s := getnum(2,s);
     until (ivalue(s) in [1..50]) or (lastchar = #72);
     if lastch = #72 then level := level-2;
     inverseoff;
     parameter[2] := s;
   end;
3 : begin
     repeat
       gotoxy(sc+46,ppos+4);
       inum := datarec.nsteps;
       s := istringequiv(inum);
       len:=length(s);
       lendiff:=4-len;
       inverseon;
       for i:=1 to lendiff do begin
         write(' ');
       end;
       write(s);
       gotoxy(sc+46,ppos+4);
       s := getnum(4,s);
     until ((ivalue(s)>=1) and (ivalue(s)<=999999))
       or (lastchar = #72);
     if lastch = #72 then level := level-2;
     inverseoff;
     parameter[3] := s;
   end;
4 : begin
     repeat
       gotoxy(sc+41,ppos+6);
       rnum := datarec.deita;
       s := rstringequiv(rnum,6);
       len:=length(s);
       lendiff:=6-len;
       inverseon;
       for i:=1 to lendiff do begin
         write(' ');
       end;
```

```
          write(s);
          gotoxy(sc+41,ppos+6);
          s := getnum(6,s);
       until ((rvalue(s) >=0.0) and (rvalue(s)<=99999.9)) or
             (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[4] := s;
    end;
5 : begin
       repeat
          gotoxy(sc+60,ppos+8);
          rnum :=datarec.error;
          s := rstringequiv(rnum,6);
          len:=length(s);
          lendiff:=6-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+60,ppos+8);
          s := getnum(6,s);
       until ((rvalue(s))=0.0001) and (rvalue(s)<=1))
          or (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[5] := s;
    end;
6 : begin
       repeat
          gotoxy(sc+37,ppos+10);
          rnum :=datarec.anfctr;
          s := rstringequiv(rnum,6);
          len:=length(s);
          lendiff:=6-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+37,ppos+10);
          s := getnum(6,s);
       until ((rvalue(s))=0.1) and (rvalue(s)<=1.0))
          or (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[6] := s;
    end;
7 : begin
       repeat
          gotoxy(sc+36,ppos+12);
          rnum := datarec.delxy;
          s := rstringequiv(rnum,6);
          len:=length(s);
```

```
                   lendiff:=6-len;
                   inverseon;
                   for i:=1 to lendiff do begin
                     write(' ');
                   end;
                   write(s);
                   gotoxy(sc+36,ppos+12);
                   s := getnum(6,s);
                until ((rvalue(s))>=0.0) and (rvalue(s)<=999999))
                   or (lastchar = #72);
                if lastch = #72 then level := level-2
                                else exitlev := true;
                inverseoff;
                parameter[7] := s;
             end;
      end;
      level := level+1;
   until (exitlev);
   datarec.nc := ivalue(parameter[1]);
   datarec.nr := ivalue(parameter[2]);
   datarec.nsteps := ivalue(parameter[3]);
   datarec.delta  := rvalue(parameter[4]);
   datarec.error  := rvalue(parameter[5]);
   datarec.anfctr := rvalue(parameter[6]);
   datarec.delxy  := rvalue(parameter[7]);
   if datarec.anfctr = 1.0 then anisotropic := false
                   else anisotropic := true;
end;


Procedure Plume_control_menu;
Var
   Exit_Plume_control_menu:boolean;
   sel : integer;
Begin
   Exit_Plume_control_menu:=false;
   Mainmenu:=false;
   Repeat
      inverseoff;
      ClrScr;
      Sr:=2;
      Sc:=7;
      Bw:=69;
      Bh:=24;
      Box(sr,sc,bw,bh);
      Datarec.Plumeprob:=true;
      Options[1]:='Input Parameters for Finite Difference Approximation      ';
      Options[2]:='Input Hydrogeologic Parameters for Plume Investigation    ';
      Options[3]:='Input Parameters for Contaminant Arrival Concentrations   ';
      Options[4]:='Create Characterizing Arrays Used in F.D. Approximations ';
      Options[5]:='Input Plume Characterizing Parameters                     ';
      Options[6]:='Begin Optimum Well Configuration Search for Plume Capture';
      Options[7]:='Solution Summary                                          ';
      Titlestr:='PLUME CONTROL MENU';
      Sel:=menu_bar(titlestr,options,7);
      New(Contaminant_Concentration);
      For i := 1 to matrixsize do
         for j := 1 to matrixsize do
            Contaminant_concentration^[i,j] := 0.0;
      case sel of
         1 : finite_param;
         2 : hydrogeo_plume_param;
         3 : contaminant_param;
         4 : Begin
               spread_sheet(datarec.nr,datarec.nc,sfilename+'.H','Water Head Ele
vation Array');
               spread_sheet(datarec.nr,datarec.nc,sfilename+'.T','Transmisivity
Array');
               spread_sheet(datarec.nr,datarec.nc,sfilename+'.S','Storativity Ar
ray');
               spread_sheet(datarec.nr,datarec.nc,sfilename+'.L','Leakance Array
');
               spread_sheet(datarec.nr,datarec.nc,sfilename+'.CON','Contaminant
Concentration Array');
               if datarec.leaky_artesian_conditions then begin
                 New(rlac);
                 New(aqtdparm);
                 New(aqtdthickness);
                 New(source_unit_head);
                 New(ql);
                 For i := 1 to matrixsize do begin
                   For j := 1 to matrixsize do begin
                     rlac^[i,j]:=0.0;
                     aqtdperm^[i,j]:=0.0;
                     aqtdthickness^[i,j]:=0.0;
                     source_unit_head^[i,j]:=0.0;
                     ql^[i,j]:=0.0
                   end;
                 end;
                 spread_sheet(datarec.nr,datarec.nc,sfilename+'.ADK','Aquitard H
ydraulic Conductivity Array');
                 spread_sheet(datarec.nr,datarec.nc,sfilename+'.ADS','Aquitard T
hickness Array');
                 spread_sheet(datarec.nr,datarec.nc,sfilename+'.SUH','Source Uni
t Head');
                end;
                if not datarec.confined then begin
                   New(aqfrperm);
                   New(aquiferbase);
                   For i := 1 to matrixsize do begin
                     For j := 1 to matrixsize do begin
                       aqfrperm^[i,j]:=0.0;
                       aquiferbase^[i,j]:=0.0;
                     end;
                   end;
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.AQK','Aquifer
Hydraulic Conductivity Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.AQB','Aquifer
Base Array');
```

```
                end;
                if datarec.induced_infiltration then begin
                   New(iir);
                   New(qi);
                   New(waterbodysurfelev);
                   New(Waterbodybedperm);
                   New(Waterbodybedthk);
                   New(Waterbodyarea);
                   New(Waterbodybedelev);
                   For i := 1 to matrixsize do begin
                     For j := 1 to matrixsize do begin
                       iir^[i,j]:=0.0;
                       qi^[i,j]:=0.0;
                       waterbodysurfelev^[i,j]:=0.0;
                       waterbodybedperm^[i,j]:=0.0;
                       waterbodybedthk^[i,j]:=0.0;
                       waterbodyarea^[i,j]:=0.0;
                       waterbodybedelev^[i,j]:=0.0;
                     end;
                   end;
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBK','Water Bo
dy Bed Hydraulic Conductivity Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBS','Water Bo
dy Bed Thickness Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBA','Water Bo
dy Area Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBB','Water Bo
dy Base Elevation');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBE','Water Bo
dy Surface Elevation Array');
                end;
                LOAD_SPREADS;

                For i := 1 to matrixsize do
                   For j := 1 to matrixsize do
                       Hprime^[i,j]:=H^[i,j];
             end;
         5 : Plume_Characteristics;
         6 : begin
                load_spreads;
                Draw_Search_Grid(datarec.nr,datarec.nc);
             end;
         7 : Plume_Solution_Summary;
         else Exit_Plume_control_menu:=true;
      end;
   Until exit_plume_control_menu;
End;


Procedure Dewatering_menu;
Var
   Exit_Dewatering_menu : boolean;
   sel : integer;
Begin
```

```
   Exit_Dewatering_menu:=false;
   mainmenu:=false;
   datarec.Plumeprob:=false;
   datarec.dewateringprob:=true;
   Repeat
      Inverseoff;
      ClrScr;
      Sr:=4;
      Sc:=7;
      Bw:=69;
      Bh:=21;
      Box(sr,sc,bw,bh);
      Options[1]:='Input Parameters for Finite Difference Approximations      ';
      Options[2]:='Input Hydrogeologic Parameters for Dewatering Investigation';
      Options[3]:='Create Characterizing Arrays used in F.D. Approximations   ';
      Options[4]:='Begin Optimum Well Configuration search                    ';
      Options[5]:='Solution Summary                                           ';
      Titlestr:='DEWATERING MENU' ;
      Sel:=Menu_bar(titlestr,options,5);
      case sel of
         1 : finite_param;
         2 : hydrogeological_param;
         3 : begin
                spread_sheet(datarec.nr,datarec.nc,sfilename+'.H','Water Head El
evation Array');
                spread_sheet(datarec.nr,datarec.nc,sfilename+'.T','Transmisivity
 Array');
                spread_sheet(datarec.nr,datarec.nc,sfilename+'.S','Storativity A
rray');
                spread_sheet(datarec.nr,datarec.nc,sfilename+'.L','Leakance Arra
y');
                if datarec.leaky_artesian_conditions then begin
                   New(rlac);
                   New(aqtdperm);
                   New(aqtdthickness);
                   New(source_unit_head);
                   New(ql);
                   For i := 1 to matrixsize do begin
                     For j := 1 to matrixsize do begin
                       rlac^[i,j]:=0.0;
                       aqtdperm^[i,j]:=0.0;
                       aqtdthickness^[i,j]:=0.0;
                       source_unit_head^[i,j]:=0.0;
                       ql^[i,j]:=0.0
                     end;
                   end;
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.ADK','Aquitard
 Hydraulic Conductivity Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.ADS','Aquitard
 Thickness Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.SUH','Source U
nit Head');
                end;
                if not datarec.confined then begin
                   New(aqfrperm);
                   New(aquiferbase);
                   For i := 1 to matrixsize do begin
                     For j := 1 to matrixsize do begin
                       aqfrperm^[i,j]:=0.0;
                       aquiferbase^[i,j]:=0.0;
                     end;
                   end;
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.AQK','Aquifer
Hydraulic Conductivity Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.AQB','Aquifer
Base Array');
                end;
                if datarec.induced_infiltration then begin
                   New(iir);
                   New(qi);
                   New(waterbodysurfelev);
                   New(Waterbodybedperm);
                   New(Waterbodybedthk);
                   New(Waterbodyarea);
                   New(Waterbodybedelev);
                   For i := 1 to matrixsize do begin
                     For j := 1 to matrixsize do begin
                       iir^[i,j]:=0.0;
                       qi^[i,j]:=0.0;
                       waterbodysurfelev^[i,j]:=0.0;
                       waterbodybedperm^[i,j]:=0.0;
                       waterbodybedthk^[i,j]:=0.0;
                       waterbodyarea^[i,j]:=0.0;
                       waterbodybedelev^[i,j]:=0.0;
                     end;
                   end;
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBK','Water Bo
dy Bed Hydraulic Conductivity Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBS','Water Bo
dy Bed Thickness Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBA','Water Bo
dy Area Array');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBB','Water Bo
dy Base Elevation');
                   spread_sheet(datarec.nr,datarec.nc,sfilename+'.WBE','Water Bo
dy Surface Elevation Array');
                end;
                LOAD_SPREADS;

                For i := 1 to matrixsize do
                   For j := 1 to matrixsize do
                       Hprime^[i,j]:=H^[i,j];

             end;
         4 : draw_search_grid(datarec.nr,datarec.nc);
         5 : dewatering_solution_summary;
         else Exit_Dewatering_menu:=true;
      end;
   Until exit_dewatering_menu;
End;
```

```
procedure initialize_load_input_parameters;
{load from sfilename.DAT}
begin
   data_created := true;
end;



Procedure Initialization_menu;
Const
   sc : integer = 2;
   sr : integer = 3;
   bw : integer = 75;
   bh : integer = 23;
   c  : integer = 63;
Var
   ch   : char;
   x    : integer;
   ppos : integer;
   status : integer;
Begin
   maxnumparams := 2;
   ClrScr;
   Init_Box(sc,sr,bw,bh);
   putat(sc+5,sr,'1) Enter Project Name (Maximum 8 Characters): ');
   putat(sc+5,sr+2,
      '2) Select Unit System that you would like to use throughout ');
   putat(sc+8,sr+3,'this program (M-Metric, I-Imperial, U-United States):');
   putat(sc+30,sr+7,'* Units *');
   putat(sc+28,sr+9,'Metric          Imperial       United States');
   putat(sc+3,sr+10,
     'Hydraulic Conductivity    m/day       gal/day/sq. ft    gal/day/sq. ft');
   putat(sc+3,sr+11,
      'Withdrawal Rate         Cu. m/day       gal/day             gal/day ');
   putat(sc+3,sr+12,
      'Transmissivity          sq. m/day      gal/day/ft          gal/day/ft');
   putat(sc+3,sr+13,
      'Distance                    m              ft                  ft');
   putat(sc+3,sr+14,
      'Time                       day            day                 day');
   putat(sc+3,sr+15,
      'Recharge                sq. m/day      gal/day/ft           gal/day/ft');
   textcolor (0);
   textbackground (1);
   ppos := sr;
   gotoxy(sc+c-11,ppos);
   s := 'Filename ';
   inverseon;
   write(s);
   gotoxy(sc+c-11,ppos);
   s := getstr_alphanum(8,s);
   inverseoff;
```

```
name := s;
sfilename := s;
assign(datafile,sfiluname+'.DAT');
{$I-}
reset(datafile);
{$I+}
status := ioresult;
if status <> 0 then
  begin
    rewrite(datafile);
    With datarec do begin
      units                                     :=  'M';
      plumeprob                                 :=false;
      dewateringprob                            :=false;
      nc                                        :=   50;
      nr                                        :=   50;
      nsteps                                    := 3650;
      delta                                     :=    2;
      error                                     := 0.01;
      anfctr                                    :=  1.0;
      delxy                                     :=   50;
      aquifer                                   :=  'C';
      confined                                  := true;
      leakage                                   :=   'N';
      Leaky_artesian_Conditions                 := false;
      Infiltration                              :=   'N';
      Induced_Infiltration                      := false;
      no_pumping_wells                          :=   10;
      For i:=1 to no_pumping_wells do begin
        With wellno[i] do begin
          well_pump_rate                        :=   500;
          percent_dewater                       :=  0.25;
          tlx                                   :=   i*2;
          tly                                   :=   i*2;
          brx                                   := i*2+1;
          bry                                   := i*2+1;
          trx                                   :=   brx;
          try                                   :=   tly;
          blx                                   :=   tlx;
          bly                                   :=   bry;
        end;    {with}
      end;    {for}
        no_dewatering_cntrl_points                :=   10;
        For i:=1 to no_dewatering_cntrl_points  do begin
          With control_Point[i] do begin
            cpxcoord                              :=   i+2;
            cpycoord                              :=   i+2;
            cphead                                :=   150;
            cpH_range                             :=     5;
          end;   {with}
        end;   {i}
        Average_site_gradient                   :=   2.5;
        Average_aquifer_thickness               :=  30.0;
        Bulk_density                            :=   2.3;
        Porosity                                :=  0.30;
```

```
          Kd                                      :=  0.00;
          Dispersivity                            :=  0.10;
          Diffusion                               := 0.0000023;
          Timeincrement                           :=     2;
          Totaltime                               :=  3650;
          nplume                                  :=    10;
          For i:=1 to nplume do begin
            With plume[i] do begin
              ptlx                                :=   i*2;
              ptly                                :=   i*2;
              pbrx                                := i*2+1;
              pbry                                := i*2+1;
              ptrx                                :=   pbrx;
              ptry                                :=   ptly;
              pblx                                :=   ptlx;
              pbly                                :=   pbry;
            end;    {with}
          end;    {for}
      end;    {with Datarec}
    end;    {if status}
  if status = 0 then
    begin
      read(datafile,datarec);
      load_spreads;
    end;
  datafopen := true;
  gotoxy(sc+c-1,ppos+3);
  inverseon;
  s := datarec.units;
  write(s);
  repeat
     gotoxy(sc+c-1,ppos+3);
     s := getstr_alphanum(1,s);
  until ((length(s)=1) and (upcase(s[1]) in ['M','I','U']));
  inverseoff;
  datarec.units := s;
  if sfilename<>'' then initialize_load_input_parameters;
  close(datafile);
  datafopen := false;
End;


Procedure Quit;
Var
   answer  : char;
Begin
   clrscr;
   if data_created then begin
      repeat
         clrscr;
         putat(12,10,'Would you like the updated data file for '
              +sfilename+ ' saved (y/n)? ');
         answer:= readkey;
      until(upcase(answer) in ['Y','N']);
      clrscr;
      case upcase(answer) of
         'Y' : begin
                  putat(21,12,'Saving data file');
                  save_data_file;
                  clrscr;
               end;
         'N' : clrscr;
      end;
   end;
   Repeat
      ClrScr;
      Putat(21,12,'Do really want to quit (Y/N)?');
      Answer:=readkey;
   Until (upcase(Answer) in ['Y','N']);
   Case upcase(Answer) of
      'Y' : Begin
               ClrScr;
               exit_main_menu := true;
            End;
      'N' : begin
               Clrscr;
               exit_main_menu := false;
            end;
   End;
End;


procedure MustHaveSname;
var
   wx,wy : integer;
   ch : char;
begin
   wx := wherex;
   wy := wherey;
   s := 'Sorry!  You must first LOAD/INITIALIZE DATA';
   putat((84-length(s)) div 2,24,s);
   delay(2000);
   gotoxy(1,wherey);
   clreol;
   while keypressed do ch := readkey;
   gotoxy(wx,wy);
end;


Procedure Main_menu;
var sel : integer;
Begin
   exit_main_menu := false;
   Mainmenu:=true;
   Repeat
      inverseoff;
      Clrscr;
      sr:=3;
      sc:=18;
      bw:=48;
      bh:=20;
      Box (sr,sc,bw,bh);
      Options[1]:= 'Load/Initialize Data';
      Options[2]:= 'Dewatering          ';
      Options[3]:= 'Plume Control       ';
      Options[4]:= 'Quit                ';
      Titlestr :=  'MAIN MENU';
      sel := Menu_bar(titlestr,options,4);
      case sel of
         1 : initialization_menu;
         2 : if sfilename<>'' then dewatering_menu
                              else musthavesname;
         3 : if sfilename<>'' then Plume_control_menu
                              else musthavesname;
         4 : quit;
         else
             quit;
      end;
   until exit_main_menu;
End;


Procedure Title_Screen;
Const
   Intro  : string[70] =
           '* Optimum Well Management for Plume Control and Dewatering *';
   Credit : string[70] = 'by Robert J. Hyde                (copyright 1990)';
   Instr  : string[70] = 'Press Any Key To Continue';
   P      : string[5]  = 'P';
Type
   Array_Type = Array[1..10,1..2] of integer;
Var
   A,B,C,D,E,F : Array_Type;
   Ch          : Char;
Begin
   ClrScr;
   Initgraph(gd,gm,'c:\tp');
   delay(500);
   SetFillStyle(8,1);
   SetLinestyle(0,0,1);
   A[1,1]:=30;  A[1,2]:=50;  A[2,1]:=30;  A[2,2]:=100;
   A[3,1]:=275; A[3,2]:=100; A[4,1]:=275; A[4,2]:=50;
   B[1,1]:=295; B[1,2]:=50;  B[2,1]:=295; B[2,2]:=100;
   B[3,1]:=600; B[3,2]:=100; B[4,1]:=600; B[4,2]:=50;
   C[1,1]:=30;  C[1,2]:=100; C[2,1]:=30;  C[2,2]:=160;
   C[3,1]:=275; C[3,2]:=160; C[4,1]:=275; C[4,2]:=100;
   D[1,1]:=295; D[1,2]:=100; D[2,1]:=295; D[2,2]:=160;
   D[3,1]:=600; D[3,2]:=160; D[4,1]:=600; D[4,2]:=100;
   E[1,1]:=295; E[1,2]:=128; E[2,1]:=500; E[2,2]:=125;
   E[3,1]:=350; E[3,2]:=134; E[4,1]:=580; E[4,2]:=145;
   E[5,1]:=420; E[5,2]:=150; E[6,1]:=500; E[6,2]:=155;
```

```
    E[7,1]:=295; E[7,2]:=140;
    F[1,1]:=275; F[1,2]:=128; F[2,1]:=50;  F[2,2]:=118;
    F[3,1]:=220; F[3,2]:=135; F[4,1]:=100; F[4,2]:=145;
    F[5,1]:=150; F[5,2]:=150; F[6,1]:=70;  F[6,2]:=155;
    F[7,1]:=275; F[7,2]:=140;
    DrawPoly(4,A); FillPoly(4,A); DrawPoly(4,B); FillPoly(4,B);
    SetFillStyle(10,1);
    DrawPoly(4,C); FillPoly(4,C); DrawPoly(4,D); FillPoly(4,D);
    Line(275,35,275,100);
    Line(275,35,295,35);
    Line(295,35,295,100);
    SetLineStyle(1,0,1);
    Line(275,100,275,160);
    Line(295,100,295,160);
    SetLineStyle(0,0,1);
    Line(275,160,295,160);
    SetTextStyle(1,0,1);
    OutTextXY(20,160,Intro);
    SetTextStyle(0,0,1);
    OutTextxy(230,180,Credit);
    OutTextxy(200,188,Instr);
    SetFillStyle(1,1);
    DrawPoly(7,E);
    FillPoly(7,E);
    DrawPoly(7,F);
    FillPoly(7,F);
    Line(280,35,280,10);
    Line(290,35,290,15);
    Line(280,10,500,10);
    Line(290,15,500,15);
    Circle(530,12,30);
    SetTextStyle(1,0,1);
    OutTextxy(526,3,P);
    Line(560,10,590,10);
    Line(560,15,590,15);
    while keypressed do ch := readkey;
    ch:=readkey;
    RestoreCrtMode;
    delay(900);
End;


Begin
  clrscr;
  Title_screen;
  Main_Menu;
  clrscr;
End.
```

```
unit globals;
{$E+}
{$N+}

interface
uses dos,crt,graph;

Const
   Maxnumoptions =  10;
   matrixsize    =  10;
   itmin         =   5;
   itmax         = 500;
   gd            : integer =   0;
   gm            : integer =   2;

Type
  string80        = string [80];
  integertype     = integer;
  Titlestrtype    = string [80];
  param_type      = string [10];
  Windowtype      = array[1..10000] of Byte;
  Windowptr       = ^Windowtype;
  picturepntrtype= array[1..5000] of byte;
  picturepntrtype2= array[1..20000] of byte;
  picturepntr     = array[1..10] of ^picturepntrtype;
  matrix          = array [1..matrixsize,1..matrixsize] of real;
  parameterstype  = array [1..10] of param_type;
  optionstype     = array [1..maxnumoptions] of string [60];
  matrixtype      = ^matrix;
  datarectype     = record
                     units                        : string80;
                     plumeprob                    : boolean;
                     dewateringprob               : boolean;
                     nc,nr                        : integer;
                     nsteps                       : integer;
                     delta                        : real;
                     maxconc                      : real;
                     error                        : real;
                     anfctr                       : real;
                     delxy                        : real;
                     aquifer                      : string [80];
                     confined                     : boolean;
                     Leakage                      : string80;
                     Leaky_Artesian_Conditions    : boolean;
                     Infiltration                 : string80;
                     Induced_Infiltration         : boolean;
                     no_pumping_wells             : integer;
                     wellno       : array [1..10] of record
                                     x                : integer;
                                     y                : integer;
                                     xrad             : real;
                                     yrad             : real;
                                     xradius          : real;
                                     yradius          : real;
                                     maxxrad          : real;
```

```
                                      maxyrad         : real;
                                      maxxl           : integer;
                                      maxyl           : integer;
                                      conc            : real;
                                      concx           : integer;
                                      concy           : integer;
                                      bestdist        : real;
                                      time            : array [1..100] of real;
                                      concentration   : array [1..100] of real;
                                      tlx             : integer;
                                      tly             : integer;
                                      brx             : integer;
                                      bry             : integer;
                                      trx             : integer;
                                      try             : integer;
                                      blx             : integer;
                                      bly             : integer;
                                      xnlines         : integer;
                                      ynlines         : integer;
                                      well_dot        : array [1..5,1..5] of int
eger;

                                      search_coord    : array [1..5,1..5] of int
eger;

                                      well_pump_rate  : real;
                                      percent_dewater : real;
                                    end;   {well_no record}
                    no_dewatering_cntrl_points  : integertype;
                    control_point : array [1..10] of record
                                      cpxcoord  : integer;
                                      cpycoord  : integer;
                                      cphead    : real;
                                      cpH_range :real;
                                    end;    {Control_point record}
                     Bulk_density                : real;
                     Average_site_gradient       : real;
                     Average_aquifer_thickness   : real;
                     porosity                    : real;
                     Kd                          : real;
                     Dispersivity                : real;
                     Diffusion                   : real;
                     Timeincrement               : integer;
                     Totaltime                   : integer;
                     nplume                      : integer;
                     plume         : array [1..10] of record
                                      ptlx      : integer;
                                      ptly      : integer;
                                      pbrx      : integer;
                                      pbry      : integer;
                                      ptrx      : integer;
                                      ptry      : integer;
                                      pblx      : integer;
                                      pbly      : integer;
                                    end;    {plume record}

                   end;  {record}
```

```
var
   datarec                                   : datarectype;
   datafile                                  : file of datarectype;
   datafopen                                 : boolean;
   s                                         : string80;
   storagefctr                               : real;
   wellpntr                                  : ^picturepntrtype2;
   elipsepntr                                : picturepntr;
   inum                                      : integer;
   lnum                                      : integer;
   rnum                                      : real;
   len                                       : integer;
   lendiff                                   : integer;
   R                                         : integer;
   xasp                                      : word;
   yasp                                      : word;
   maxnumparams                              : integer;
   mainmenu                                  : boolean;
   titlestr                                  : titlestrtype;
   options                                   : optionstype;
   numoptions                                : integer;
   sel                                       : integer;
   sfilename                                 : string80;
   sr,sc,bw,bh                               : integer;
   escstatus                                 : integer;
   pwidth                                    : integer;
   lastchar                                  : char;
   lm                                        : word;
   exit_main_menu                            : Boolean;
   exit_search                               : boolean;
   ch                                        : char;
   parameter                                 : parameterstype;
   paramtext                                 : array [1..10] of boolean;
   window                                    : integer;
   maxwindow                                 : integer;
   anisotropic                               : boolean;
   well_dry                                  : boolean;
   max_no_wells                              : integer;
   data_created                              : boolean;
   project_name                              : string80;
   vspace                                    : real;
   hspace                                    : real;
   xc,yc                                     : array [1..10] of integer;
   xrad,yrad,xcenter,ycenter                 : integer;
   i,j                                       : integer;
   Txavg_up                                  : real;
   Txavg_down                                : real;
   Tyavg_right                               : real;
   Tyavg_left                                : real;
   Txavg                                     : real;
   Tyavg                                     : real;
   Kavg                                      : real;
   Rx,Ry                                     : real;
   Amount_dewatered                          : real;
   xr,yr                                       : integer;
   reg                                         : registers;
   name                                        : string80;
   x,y                                         : array[1..50] of real;

   rlac                                        : matrixtype;
   Contaminant_Concentration                   : matrixtype;
   aqtdperm                                    : matrixtype;
   aqtdthickness                               : matrixtype;
   iir                                         : matrixtype;
   source_unit_head                            : matrixtype;
   ql                                          : matrixtype;
   aqfrperm                                    : matrixtype;
   qi                                          : matrixtype;
   waterbodysurfelev                           : matrixtype;
   waterbodybedperm                            : matrixtype;
   waterbodybedthk                             : matrixtype;
   waterbodyarea                               : matrixtype;
   waterbodybedelev                            : matrixtype;
   aquiferbase                                 : matrixtype;
   LHd                                         : matrixtype;
   H                                           : matrixtype;
   Ho                                          : matrixtype;
   Q                                           : matrixtype;
   Stor                                        : matrixtype;
   T                                           : matrixtype;
   Tx                                          : matrixtype;
   Ty                                          : matrixtype;
   Leakance                                    : matrixtype;
   Hinitial                                    : matrixtype;
   Hprime                                      : matrixtype;

procedure Cursor_off;
procedure Cursor_on;
Function Getnum(cwidth:Integer; s:string80):string80;
procedure Save_data_file;
procedure Reopen_data_file;
function lastch : char;
function getstr_alphanum(n:integer;s:string80) : string80;
procedure Inverseoff;
procedure Inverseon;
function IValue(s:string80) : integer;
function RValue(s:string80) : real;
procedure Putat (c,r : integer; s : string80);
Procedure Param_Box (sr,sc,bw,bh : integer);
Procedure Box (sr,sc,bw,bh : integer);
Procedure Message_Box (sr,sc,bw,bh : integer);
Procedure Init_Box (sr,sc,bw,bh : integer);
Function Menu_bar (Title:titlestrtype; Opts:optionstype;
                   Numopts:integer) : Integer;
Function istringequiv(inum:integer) : string80;
function rstringequiv(rnum:real;lnum:integur) : string80;
```

```
implementation

var xx,yy : integer;


procedure Save_data_file;
begin
   if not datafopen then begin
      assign(datafile,sfilename+'.DAT');
      reset(datafile);
   end;
   write(datafile,datarec);
   close(datafile);
end;


procedure Reopen_data_file;
begin
   if datafopen then
      seek(datafile,0)
   else begin
      assign(datafile,sfilename+'.DAT');
      reset(datafile);
   end;
   read(datafile,datarec);
   close(datafile);
end;

procedure Cursor_off;
begin
   with reg do begin
      ah := 1;
      ch := 32;
   end;
   intr($10,reg);
end;


procedure Cursor_on;
begin
   with reg do begin
      ah:=1;
      if lm = mono then begin
                           ch := 11;
                           cl := 12;
                        end else
                        begin
                           ch := 6;
                           cl := 7;
                        end;
   end;
   intr($10,reg);
end;
```

```
Function Getnum(cwidth:Integer; s:string80):string80;
Var
   x : integer;
   Exit:Boolean;
   Wordx:Integer;
   Word:string80;
   dot,dash,expon,negexp,numb : boolean;
   ch:char;
Begin
   Exit:=False;
   dot := false;
   dash := false;
   expon := false;
   negexp := false;
   numb := false;
   if length(s)>0 then begin
      numb := true;
      dash := true;
   end;
   for x:= 1 to length(s) do begin
      if s[x] = '.' then dot := true;
      if s[x] = 'E' then expon := true;
      if (s[x]='-') and (x>1) and (s[x-1]='E') then negexp := true
                                               else
         if (s[x]='-') then dash := true;
   end;
   Wordx:=Length(s);
   Word:=s;
   gotoxy(wherex+wordx,wherey);
   Repeat
     Ch:=readkey;
     Case ch of
        '.' :  if not dot then begin
                                  Wordx:=wordx+1;
                                  If wordx>cwidth then
                                     wordx:=cwidth
                                  Else
                                  Begin
                                     Write(Ch);
                                     Word[wordx]:=Ch;
                                     dot := true;
                                     dash := true;
                                  End;  {if}
                               End; {begin}
    'e','E' :  if numb and (not expon) then begin
                                               Wordx:=wordx+1;
                                               If wordx>cwidth then
                                                  wordx:=cwidth
                                               Else
                                               Begin
                                                  Write(Ch);
                                                  Word[wordx]:=upcase(Ch);
                                                  expon := true;
                                                  dot := true;
```

```
                                             dash := true;
                                          End;  {if}
                                       End; {begin}
   '-' : if (not dash) or ((not negexp) and (word[wordx]='E') and numb)
         then Begin
                 Wordx:=wordx+1;
                 If wordx>cwidth then
                                     wordx:=cwidth
                                  Else Begin
                                     Write(Ch);
                                     Word[wordx]:=Ch;
                                  if (numb and (word[wordx-1]='E')) then
                                    begin
                                       negexp := true;
                                       dash := true;
                                    end else
                                       dash := true;
                                   End;  {if}
              End; {begin}
'0'..'9': Begin
             Wordx:=wordx+1;
             If wordx>cwidth then
                wordx:=cwidth
             Else Begin
                Write(Ch);
                Word[wordx]:=Ch;
                dash := true;
                numb := true;
             End;  {if}
          End; {begin}
#8:Begin
      Wordx:=wordx-1;
      If wordx<0 then begin
         Wordx:=0;
         dot := false;
         dash := false;
         numb := false;
         expon := false;
         negexp := false;
      end Else begin
         Write(#8,' ',#8);
         if word[wordx+1] = '.' then dot := false;
         if word[wordx+1] = 'E' then expon := false;
         if (word[wordx+1] = '-') and (word[wordx]<>'E') then
            dash := false
         else if (word[wordx+1]='-') then
            negexp := false;
         Word[wordx+1]:=' ';
         if wordx = 0 then begin
            dot := false;
            dash := false;
            expon := false;
            negexp := false;
            numb := false;
         end;
```

```
            End;    {If}
         End;    {begin}
   #13 : Begin
            LastChar:=#13;
            Exit:=True;
            word[0]:=Chr(wordx);
            Gotoxy(wherex-wordx,wherey);
         End;   {Begin}
    #0 : Begin
            ch:=readkey;
            Case ch of
               #59,#75,#77,#72,#80,#71:Begin
                                          Gotoxy(wherex-wordx,wherey);
                                          Lastchar:=Ch;
                                          Exit:=True;
                                          Word[0]:=chr(wordx);
                                       End;   {begin}
               #60 : begin
                        save_data_file;
                        reopen_data_file;
                     end;
            End;  {Case}
         End;  {Begin}
      End;   {Case}
   Until Exit;
   Getnum:=Word;
End; {Function}


function lastch : char;
begin
   lastch := lastchar;
   lastchar := #0;
end;


function getstr_alphanum(n:integer; s:string80) : string80;
var x  : integer;
    ch : char;
    w  : string80;
    L  ; integer;
begin
   x := length(s);
   ch := #0;
   w := s;
   L:=x;
   gotoxy(wherex+L,wherey);
   while (ch<>#13) do begin
      ch := readkey;
      if (ch in ['0'..'9','a'..'z','A'..'Z']) and (x<n) then begin
         w := w + ch;
         write(ch);
         x := x + 1;
      end else
      if (ch = #8) then begin
```

```
         if x>0 then begin
            write(#8,' ',#8);
            w[0] := chr(ord(w[0])-1);
            x := x - 1;
         end;
      end else
      if (ch = #0) then begin
         if keypressed then lastchar := readkey;
         case lastchar of
            #72 : x:= n;
            #60 : begin
                    save_data_file;
                    reopen_data_file;
                  end;
         end; {case}
      end;  {if}
   end; {while}
   getstr_alphanum := w;
end;  {getstr_alphanum}


procedure Inverseoff;
begin
   textcolor(white);
   textbackground(black);
end;


procedure Inverseon;
begin
   textcolor(black);
   textbackground(white);
end;

Function istringequiv(inum:integer) : string80;
Var st : string80;
begin
  str(inum,st);
  istringequiv :=st;
end;

function rstringequiv(rnum:real;lnum:integer) : string80;
var st : string80;
begin
  str(rnum:lnum,st);
  rstringequiv := st;
end;


function IValue(s:string80) : integer;
var i : integer;
    e : integer;
begin
   val(s,i,e);
   if e = 0 then ivalue := i
```

```
            else ivalue := -maxint;
end;


function RValue(s:string80) : real;
var r : real;
    e : integer;
begin
   val(s,r,e);
   if e = 0 then rvalue := r
            else rvalue := -maxint*1.0;
end;



procedure Putat (c,r : integer; s : string80);
begin
   GoToXY(c,r);
   write (s);
end;



Procedure Param_Box (sr,sc,bw,bh : integer);
Var
   x : integer;
   ch : char;
Begin
   cursor_off;
   ClrScr;
      For x:=1 to bh-1 do Begin
         putat(sc,sr+x,#186);
         putat (sc+bw-1,sr+x,#186);
      End;
      For x:=1 to bw-2 do Begin
         putat (sc+x,sr,#205);
         putat (sc+x,sr+bh-5,#205);
         putat (sc+x,sr+bh-1,#205);
      End;
      putat(sc,sr,#201);
      putat(sc+bw-1,sr,#187);
      putat(sc,sr+bh-1,#200);
      putat(sc+bw-1,sr+bh-1,#188);
      putat(sc,sr+bh-5,#204);
      putat(sc+bw-1,sr+bh-5,#185);
      putat(sc+10,sr+bh-4,
         'Use '+#24+' To Move to Previous Line');
      putat(sc+10,sr+bh-2,
         'Use RETURN After Each Entry To Proceed');
      cursor_on;
end;
```

```
Procedure Box (sr,sc,bw,bh : integer);
Var
   x : integer;
Begin
   cursor_off;
   ClrScr;
   If (bw>=34) and (bh>=7) then Begin
      For x:=1 to bh-2 do Begin
         putat(sc,sr+x,#186);
         putat (sc+bw-1,sr+x,#186);
      End;
      For x:=1 to bw-2 do Begin
         putat (sc+x,sr,#205);
         putat (sc+x,sr+bh-6,#205);
         putat (sc+x,sr+bh-1,#205);
      End;
      putat(sc,sr,#201);
      putat(sc+bw-1,sr,#187);
      putat(sc,sr+bh-1,#200);
      putat(sc+bw-1,sr+bh-1,#188);
      putat(sc,sr+bh-6,#204);
      putat(sc+bw-1,sr+bh-6,#185);
      putat(sc+((bw-38) shr 1),sr+bh-4,
         'Use '+#24+' Or '+#25+' To Highlight Option');
      putat(sc+((bw-38) shr 1),sr+bh-3,'Press Return To Select Option');
      If Mainmenu = false then
         putat(sc + ((bw-38)shr 1),sr + bh - 2,
            'Press Escape To Go Back To Previous Menu');
   End;
   cursor_on;
End;


Procedure Message_Box (sr,sc,bw,bh : integer);
Var
   x : integer;
Begin
   ClrScr;
   cursor_off;
   If (bw>=66) and (bh>=7) then Begin
      For x:=1 to bh-1 do Begin
         putat(sc,sr+x,#186);
         putat (sc+bw-1,sr+x,#186);
      End;
      For x:=1 to bw-2 do Begin
         Putat(sc+x,sr,#205);
         Putat(sc+x,sr+bh-5,#205);
         Putat(sc+x,sr+bh-1,#205);
      End;
      Putat(sc,sr,#201);
      Putat(sc+bw-1,sr,#187);
      Putat(sc,sr+bh-1,#200);
      Putat(sc+bw-1, sr+bh-1,#188);
      Putat(sc,sr+bh-4,#204);
      Putat(sc+bw-1,sr+bh-4,#185);
      Putat(sc+5,sr+bh-3,'Press any Key To Proceed');
   End;
   cursor_on;
End;


Procedure Init_Box (sr,sc,bw,bh : integer);
Var
   x : integer;
Begin
   ClrScr;
   cursor_off;
   If (bw>=66) and (bh>=7) then Begin
      For x:=1 to bh-5 do Begin
         putat(sc,sr+x,#186);
         putat (sc+bw-1,sr+x,#186);
      End;
      For x:=1 to bw-2 do Begin
         putat (sc+x,sr,#205);
         putat (sc+x,sr+6,#205);
         putat (sc+x,sr+bh-5,#205);
      End;
      putat(sc,sr,#201);
      putat(sc,sr+6,#204);
      putat(sc+bw-1,sr+6,#185);
      putat(sc+bw-1,sr,#187);
      putat(sc,sr+bh-5,#200);
      putat(sc+bw-1,sr+bh-5,#188);
   End;
   cursor_on;
End;



Function Menu_bar (Title:titlestrtype; Opts:optionstype;
                   Numopts:integer) : Integer;
Var
   x    : integer;
   stop : boolean;
   ch   : char;
Begin
   putat((80-length(title)) shr 1,sr+2,Title);
   For x:=1 to numopts do Begin
      GoToxy(sc+5,sr+5+(2*(x-1)));
      Write(Opts[x]);
   End;
   X:=1;
   stop:=false;
   Repeat
      GoToxy(sc+5,sr+5+(2*(x-1)));
      textcolor(black);
      textbackground(white);
      Write (Opts[x]);
      Repeat
         Ch:=readkey;
         If Ch=#0 then Ch:=readkey;
      Until (Ch in [#72,#80,#13,#27,#60]);
      Textcolor(white);
      Textbackground(black);
      Case Ch of
         #13 : Begin
                  Stop:=true;
                  Menu_bar:=x;
               End;
         #72 : Begin
                  putat(sc+5,sr+5+(2*(x-1)),Opts[x]);
                  x:=x-1;
                  If x = 0 then x:=numopts;
               End;
         #80 : Begin
                  putat(sc+5,sr+5+(2*(x-1)),Opts[x]);
                  x:=x+1;
                  If x>numopts then x:=1
               End;
         #60 : begin
                 save_data_file;
                 reopen_data_file;
               end;
         #27 : Begin
                  Menu_bar:=numopts+1;
                  Stop:=true;
               End;
      End;
   Until Stop;
End;


begin
    new(H);
    new(Ho);
    new(Q);
    new(Stor);
    new(T);
    new(Tx);
    new(Ty);
    new(LHd);
    new(Leakance);
    new(hinitial);
    new(Hprime);
for xx := 1 to matrixsize do
   for yy := 1 to matrixsize do begin
       H^[xx,yy]           := 0.0;
       ho^[xx,yy]          := 0.0;
       Q^[xx,yy]           := 0.0;
       Stor^[xx,yy]         := 0.0;
       T^[xx,yy]            := 0.0;
       Tx^[xx,yy]           := 0.0;
       Ty^[xx,yy]           := 0.0;
       Lhd^[xx,yy]          := 0.0;
       Leakance^[xx,yy]     := 0.0;
       Hinitial^[xx,yy]     := 0.0;
       Hprime^[xx,yy]       := 0.0;
   end;
   For i:=1 to 10 do begin
     new(elipsepntr[i]);
   end;
    new(wellpntr);
   datarec.plumeprob          := false;
   datarec.dewateringprob     := false;
   lm                 := lastmode;
   sfilename          := '';
   data_created       := false;
   datafopen          := false;
   well_dry           := false;
end.
```

```
Unit Hydrosys;

Interface
uses Crt,dos,graph,globals,spredsht,solutsys;

procedure hydrogeological_param;


implementation



Procedure hydrogeological_param;
const
   c      : integer=63;
   title  : string[30] = 'Hydrolgeological Parameters';
   title1 : string[30] = 'Dewatering Control Point Data';
   title2 : string[30] = 'Pumping Well Characteristics';
   title3 : string[20] = 'Aquifer Conditions';
   sc     : integer=5;
   sr     : integer=2;
   bw     : integer=70;
   bh     : integer=22;
   pwidth : integer=10;
var
   ch         : char;
   level : integer;
   exitlev : boolean;
   ppos      : integer;
   exit_menu : boolean;
   window    : integer;
   maxwindow : integer;
   x         : integer;
   exit      : boolean;
   parameters: parameterstype;
   diff      : integer;
   maxnumparams    : integer;
   exit_window : boolean;
begin
   clrscr;
   param_box(sr,sc,bw,bh);
   putat((80-length(title)) shr 1, sr+1,title);
   putat(sc+4,sr+6,'Is the aquifer confined or unconfined (C/U)? ');
   putat(sc+4,sr+9,'Input number of site dewatering control points: ');
   putat(sc+4,sr+12,'Input number of pumping wells utilized: ');
   textcolor(0);
   textbackground(1);
   level := 1;
   ppos:= sr+6;
   exitlev := false;
   repeat
      case level of
         1 : begin
                gotoxy(sc+c-14,ppos);
                s := datarec.aquifer;
```

```
     len:=length(s);
     lendiff:=1-len;
     inverseon;
     for i:=1 to lendiff do begin
       write(' ');
     end;
     write(s);
     repeat
       gotoxy(sc+c-14,ppos);
       s := getstr_alphanum(1,s);
     until (((length(s)=1) and (upcase(s[1]) in ['C','U'])) or
     (lastchar = #72));
     if lastch = #72 then level := level - 1;
     inverseoff;
     parameter[1] := upcase(s[1]);
   end;
2 : begin
     repeat
       gotoxy(sc+c-11,ppos+3);
       inum:=datarec.no_dewatering_cntrl_points;
       s:=istringequiv(inum);
       len:=length(s);
       lendiff:=2-len;
       inverseon;
       for i:=1 to lendiff do begin
         write(' ');
       end;
       write(s);
       gotoxy(sc+c-11,ppos+3);
       s := getnum(2,s);
     until (ivalue(s) in [1..10]) or (lastchar = #72);
     if lastch = #72 then level := level-2;
     inverseoff;
     parameter[2] := s;
   end;
3 : begin
     repeat
       gotoxy(sc+c-19,ppos+6);
       inum := datarec.no_pumping_wells;
       s := istringequiv(inum);
       len:=length(s);
       lendiff:=2-len;
       inverseon;
       for i:=1 to lendiff do begin
         write(' ');
       end;
       write(s);
       gotoxy(sc+c-19,ppos+6);
       s := getnum(2,s);
     until (ivalue(s) in [1..10]) or (lastchar = #72);
     if lastch = #72 then level := level-2
                   else exitlev := true;
     inverseoff;
     parameter[3] := s;
   end;
```

```
      end;
      leval := level+1;
   until (exitlev);
   datarec.aquifer:=parameter[1];
   datarec.no_dewatering_cntrl_points:=ivalue(parameter[2]);
   datarec.no_pumping_wells:= ivalue(parameter[3]);
   Case datarec.aquifer[1] of
      #99,#67 : datarec.Confined:=true;
    #117,#85 : datarec.Confined:=false;
   end;
   clrscr;
   if datarec.no_dewatering_cntrl_points <> 0 then begin
      exit_window:= false;
      maxwindow:= datarec.no_dewatering_cntrl_points;
      window:= 1;
      repeat
         clrscr;
         param_box(sr,sc,bw+2,bh+2);
         putat((80-length(title)) shr 1, sr+1,title1);
         putat(wherex-length(title1), sr+3,'Dewatering Control Point # ');
         write(window);
         putat(sc+4, sr+6,'Enter x grid coordinate:  ');
         putat(sc+4, sr+9,'Enter y grid coordinate:  ');
         putat(sc+4,sr+12,'Enter desired water elevation for this control point:
 ');
         putat(sc+4,sr+15,'Enter acceptable range water elevation at control poi
nt:');
         diff:= 4;
         ppos:= sr+6;
         level := 1;
         exitlev := false;
         repeat
            case level of
               1 : begin
                      repeat
                         gotoxy(sc+29,ppos);
                         inum := datarec.control_point[window].cpxcoord;
                         s := istringequiv(inum);
                         len:=length(s);
                         lendiff:=2-len;
                         inverseon;
                         for i:=1 to lendiff do begin
                           write(' ');
                         end;
                         write(s);
                         gotoxy(sc+29,ppos);
                         s := getnum(2,s);
                      until (ivalue(s) in [1..datarec.nc]) or (lastchar = #72);
                      if lastch = #72 then level := level-1;
                      inverseoff;
                      parameter[1] := s;
                   end;
               2 : begin
                      repeat
                         gotoxy(sc+29,ppos+3);
```

```
          inum := datarec.control_point[window].cpycoord;
          s := istringequiv(inum);
          len:=length(s);
          lendiff:=2-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+29,ppos+3);
          s := getnum(2,s);
       until (ivalue(s) in [1..datarec.nr]) or (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[2] := s;
    end;
3 : begin
       repeat
          gotoxy(sc+58,ppos+6);
          rnum := datarec.control_point[window].cphead;
          s := rstringequiv(rnum,6);
          len:=length(s);
          lendiff:=6-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+58,ppos+6);
          s := getnum(6,s);
       until ((rvalue(s) >= 0.0) and (rvalue(s) <= 999999))
          or (lastchar = #72);
       if lastch = #72 then level := level-2;
       inverseoff;
       parameter[3] := s;
    end;
4 : begin
       repeat
          gotoxy(sc+61,ppos+9);
          rnum := datarec.control_point[window].cph_range;
          s := rstringequiv(rnum,4);
          len:=length(s);
          lendiff:=4-len;
          inverseon;
          for i:=1 to lendiff do begin
            write(' ');
          end;
          write(s);
          gotoxy(sc+61,ppos+9);
          s := getnum(4,s);
       until ((rvalue(s) >= 0.0) and (rvalue(s) <= 9999))
          or (lastchar = #72);
       if lastch = #72 then level := level-2
                        else exitlev := true;
       inverseoff;
```

```
                      parameter[4] := s;
                   end;
            end;
            level := level+1;
         until (exitlev);
         with datarec.control_point[window] do begin
            cpxcoord:= ivalue(parameter[1]);
            cpycoord:= ivalue(parameter[2]);
            cphead:= rvalue(parameter[3]);
            cpH_range:=rvalue(parameter[4]);
         end;
         window := window + 1;
         if window > maxwindow then exit_window := true;
      until exit_window;
   end;
   if datarec.no_pumping_wells <> 0 then begin
      exit_window:=  false;
      maxwindow:= datarec.no_pumping_wells;
      window:= 1;
      repeat
         clrscr;
         param_box(sr,sc,bw+2,bh+2);
         putat((80-length(title2)) shr 1 , sr+1,title2);
         putat(wherex-length(title2), sr+3,'Pumping Well #');
         write(window);
         putat(sc+2,sr+7,'Pumping Rate: ');
         putat(sc+2,sr+9,'Enter Maximum Percent Dewatering of Available Yield: '
);
         putat(sc+2,sr+11,'Enter x coord for top left corner of search area: ');
         putat(sc+2,sr+13,'Enter y coord for top left corner of search area: ');
         putat(sc+2,sr+15,'Enter x coord for bottom right corner of search area
: ');
         putat(sc+2,sr+17,'Enter y coord for bottom right corner of search area
: ');
         diff:= 2;
         ppos:= sr + 7;
         level := 1;
         exitlev := false;
         repeat
            case level of
               1 : begin
                      repeat
                         gotoxy(sc+21,ppos);
                         rnum := datarec.wellno[window].well_pump_rate;
                         s := rstringequiv(rnum,10);
                         len:=length(s);
                         lendiff:=10-len;
                         inverseon;
                         for i:=1 to lendiff do begin
                           write(' ');
                         end;
                         write(s);
                         gotoxy(sc+21,ppos);
                         s := getnum(10,s);
```

```
                until ((rvalue(s) >= -9999999.99) and (rvalue(s) <= 999999
9.99))
                  or (lastchar = #72);
                if lastch = #72 then level := level-1;
                inverseoff;
                parameter[1] := s;
              end;
           2 : begin
                repeat
                  gotoxy(sc+61,ppos+2);
                  rnum := datarec.wellno[window].percent_dewater;
                  s := rstringequiv(rnum,6);
                  len:=length(s);
                  lendiff:=6-len;
                  inverseon;
                  for i:=1 to lendiff do begin
                    write(' ');
                  end;
                  write(s);
                  gotoxy(sc+61,ppos+2);
                  s := getnum(6,s);
                until ((rvalue(s) >= 0.0) and (rvalue(s) <= 1.0))
                  or (lastchar = #72);
                if lastch = #72 then level := level-2;
                inverseoff;
                parameter[2] := s;
              end;
           3 : begin
                repeat
                  gotoxy(sc+61,ppos+4);
                  inum := datarec.wellno[window].tlx;
                  s := istringequiv(inum);
                  len:=length(s);
                  lendiff:=2-len;
                  inverseon;
                  for i:=1 to lendiff do begin
                    write(' ');
                  end;
                  write(s);
                  gotoxy(sc+61,ppos+4);
                  s := getnum(2,s);
                until ((rvalue(s) >= 0.0) and (rvalue(s) <= datarec.nc))
                  or (lastchar = #72);
                if lastch = #72 then level := level-2;
                inverseoff;
                parameter[3] := s;
              end;
           4 : begin
                repeat
                  gotoxy(sc+61,ppos+6);
                  inum := datarec.wellno[window].tly;
                  s := istringequiv(inum);
                  len:=length(s);
                  lendiff:=2-len;
                  inverseon;
                  for i:=1 to lendiff do begin
                  write(' ');
                end;
                write(s);
                gotoxy(sc+61,ppos+6);
                s := getnum(2,s);
             until ((rvalue(s) >= 0.0) and (rvalue(s) <= datarec.nr))
                or (lastchar = #72);
             if lastch = #72 then level := level-2;
             inverseoff;
             parameter[4] := s;
          end;
      5 : begin
             repeat
                gotoxy(sc+61,ppos+8);
                inum := datarec.wellno[window].brx;
                s := istringequiv(inum);
                len:=length(s);
                lendiff:=2-len;
                inverseon;
                for i:=1 to lendiff do begin
                  write(' ');
                end;
                write(s);
                gotoxy(sc+61,ppos+8);
                s := getnum(2,s);
             until ((rvalue(s) >= 0.0) and (rvalue(s) <= datarec.nc))
                or (lastchar = #72);
             if lastch = #72 then level := level-2;
             inverseoff;
             parameter[5] := s;
          end;
      6 : begin
             repeat
                gotoxy(sc+61,ppos+10);
                inum := datarec.wellno[window].bry;
                s := istringequiv(inum);
                len:=length(s);
                lendiff:=2-len;
                inverseon;
                for i:=1 to lendiff do begin
                  write(' ');
                end;
                write(s);
                gotoxy(sc+61,ppos+10);
                s := getnum(2,s);
             until ((rvalue(s) >= 0.0) and (rvalue(s) <= datarec.nr))
                or (lastchar = #72);
             if lastch = #72 then level := level-2
                                else exitlev := true;
             inverseoff;
             parameter[6] := s;
          end;
   end;
   level := level+1;
until (exitlev);
```

```
        with datarec.wellno[window]  do begin
           well_pump_rate:= rvalue(parameter[1]);
           percent_dewater := rvalue(parameter[2]);
           tlx:= ivalue(parameter[3]);
           tly:= ivalue(parameter[4]);
           brx:= ivalue(parameter[5]);
           bry:= ivalue(parameter[6]);
           trx := brx;
           try := tly;
           blx := tlx;
           bly := bry;
        end;
        window:= window + 1;
        if window > maxwindow then exit_window := true;
     until exit_window;
  end;
  param_box(sr,sc,bw,bh);
  putat((80-length(title)) shr 1, sr+2,title3);
  exit_window:= false;
  maxwindow:= 1;
  window:= 1;
  Repeat
     putat(sc+3,sr+7,'Do You Have Leaking Artesian Conditions (Y/N)? ');
     putat(sc+3,sr+12,'Do You Have Induced Infiltration from a Stream or Pond?
(Y/N)? ');
     textcolor(0);
     textbackground(1);
     exit_menu:=false;
     exit:=false;diff:=5;
     ppos:= sr+7;
     level := 1;
     exitlev := false;
     repeat
        case level of
           1 : begin
                  gotoxy(sc+50,ppos);
                  s := datarec.leakage;
                  len:=length(s);
                  lendiff:=1-len;
                  inverseon;
                  for i:=1 to lendiff do begin
                    write(' ');
                  end;
                  write(s);
                  repeat
                     gotoxy(sc+50,ppos);
                     s := getstr_alphanum(1,s);
                  until (((length(s)=1) and (upcase(s[1]) in ['Y','N'])) or
                  (lastchar=#72));
                  inverseoff;
                  parametor[1] := upcase(s[1]);
               end;
           2 : begin
                  gotoxy(sc+66,ppos+5);
                  s := datarec.Infiltration;
```

```
                   len:=length(s);
                   lendiff:=1-len;
                   inverseon;
                   for i:=1 to lendiff do begin
                     write(' ');
                   end;
                   write(s);
                   repeat
                      gotoxy(sc+66,ppos+5);
                      s := getstr_alphanum(1,s);
                   until (((length(s)=1) and (upcase(s[1]) in ['Y','N']))
                         or (lastchar = #72));
                   if lastch = #72 then level := level-2
                                    else exitlev := true;
                   inverseoff;
                   parameter[2] := upcase(s[1]);
                end;
         end;
         level := level+1;
      until (exitlev);
      datarec.Leakage:=parameter[1];
      datarec.Infiltration:=parameter[2];
      window:= window + 1;
      if window > maxwindow then exit_window := true;
   Until Exit_Window;
   Case datarec.Leakage[1] of
      'Y','y' : datarec.Leaky_Artesian_Conditions:=true;
      'N','n' : datarec.Leaky_Artesian_Conditions:=false;
   end;
   Case datarec.Infiltration[1] of
      'Y','y' : datarec.Induced_Infiltration:=true;
      'N','n' : datarec.Induced_Infiltration:=false;
   end;
end;

end.   {unit}
```

```
unit spredsht;
interface
Uses Crt,dos,globals;

procedure Spread_sheet (nr,nc:integer; fname,tit:string80);

implementation
Const
  Maxcol=matrixsize;
  Maxrow=matrixsize;
  cwidth=11;        {12-11 = 1 blank}
Type
  Cell_Type=string[cwidth];
  Spread_Type=Array[1..maxrow,1..maxcol] of Cell_Type;
Var
  numrow : integer;
  status : boolean;
  numcol : integer;
  Spread : Spread_Type;
  X,Y,Z,XC,YC,Scrtopx,Scrtopy:Integer;
  Ch:Char;
  LastChar:Char;
  Exit_Cmd:Boolean;
  Exit_Cell:Boolean;
  Exit:Boolean;
  Saved_SS:Boolean;
  cd:STRING80;
  lm : integer;
  cdCode:Integer;
  cdnum:real;
  ValOK:Boolean;
  reg : registers;
  data : text;


procedure Spread_sheet (nr,nc:integer; fname,tit:string80);


procedure Cursor_off;
begin
   with reg do begin
      ah := 1;
      ch := 32;
   end;
   intr($10,reg);
end;


procedure Cursor_on;
begin
   with reg do begin
      ah:=1;
      if lm = mono then begin
                           ch := 11;
                           cl := 12;
```

```
                      end else
                      begin
                         ch := 6;
                         cl := 7;
                      end;
   end;
   intr($10,reg);
end;


Procedure Move_Cursor;
Begin
   Gotoxy(4+(xc-scrtopx)*12,3+(yc-scrtopy));
End;


procedure Title;
begin
   gotoxy((78-length(tit)) div 2,25);
   write(tit);
end;


Procedure Drawtopline;
Var
  x:Integer;
Begin
  TextColor(black);
  TextBackGround(white);
  For x:=1 to 6 do begin
    GoToxy(8+(x-1)*12,1);
    If Scrtopx>26 then Write('A') else Write(' ');
    Write(chr((scrtopx+x-2)mod 26 +65));
  End;
  textcolor(white);
  textbackground(black);
End;


Procedure Drawsideline;
Var
  y:integer;
Begin
   textcolor(black);
   textbackground(white);
  For y:=1 to 20 do begin
    Gotoxy(1,y+2);
    Write((scrtopy+y-1):2);
  End;
  textcolor(white);
  textbackground(black);
End;
```

```
Procedure Drawsheet ;
Var
  x,y,z:Integer;
Begin
  textcolor(white);
  textbackground(black);
  For x:=1 to 6 do begin
    For y:=1 to 20 do begin
      Gotoxy(4+(x-1)*12,3+(y-1));
      if ((scrtopx+x-1)<=maxcol) and ((scrtopy+y-1)<=maxrow) then begin
         Write (Spread[scrtopx+x-1,scrtopy+y-1]);
         For z:=length(spread[scrtopx+x-1,scrtopy+y-1])+1 to cwidth do
           Write(' ');
      end;
    End;
  End;
  title;
End;


Procedure Draw_Commands;
Begin
  Gotoxy(6,24);
  textcolor(black);
  textbackground(white);
  Write('Save              Load        Clear        Edit        Replicate       Quit '
);
  textcolor(white);
  textbackground(black);
End;


procedure Blank_commands;
begin
   Gotoxy(6,24);
   Write('
');
end;


Procedure Draw_No_Commands;
Begin
  GoToxy(6,24);
  Write('                     * Press F1 For Commands *                         '
);
End;


function Validcell(s:string80) : boolean;
var v : boolean;
    t : integer;
    x : integer;
begin
   t := 0;
   v := true;
   for x := 2 to length(s) do
      t := (t * 10) + (ord(s[x])-ord('0'));
   case length(s) of
      2..4 : begin
               if not((upcase(s[1]) in ['A'..'Z']) and (t<=nr)) then
                  v := false;
            end;
   else
      v := false;
   end;
   validcell := v;
end;


procedure Copycells (s,f:string80);
var x,y : integer;
    x1,y1 : integer;
    x2,y2 : integer;
    curc : cell_type;
begin
   curc := spread[xc,yc];
   x1 := ord(upcase(s[1]))-ord('A')+1;
   x2 := ord(upcase(f[1]))-ord('A')+1;
   y1 := 0;
   y2 := 0;
   for x := 2 to length(s) do
      y1 := (y1 * 10) + (ord(s[x])-ord('0'));
   for x := 2 to length(f) do
      y2 := (y2 * 10) + (ord(f[x])-ord('0'));
   for x := x1 to x2 do
      for y := y1 to y2 do
         spread[x,y] := curc;
end;


Procedure Highlight;
Var
  x:integer;
Begin
  TextColor(black);
  TextBackGround(white);
  For x:=1 to cwidth do
    If x>Length(spread[xc,yc]) then write(' ')
    else write(Spread[xc,yc][x]);
  GoToxy(wherex-cwidth,wherey);
End;


Procedure UnHighLight;
Var
  x:integer;
```

```
Begin
  TextColor(white);
  Textbackground(black);
  For x:=1 to cwidth do
    If x>Length(spread[xc,yc]) then write (' ')
    Else write(Spread[xc,yc][x]);
  Gotoxy(Wherex-cwidth,wherey);
End;


procedure Replicate;
var
   celladd : string80;
   celladd2 : string80;
begin
   blank_commands;
   repeat
      gotoxy(6,24);
      write('REPLICATE CURRENT CELL STARTING WHERE (e.g. A ) : ');
      readln(celladd);
   until validcell(celladd);
   blank_commands;
   repeat
      gotoxy(6,24);
      write('REPLICATE CURRENT CELL ENDING WHERE (e.g. Z12) : ');
      readln(celladd2);
   until validcell(celladd2);
   copycells(celladd,celladd2);
   cursor_off;
   TextColor(black);
   TextBackGround(white);
   clrscr;
   DrawTopLine;
   DrawSideLine;
   textcolor(white);
   textbackground(black);
   DrawSheet;
   Draw_Commands;
   move_cursor;
   highlight;
   cursor_on;
end;


function Load : boolean;
var
   status : integer;
   x,y : integer;
   l : boolean;
Begin
   assign(data,fname);
   {$I-}
   reset(data);
   {$I+}
   status := ioresult;
   if status <> 0 then begin
      l := false;
      rewrite(data);
   end else begin
      {$I-}
      for x := 1 to maxrow do
         for y := 1 to maxcol do
            readln(data,spread[x,y]);
      {$I+}
      status := ioresult;
      if status <> 0 then l := falsu
                     else l := true;
   end;
   close(data);
   load := l;
End;


procedure Save;
var
   x,y : integer;
Begin
   assign(data,fname);
   rewrite(data);
   for x := 1 to maxcol do
      for y := 1 to maxrow do begin
         writeln(data,spread[x,y]);
      end;
   close(data)
End;


Procedure Quit;
Begin
  Exit:=true;
  Exit_cmd:=true;
End;


Procedure Edit;
Begin
  Exit_cmd:=Truc;
  Saved_ss:=false;
  cursor_off;
  Draw_No_Commands;
  cursor_on;
End;
```

```
procedure Newspread;
begin
   clrscr;
   writeln;
   swapvectors;
   exec('\COMMAND.COM','/C DIR/W');
{   writeln(doserror);
   readln;}
   swapvectors;
   writeln;
   write('Please enter new name for spreadsheet: ');
   readln(fname);
end;


Procedure Clear;
Var
  y,x:integer;
Begin
  For x:=1 to maxcol do
    For y:=1 to maxrow do Spread[x,y]:='';
  Saved_ss:=false;
  cursor_off;
  Drawsheet;
  cursor_on;
End;


Function GetText_CR(cwidth:Integer):string80;
Var
  Exit:Boolean;
  Wordx:Integer;
  Word:string80;
  dot,dash,expon,negexp,numb : boolean;
  ch:char;
Begin
  Exit:=False;
  Exit_Cell:=False;
  dot := false;
  dash := false;
  expon := false;
  negexp := false;
  numb := false;
  if length(spread[xc,yc])>0 then begin
     numb := true;
     dash := true;
  end;
  for x:= 1 to length(spread[xc,yc]) do begin
     if spread[xc,yc][x] = '.' then dot := true;
     if spread[xc,yc][x] = 'E' then expon := true;
     if (spread[xc,yc][x]='-') and (x>1) and (spread[xc,yc][x-1]='E') then
        negexp := true
     else if (spread[xc,yc][x]='-') then
        dash := true;
  end;
```

```
Wordx:=Length(Spread[xc,yc]);
Word:=Spread[xc,yc];
Move_Cursor;
 gotoxy(wherex+wordx,wherey);
Repeat
  Ch:=readkey;
  Case ch of
    '.' :  if not dot then begin
                              Wordx:=wordx+1;
                              If wordx>cwidth then
                                 wordx:=cwidth
                              Else
                              Begin
                                 Write(Ch);
                                 Word[wordx]:=Ch;
                                 dot := true;
                                 dash := true;
                              End;  {if}
                          End; {begin}
    'e','E' :  if numb and (not expon) then begin
                              Wordx:=wordx+1;
                              If wordx>cwidth then
                                 wordx:=cwidth
                              Else
                              Begin
                                 Write(Ch);
                                 Word[wordx]:=upcase(Ch);
                                 expon := true;
                                 dot := true;
                                 dash := true;
                              End;  {if}
                          End; {begin}
    '-': if (not dash) or ((not negexp) and (word[wordx]='E') and numb)
         then Begin
                                 Wordx:=wordx+1;
                                 If wordx>cwidth then
                                    wordx:=cwidth
                                 Else Begin
                                    Write(Ch);
                                    Word[wordx]:=Ch;
                                    if (numb and (word[wordx-1]='E')) then
                                    begin
                                       negexp := true;
                                       dash := true;
                                    end else
                                    dash := true;
                                 End;  {if}
                         End; {begin}
    '0'..'9': Begin
                  Wordx:=wordx+1;
                  If wordx>cwidth then
                     wordx:=cwidth
                  Else Begin
                     Write(Ch);
                     Word[wordx]:=Ch;
                  dash := true;
                  numb := true;
               End;  (if)
            End; (begin)
  #8:Begin
       Wordx:=wordx-1;
       If wordx<0 then begin
         Wordx:=0;
         dot := false;
         dash := false;
         numb := false;
         expon := false;
         negexp := false;
       end Else begin
         Write(#8,' ',#8);
         if word[wordx+1] = '.' then dot := false;
         if word[wordx+1] = 'E' then expon := false;
         if (word[wordx+1] = '-') and (word[wordx]<>'E') then
            dash := false
         else if (word[wordx+1]='-') then
            negexp := false;
         Word[wordx+1]:=' ';
         if wordx = 0 then begin
            dot := false;
            dash := false;
            expon := false;
            negexp := false;
            numb := false;
         end;
       End;    (If)
     End;    (begin)
  #13:Begin
        LastChar:=#13;
        Exit:=True;
        Exit_Cell:=true;
        word[0]:=Chr(wordx);
        Gotoxy(wherex-wordx,wherey);
      End;   (Begin)
  #0:Begin
       ch:=readkey;
       Case ch of
         #59,#75,#77,#72,#80,#71:Begin
                                   Gotoxy(wherex-wordx,wherey);
                                   Lastchar:=Ch;
                                   Exit:=True;
                                   Exit_Cell:=True;
                                   Word[0]:=chr(wordx);
                                 End;   (begin)
         #60 : begin
                 save_data_file;
                 reopen_data_file;
               end;
       End;  (Case)
     End;  (Begin)
End;   (Case)
```

```
  Until Exit;
  GetText_CR:=Word;
End; (Function)

Begin
  lm := lastmode;
  TextColor(black);
  TextBackGround(white);
  if fname = '' then newspread;
  ClrScr;
  Exit:=False;
  xc:=1;yc:=1;
  scrtopx:=1;scrtopy:=1;
  status := Load;
  if not status then begin
     For x:=1 to maxcol do
        For y:=1 to maxrow do
           Spread[x,y]:='';
  end;
  cursor_off;
  DrawTopLine;
  DrawSideLine;
  textcolor(white);
  textbackground(black);
  DrawSheet;
  Draw_No_Commands;
  cursor_on;
  Repeat
    Repeat
       move_cursor;
       highlight;
       cd:=GetText_CR(cwidth);
       unhighlight;
       Val(cd,cdnum,cdcode);
       ValOK:=(exit_cell=true) or (cdcode=0);
    Until ValOK;
    Spread[xc,yc]:=cd;
    Case LastChar of
      #59:Begin      (F1)
            move_cursor;
            highlight;
            Exit_Cmd:=False;
            cursor_off;
            Draw_Commands;
            Repeat
              Ch:=Readkey;
              cursor_on;
              Case Ch of
                #67,#99   : Clear,
                's','S'   : Save;
                'r','R'   : replicate;
                'l','L'   : begin
                               newspread;
                               clrscr;
                               status := Load;
```

```
                       if not status then begin
                          For x:=1 to maxcol do
                             For y:=1 to maxrow do
                                Spread[x,y]:='';
                       end;
                       cursor_off;
                       TextColor(black);
                       TextBackGround(white);
                       clrscr;
                       xc:=1;yc:=1;
                       scrtopx:=1;scrtopy:=1;
                       DrawTopLine;
                       DrawSideLine;
                       textcolor(white);
                       textbackground(black);
                       DrawSheet;
                       Draw_Commands;
                       cursor_on;
                    end;
          #113,#81 : begin
                       save;
                       exit_cmd := true;
                       exit := true;
                    end;
          #101,#69 : Edit;
         #0:Begin
               ch:=readkey;
               Case ch of
                  #60 : begin
                          save_data_file;
                          reopen_data_file;
                        end;
               End;  {Case}
            end;
        End; {case}
        cursor_off;
      Until Exit_Cmd;
      cursor_on;
    End;
#75:Begin    { <-- }
      Unhighlight;
      XC:=XC-1;
      If xc>=scrtopx then begin
        Move_Cursor;Highlight;
      end;
      If Xc<1 then
        begin
          Write(#7);
          XC:=1;
        End
      else If Xc<scrtopx then
        begin
          scrtopx:=scrtopx-1;
          cursor_off;
          drawtopline;
          Drawsheet;
          Move_Cursor;Highlight;
          cursor_on;
        End;
    End;
#77:Begin    { --> }
      Unhighlight;
      xc:=xc+1;
      If xc<=scrtopx+5 then begin
        Move_Cursor;Highlight;
      end;
      If xc>nc then
        Begin
          Write(#7);
          xc:=nc;
        End
      Else If xc>scrtopx+5 then
        begin
          scrtopx:=scrtopx+1;
          cursor_off;
          drawtopline;
          Drawsheet;
          Move_Cursor;Highlight;
          cursor_on;
        End;
    End;
#72:Begin     { ^ }
      Unhighlight;
      yc:=yc-1;
      If yc>=scrtopy then begin
        Move_Cursor;Highlight;
      end;
      If yc<1 then
      Begin
        Write(#7);
        yc:=1;
      End
      Else If yc<scrtopy then
      Begin
        scrtopy:=scrtopy-1;
        cursor_off;
        Drawsideline;
        Drawsheet;
        Move_Cursor;Highlight;
        cursor_on;
      End;
    End;
#80:Begin     { v }
      Unhighlight;
      yc:=yc+1;
      If yc<=scrtopy+19 then begin
        Move_Cursor;Highlight;
      end;
      If yc>nr then
        Begin
```

```
                 unhighlight;
                 yc:=nr;
                 Write(#7);
              End
            Else if yc>scrtopy +19 then
              Begin
                scrtopy:=scrtopy+1;
                cursor_off;
                Drawsideline;
                Drawsheet;
                Move_Cursor;Highlight;
                cursor_on;
              End;
          End;

      #71:Begin
            xc:=1; yc:=1;
            scrtopx:=1; scrtopy:=1;
            cursor_off;
            Drawtopline;
            Drawsideline;
            Drawsheet;
            cursor_on;
          End;
      #13:Begin
            Unhighlight;
            yc:=yc+1;
            If yc<=scrtopy+19 then
              Move_Cursor;Highlight;
            If yc>nr then
              Begin
                 unhighlight;
                 yc:=nr;
                 Write(#7);
              End
            Else if yc>scrtopy +19 then
              Begin
                scrtopy:=scrtopy+1;
                cursor_off;
                Drawsideline;
                Drawsheet;
                Move_Cursor;Highlight;
                cursor_on;
              End;
          End;
    End;  {Case}
  Until Exit;

End;

end. {unit}
```

```
{$N+} {$E+}
unit solutsys;
interface
uses dos,crt,graph,globals;

Procedure Finite_difference_approximation;
Procedure Draw_search_grid(nc,nr : integer);
Procedure Erase_well;
Procedure Put_well(x:integer);
Procedure Set_Well(n,a,b:integer);
Procedure Search(n:integer);
Procedure Drawellipse(xc,yc,xr,yr : real);
Procedure Put_plume_area(Hspace,Vspace,nplume:Integer);
Procedure Put_search_areas(Hspace,Vspace,no_pumping_wells:Integer);
Procedure Create_plume_solution_summary;
Procedure Create_dewatering_solution_summary;
Procedure Assess_plume_coverage;
Procedure Capture(i:integer);
Procedure Plume_Solution_Summary;
Procedure Dewatering_Solution_Summary;
procedure getmaxinelipse(x:integer);
function Getdistance (i,j,x:integer) : real;
function inellipse(i,j,x:integer) : boolean;
function inany(i,j : integer) : boolean;
function getconc : real;

implementation

  var poutfile : text;
      doutfile : text;


  procedure dewatering_solution_summary;
  begin
  end;

  procedure plume_solution_summary;
  begin
  end;


  procedure create_dewatering_solution_summary;
  begin
  end;


  function Getdistance (i,j,x:integer) : real;
  var g : real;
  begin
      getdistance := sqrt(sqr(i-datarec.wellno[x].x) +
                          sqr(j-datarec.wellno[x].y));
  end;
```

```
procedure getmaxinelipse(x:integer);
var i,j,k,l : integer;
begin
   datarec.wellno[x].conc := 0.0;
   for i:= 1 to datarec.nc do begin
      for j := 1 to datarec.nr do begin
         if inellipse(i,j,x) and
            not ((i=datarec.wellno[x].x) and
                  (j=datarec.wellno[x].y)) and
             (contaminant_concentration^[i,j]>=datarec.wellno[x].conc) then
         begin
               datarec.wellno[x].conc := contaminant_concentration^[i,j];
               datarec.wellno[x].concx := i;
               datarec.wellno[x].concy := j;
         end;
      end;
   end;
end;


procedure create_plume_solution_summary;
var z : integer;
begin

   for z:= 1 to datarec.no_pumping_wells do begin
      datarec.wellno[z].x := datarec.wellno[z].maxxl;
      datarec.wellno[z].y := datarec.wellno[z].maxyl;
      getmaxinelipse(z);
      datarec.wellno[z].bestdist :=
             getdistance(datarec.wellno[z].concx,datarec.wellno[z].concy
                         ,z) *  datarec.delxy;
   end;
   finite_difference_approximation;
end;


function inellipse(i,j,x:integer) : boolean;
var r1,r2 : real;
begin
   r1 := (i*1.0) - (datarec.wellno[x].x*1.0);
   r1 := (r1/datarec.wellno[x].xrad) * (r1/datarec.wellno[x].xrad);
   r2 := (j*1.0) - (datarec.wellno[x].y*1.0);
   r2 := (r2*r2) / (datarec.wellno[x].yrad*datarec.wellno[x].yrad);
   if r1 + r2 <= 1.0 then inellipse := true
                     else inellipse := false;
end;


function inany(i,j : integer) : boolean;
var
   inone : boolean;
   x     : integer;
begin
```

```
     inone := false;
     for x := 1 to datarec.no_pumping_wells do
        if inellipse(i,j,x) then inone := true;
     inany := inone;
  end;


  function getconc : real;
  var
     totconc : real;
     i,j     : integer;
  begin
     totconc := 0.0;
     for i:= 1 to datarec.nc do
        for j := 1 to datarec.nr do
           if inany(i,j) then
              totconc := totconc + contaminant_concentration^[i,j];
     getconc := totconc;
  end;


  procedure Assess_plume_coverage;
  var r : real;
       x : integer;
  begin
     r := getconc;
     if r > datarec.maxconc then begin
        datarec.maxconc := r;
        for x := 1 to datarec.no_pumping_wells do begin
           datarec.wellno[x].maxxl   := datarec.wellno[x].x;
           datarec.wellno[x].maxyl   := datarec.wellno[x].y;
           datarec.wellno[x].maxxrad := datarec.wellno[x].xradius;
           datarec.wellno[x].maxyrad := datarec.wellno[x].yradius;
        end;
     end;
   end;


procedure Erase_well;
Begin
   putimage(0,0,wellpntr^,0);
   If datarec.plumeprob then Drawellipse(xcenter,ycenter,xr,yr);
End;  {procedure}


Procedure Finite_Difference_Approximation;
Type
  BGType=Array[1..50] of real;
Var
  Convergence,Successful,Exit_FD          :Boolean;
  B                                       :BGtype;
  G                                       :BGType;
  Lhi,Lhj,hi,hj,bgi,bgj,ci,cj             :Integer;
```

```
ni,nj,nci,nrj,ri,rj,wi,wj,i,j            :Integer;
istep,iter,sum                           :integer;
Time,Year                                :Real;
Hd,F,E                                   :Real;
ac,bc,cc,dc,ar,br,cr,dr                  :Real;
wc,wr,HA,wdiff,totout,totin,store,pump   :real;
Delxsq,delta                             :Real;
ch                                       :char;
Percent_step                             :real;


Procedure Print_head_map(time:real);
Var i,j   : integer;
  ch      : char;
Begin
  Restorecrtmode;
  Clrscr;
  writeln('Time in Days = ',Time:4:1);
  writeln('istep = ',istep);
  Writeln('iteration number = ',iter);
  Writeln('totin = ',totin);
  Writeln('totout = ',totout);
  Ch:=readkey;
  For i:=1 to datarec.nc do begin
    For j:=1 to datarec.nr do begin
      Write(H^[i,j]:5:1);
      Write('  ');
    end;
    writeln(' ');
  end;
  ch:=readkey;
  Initgraph(gd,gm,'c:\tp');
end;


Procedure unsuccessfull;
Const
  sc                                     : integer=5;
  sr                                     : integer=2;
  bw                                     : integer=70;
  bh                                     : integer=25;
Var
  ch                                     : char;
Begin
  RestoraCrtMode;
  ClrScr;
  Box(sc,sr,bw,bh);
  Putat(5,10,'An Optimum Dewatering Well Configuration has not been Found');
  Putat(5,13,'The Calculated Heads did not Match the Stipulated Control-Point
Heads');
  Putat(5,16,'Try first to Increase your Error Margin for Matching the Control
-Point Heads');
  Putat(5,19,'Subsequently, try altering other input parameter values');
  Exit_FD := true;
  Istep := datarec.nsteps;
  Exit_search:=true;
  ch := readkey;
End;


Procedure Success(time:real);

var
   window1,window2 : windowptr;


  Procedure Window (Var Win1,Win2:Windowptr;time:real);
  Const
    W1= '*success*';
    w2='An optimum well configuration';
    W3='       has been found';
    W4='The simulated pumping period equals';
    W5='The solution will be written to an';
    W6='output file called "filename".OUT';
    W7='     <PRESS RETURN TO CONTINUE>';
    W8='    <PRESS ESC TO EXIT SEARCH>';
    W9='Days';
    ClipOn=true;
  Var
    CH          : Char;
    timestr     : string80;
    i,j         : integer;

  Begin

     str(time:5:2,timestr);
     SetViewPort( 323,20,635,196,clipon);
     ClearViewPort;
     SetLineStyle(0,0,3);
     Rectangle(0,0,312,176);
     SetTextStyle(1,0,1);
     SetTextJustify(1,0);
     Moveto(160,15);OutText(W1);
     SetTextStyle(0,0,1);
     SetTextJustify(0,1);
     MoveTo(40,23);OutText(W2);
     MoveTo(60,31);OutText(W3);
     SetTextJustify(0,1);
     Moveto(23,70);Outtext(W4);
     Moveto(150,80);Outtext(timestr);
     Moveto(195,80);Outtext(W9);
     MoveTo(23,96);OutText(W5);
     MoveTo(23,104);OutText(W6);
     MoveTo(16,158);OutText(W7);
     moveto(16,171);outtext(w8);
     GetImage(323,20,635,196,Win2^);

     Repeat
       CH:=readkey;
```

```
        Until (CH in[#13,#27]);
        Case ch of
          #13 : begin
                  exit_fd:=true;
                  istep:=datarec.nsteps;
                  successful:=false;
                  Append(doutfile);
                  Writeln(doutfile);
                  For i := 1 to datarec.no_pumping_wells do begin
                     Writeln(doutfile,'Pumping rate for well # ',i,' = '
                             ,datarec.wellno[i].well_pump_rate:10:2);
                     writeln(doutfile,'Grid coordinates for well ',i,' ( '
                             ,datarec.wellno[i].x,' , ',datarec.wellno[i].y,' )')
 ;

                     writeln(doutfile);
                  End;
                  For i := 1 to datarec.no_dewatering_cntrl_points do begin
                     Write(doutfile,'Control point ',i,' coordinates = ('
                                  ,datarec.control_point[i].cpxcoord);
                     writeln(doutfile,', ',datarec.control_point[i].cpycoord,')')
 ;

                     Writeln(doutfile,'Control point ',i,' desired elevation = ('
                                     ,datarec.control_point[i].cphead:7:2,' )');
                     Writeln(doutfile,'Control point ',i,' calculated elevation =
  ('

                     ,h^[datarec.control_point[i].cpycoord,datarec.control_point[
i].cpxcoord]:7:2,' )');
                  end;
                  Writeln(doutfile,'The following is the solution head map achiev
ed');

                  Writeln(doutfile,'after a simulated pumping period of ',time:4:
1,' days');

                  Writeln(doutfile);
                  For i:=1 to datarec.nc do begin
                     for j:=1 to datarec.nr do begin
                        write(doutfile,H^[i,j]:7:2);
                     end;
                     writeln(doutfile);
                  end;
                  writeln(doutfile);
                  close(doutfile);
                end;
          #27 : begin
                  exit_fd:=true;
                  istep:=datarec.nsteps;
                  exit_search:=true;
                  Append(doutfile);
                  Writeln(doutfile);
                  For i := 1 to datarec.no_pumping_wells do begin
                     Writeln(doutfile,'Pumping rate for well # ',i,' = ',
                                        datarec.wellno[i].well_pump_rate);
                     writeln(doutfile,'Grid coordinates for well ',i,' = ( '
                                     ,datarec.wellno[i].x,' , ',datarec.wellno[i]
.y,' )');

                     writeln(doutfile);
                End;
                For i := 1 to datarec.no_dewatering_cntrl_points do begin
                   Writeln(doutfile,'Control point ',i,'coordinates = (',datare
c.control_point[i].cpxcoord,
                                   ' , ',datarec.control_point[i].cpycoord,' )
');
                   Writeln(doutfile,'Control point ',i,'desired elevation = ('
                                  ,datarec.control_point[i].cphead,' )');
                   Writeln(doutfile,'Control point ',i,'calculated elevation =
('
                                  ,h^[datarec.control_point[i].cpycoord,datare
c.control_point[i].cpxcoord],' )');
                end;
                Writeln(doutfile,'The following is the solution head map');
                Writeln(doutfile,'achieved after ',time:4:1,' days of pumping:'
);
                Writeln(doutfile);
                For i:=1 to datarec.nc do begin
                   for j:=1 to datarec.nr do begin
                      write(doutfile,H^[i,j]:7:2);
                   end;
                   writeln(doutfile);
                end;
                writeln(doutfile);
                close(doutfile);
              end;
       end;  {case}
    End;

  Begin
    New(Window1);
    New(Window2);
    GetImage(323,20,635,196,Window1^);
    Window (Window1,window2,time);
    SetViewPort(0,0,639,199,clipon);
    Clearviewport;
    PutImage(323,20,Window1^,0);
    dispose(window1);
    dispose(window2);
  End;


  Procedure Compare_control_points(time:real);
  Var
    i : integer;
    diff: real;
    successful:boolean;
  Begin
    successful:=true;
    for i:=1 to datarec.no_dewatering_cntrl_points do begin
        diff:=abs(h^[datarec.control_point[i].cpycoord,datarec.control_point[i].
cpxcoord]
                        -datarec.control_point[i.].cphead);
        If (diff>datarec.control_point[i].cph_range) then begin
          i:=datarec.no_dewatering_cntrl_points;
          successful:=false;
        end;  {if}
    end;  {for}
    If successful then  success(time);
  end;  {begin}


  Procedure Itmax_Exceeded;
  Const
    sc:integer=5;
    sr:integer=2;
    bw:integer=70;
    bh:integer=25;
  Begin
    RestoreCrtMode;
    ClrScr;
    Box(sc,sr,bw,bh);
    Putat(20,10,'Maximum Number of Iterations Has Been Exceeded!');
    Putat(20,14,'Finite Difference Approximations have Converged Unsuccessfully'
);
    Putat(20,18,'Try first using another value for Delta or Error');
    Exit_FD:=true;
    Istep:=datarec.nsteps;
    Exit_Search:=true;  {.....and go to main menu...}
    ch := readkey;
  End;



Begin
  delta := datarec.delta;
  Time  := 0.0;
  Exit_fd :=false;
  Case datarec.units[1] of
    'I','i' : Begin
                storagefctr := 6.23;
              end;
    'U','u' : Begin
                storagefctr := 7.48;
              end;
    'M','m' : Begin
                storagefctr := 1.0;
              end;
  end;  {case}

  Delxsq:=datarec.delxy*datarec.delxy*storagefctr;

  For i := 1 to datarec.nc do
      for j := 1 to datarec.nr do begin
        ho^[i,j] := h^[i,j];
        Q^[i,j]  := Leakance^[i,j];
        tx^[i,j] := t^[i,j];
        if anisotropic then ty^[i,j] := datarec.anfctr * t^[i,j]
                        else ty^[i,j] := t^[i,j];
  End;
```

```
For i := 1 to datarec.no_pumping_wells do begin
   q^[datarec.wellno[i].y,datarec.wellno[i].x] :=
       datarec.wellno[i].well_pump_rate;
End;


If datarec.confined then
  For i:=1 to datarec.nc do
    for j:=1 to datarec.nr do
      if datarec.leaky_artesian_conditions then
        rlac^[i,j]:=aqtdperm^[i,j]/aqtdthickness^[i,j];


If not datarec.confined then
  for i:=1 to datarec.nc do
    for j:=1 to datarec.nr do begin
      If (h^[i,j]<aquiferbase^[i,j]) then
                    h^[i,j]:=aquiferbase^[i,j] + 0.01;
      Tx^[i,j]:=aqfrperm^[i,j]*sqrt((H^[i,j]-aquiferbase^[i,j])
                    *(H^[i+1,j]-aquiferbase^[i+1,j]));
      Ty^[i,j]:=aqfrperm^[i,j]*sqrt((H^[i,j]-aquiferbase^[i,j])
                    *(H^[i,j+1]-aquiferbase^[i,j+1]));
      If datarec.induced_infiltration then
        iir^[i,j]:=(waterbodybedperm^[i,j]/waterbodybedthk^[i,j])
                               *waterbodyarea^[i,j];
    End;


for i:=1 to datarec.nc do begin
  for j:=1 to datarec.nr do begin
    If datarec.leaky_artesian_conditions then begin
      ql^[i,j]:=rlac^[i,j]*(source_unit_head^[i,j]-H^[i,j]);
    end;
    if datarec.induced_infiltration then begin
      if (h^[i,j]>waterbodybedelev^[i,j]) then
      qi^[i,j]:=iir^[i,j]*(waterbodysurfelev^[i,j]-h^[i,j]);
      if (h^[i,j]<waterbodybedelev^[i,j]) then
      qi^[i,j]:=iir^[i,j]*(waterbodysurfelev^[i,j]-waterbodybedelev^[i,j]);
    end;
  end;
end;

For i:= 1 to datarec.nc do begin
   for j := 1 to datarec.nr do begin
      Hinitial^[i,j] := h^[i,j];
   end;
end;

For istep:=1 to datarec.nsteps do begin

  {Preliminary Head Predictor}
  For hi:=1 to datarec.nc do begin
```

```
    For hj:=1 to datarec.nr do begin
      Hd:=h^[hi,hj]-ho^[hi,hj];
      ho^[hi,hj]:=h^[hi,hj];
      F:=1.0;
      If (lhd^[hi,hj]<>0.0) then
        Begin
          If (istep>2) then F:=Hd/lhd^[hi,hj];
          If (F>5.0) then F:=5.0;
          If (F<0.0) then F:=0.0;
        End;
      lhd^[hi,hj]:=Hd;
      h^[hi,hj]:=h^[hi,hj]+Hd*F;
    End; {hj}
  End; {hi}

  {Refine Head Estimates}
  Time:=Time+delta;       {days+seconds*day/seconds}
  iter:=0;
  Convergence:=false;


  Repeat
    E:= 0.0;
    iter:=iter + 1;
    {Column Calculations}
    For nci:=1 to datarec.nc do begin
      {Initialize B & G Arrays}
      For BGj:=1 to datarec.nr do begin
        B[bgj]:=0.0; G[bgj]:=0.0;
      end;
      ci:=nci;
      Sum:=istep+iter;
      If((Sum MOD 2)=1) then ci:=datarec.nc-ci+1;
      For cj:=1 to datarec.nr do begin
        {calculate b & g arrays}
        ac:=0.0;
        bc:=Stor^[ci,cj]*Delxsq/delta;
        cc:=0.0;
        dc:=ho^[ci,cj]*Stor^[ci,cj]*Delxsq/delta-q^[ci,cj];

           datarec.leaky_artesian_conditions then begin
           bc := bc+rlac^[ci,cj];
           dc := dc+(rlac^[ci,cj]*source_unit_head^[ci,cj]);
        End;  {leaky conditions}
        If datarec.induced_infiltration then begin
           bc := bc+iir^[ci,cj];
           dc := dc+(iir^[ci,cj]*waterbodysurfelev^[ci,cj]);
        End;  {induced infiltration}

        If ((cj-1)<>0) then
          begin
            ac:=-tx^[ci,cj-1];
            bc:=bc+tx^[ci,cj-1];
          end;  {if}
        If ((cj-datarec.nr)<>0) then
```

```
     begin
       bc:=bc+tx^[ci,cj];
       cc:=-tx^[ci,cj];
     end;  {if}
    If ((ci-1)<>0) then
     begin
       bc:=bc+ty^[ci-1,cj];
       dc:=dc+h^[ci-1,cj]*ty^[ci-1,cj];
     end;  {if}
    If ((ci-datarec.nc)<>0) then
     begin
       bc:=bc+ty^[ci,cj];
       dc:=dc+h^[ci+1,cj]*ty^[ci,cj];
     end;  {if}
    wc:=uc-ac*B[cj-1];
    B[cj]:=cc/wc;
    G[cj]:=(dc-ac*G[cj-1])/wc;
  End; {cj}
  {Re-estimate Heads}
  E:=E+abs(h^[ci,datarec.nr]-G[datarec.nr]);
  h^[ci,datarec.nr]:=G[datarec.nr];
  nj:=datarec.nr-1;
  While (nj<>0) do begin
    HA:=G[nj]-B[nj]*h^[ci,nj+1];
    E:=E+abs(HA-h^[ci,nj]);
    h^[ci,nj]:=Ha;
    nj:=nj-1;
  End; {While}
End; {ci}
{Row Calculations}
For nrj:=1 to datarec.nr do begin
  {Initialize B & G Arrays}
  For bgi:=1 to datarec.nc do begin
    B[bgi]:=0.0; G[bgi]:=0.0;
  End; {bgi}
  rj:=nrj;
  If((sum MOD 2) =1) then rj:=datarec.nr-rj+1;
  For ri:=1 to datarec.nc do begin
    {Calculate B & G Arrays}
    ar:=0.0;
    br:=Stor^[ri,rj]*Delxsq/delta;
    cr:=0.0;
    dr:=ho^[ri,rj]*Stor^[ri,rj]*Delxsq/delta-q^[ri,rj];

    If datarec.leaky_artesian_conditions then begin
       br := br+rlac^[ri,rj];
       dr := dr+(rlac^[ri,rj]*source_unit_head^[ri,rj]);
    End;  {leaky conditions}
    If datarec.induced_infiltration then begin
       br := br+iir^[ri,rj];
       dr := dr+(iir^[ri,rj]*waterbodysurfelev^[ri,rj]);
    End;  {induced infiltration}

    If ((rj-1)<>0) then
      begin
```

```
              br:=br+tx^[ri,rj-1];
              dr:=dr+h^[ri,rj-1]*tx^[ri,rj-1];
             end; {if}
           If ((rj-datarec.nr)<>0) then
             bagin
               br:=br+tx^[ri,rj];
               dr:=dr+h^[ri,rj+1]*tx^[ri,rj];
             end; {if}
           If ((ri-1)<>0) then
             begin
               ar:=-ty^[ri-1,rj];
               br:=br+ty^[ri-1,rj];
             end; {if}
           If ((ri-datarec.nc)<>0) then
             begin
               br:=br+ty^[ri,rj];
               cr:=-ty^[ri,rj];
             end; {if}
           wr:=hr-ar*B[ri-1];
           B[ri]:=cr/wr;
           G[ri]:=(dr-ar*G[ri-1])/wr;
         End; {ri}
         E:=E+abs(h^[datarec.nc,rj]-G[datarec.nc]);
         h^[datarec.nc,rj]:=G[datarec.nc];
         ni:=datarec.nc-1;
         While (ni<>0) do begin
           HA:=G[ni]-B[ni]*h^[ni+1,rj];
           E:=E+abs(h^[ni,rj]-HA);
           h^[ni,rj]:=HA;
           ni:=ni-1;
         End; {While}
       End; {rj}

       {Water Balance}
       Store:=0.0; Pump:=0.0;

       for i:=1 to datarec.nc do begin
         for j:=1 to datarec.nr do begin
           If datarec.leaky_artesian_conditions then begin
              ql^[i,j]:=rlac^[i,j]*(source_unit_head^[i,j]-H^[i,j]);
           end;
           If datarec.induced_infiltration then begin
           If (h^[i,j]>waterbodybedelev^[i,j]) then
              qi^[i,j]:=iir^[i,j]*(waterbodysurfelev^[i,j]-h^[i,j]);
           If (h^[i,j]<waterbodybedelev^[i,j]) then
              qi^[i,j]:=iir^[i,j]*(waterbodysurfelev^[i,j]-waterbodybedelev^[i,j
]);
           end;
         end;
       end;


       For wi:=1 to datarec.nc do begin
         For wj:=1 to datarec.nr do begin
           Store:=Store+Stor^[wi,wj]*Delxsq*(ho^[wi,wj]-h^[wi,wj])/delta;
```

```
          Pump:=Pump+q^[wi,wj];
          If datarec.leaky_artesian_conditions then
             store := store+ql^[wi,wj];
          If datarec.induced_infiltration then
             store := store+qi^[wi,wj];
        End; (wj)
      End;  (wi)


      Totin:=Store;
      Totout:=Pump;
      Wdiff:=abs(1.0-Totout/Totin)*100;
      If(Iter>Itmax) then Itmax_Exceeded;
      If (totout<>0.0) and (Wdiff>5.0) then Convergence:=false
      Else If (E>datarec.error) then Convergence:=false
      Else If (Iter<Itmin) then Convergence :=false
      Else Convergence:=True;
   Until Convergence or Exit_FD;

   Year:=Time/365;
   delta:=delta*1.2;

   Print_head_map(delta);

   for i := 1 to datarec.no_pumping_wells do begin
      Amount_dewatered := h^[datarec.wellno[i].x,datarec.wellno[i].y]/
          hinitial^[datarec.wellno[i].x,datarec.wellno[i].y];
      If (amount_dewatered < datarec.wellno[i].percent_dewater) then begin
             istep := datarec.nsteps;
             well_dry := true;
      end;
   end;

   Percent_step := istep/datarec.nsteps;
   If datarec.plumeprob and (percent_step>0.10) then begin
      Append(poutfile);
      writeln(poutfile);

      For i := 1 to datarec.no_pumping_wells do begin
          Writeln(poutfile,'Well Number ',i);
          Writeln(poutfile,'Pumping rate = ',datarec.wellno[i].well pump_rate:1
0:2);
          writeln(poutfile,'Grid coordinates for well  ',i,' ( '
                ,datarec.wellno[i].maxxl,' , ',datarec.wellno[i].maxyl,' )');
          If (datarec.wellno[i].well_pump_rate>0.0) then begin
             writeln(poutfile,'Capture width in the X direction = ',datarec.wel
lno[i] maxxrad);
             writeln(poutfile,'Capture width in the Y direction = ',datarec.wel
lno[i].maxyrad);
          End;
          writeln(poutfile);
      End;
      Writeln(poutfile,'The following is the solution head map');
      Writeln(poutfile,'achieved after a simulated pumping period of ',time:4:1
,' days');
```

```
       Writeln(poutfile);
       for i := 1 to datarec.nc do begin
          for j := 1 to datarec.nr do
             write(poutfile,h^[i,j]:7:2);
          writeln(poutfile);
       end;
       writeln(poutfile);
       close(poutfile);
    end;

    If datarec.dewateringprob then compare_control_points(delta);

  End; {istep}


End; {finite_approx}



 Procedure Drawellipse(xc,yc,xr,yr : real);
  begin
     setcolor(0);    {to draw black well}
     fillellipse(trunc(xc),trunc(yc),trunc(xr),trunc(yr));
     setcolor(1);    {turn pixels on to redraw image}
  end;


  Procedure Capture(i:integer);
  var
    radx,rady             : real;
    xcenter,ycenter       : real;
    converted_pump_rate   : real;
    ch                    : char;
    ii,jj                 : integer;
  Begin

      If (datarec.wellno[i].well_pump_rate>0.0) then begin

        Txavg_up:=(T^[datarec.wellno[i].x,datarec.wellno[i].y]
                  +T^[datarec.wellno[i].x-1,datarec.wellno[i].y]
                  +T^[datarec.wellno[i].x-2,datarec.wellno[i].y])/3.0;
        Txavg_down:=(T^[datarec.wellno[i].x,datarec.wellno[i].y]
                  +T^[datarec.wellno[i].x+1,datarec.wellno[i].y]
                  +T^[datarec.wellno[i].x+2,datarec.wellno[i].y])/3.0;
        Tyavg_right:=(T^[datarec.wellno[i].x,datarec.wellno[i].y]
                  +T^[datarec.wellno[i].x,datarec.wellno[i].y+1]
                  +T^[datarec.wellno[i].x,datarec.wellno[i].y+2])/3.0;
        Tyavg_left:=(T^[datarec.wellno[i].x,datarec.wellno[i].y]
                  +T^[datarec.wellno[i].x,datarec.wellno[i].y-1]
                  +T^[datarec.wellno[i].x,datarec.wellno[i].y-2])/3.0;

        If (txavg_up >= txavg_down) then txavg:=txavg_down;
        If (txavg_up <= txavg_down) then txavg:=txavg_up;
        If (tyavg_right >= tyavg_left) then tyavg:=tyavg_left;
        If (tyavg_right <= tyavg_left) then tyavg:=tyavg_right;
        Rx:=3.14*((datarec.wellno[i].well_pump_rate)/(Txavg*(6.28)*datarec.avera
ge_site_gradient));
        Ry:=3.14*((datarec.wellno[i].well_pump_rate)/(Tyavg*(6.28)*datarec.avera
ge_site_gradient));

        datarec.wellno[i].xrad:=rx/datarec.delxy;
        datarec.wellno[i].yrad:=ry/datarec.delxy;

        xcenter:=vspace*(datarec.wellno[i].x-1)+30;
        ycenter:=hspace*(datarec.wellno[i].y-1)+38;

        radx:=datarec.wellno[i].xrad*vspace;
        rady:=datarec.wellno[i].yrad*hspace;

        datarec.wellno[i].xradius := rx;
        datarec.wellno[i].yradius := ry;

        Drawellipse(xcenter,ycenter,radx,rady);
      end; {if}
  end; {begin}


Procedure Put_well(x:integer);
Var
    a : array [1..5,1..2] of integer;
    y,z : integer;
    radx,rady : integer;
Begin
    Radx:=trunc(vspace*(datarec.wellno[x].x-1))+30;
    Rady:=trunc(hspace*(datarec.wellno[x].y-1))+38;
    datarec.wellno[x].well_dot[1,1]:=radx+4;
    datarec.wellno[x].well_dot[1,2]:=rady;
    datarec.wellno[x].well_dot[2,1]:=radx;
    datarec.wellno[x].well_dot[2,2]:=rady-4;
    datarec.wellno[x].well_dot[3,1]:=radx-4;
    datarec.wellno[x].well_dot[3,2]:=rady;
    datarec.wellno[x].well_dot[4,1]:=radx;
    datarec.wellno[x].well_dot[4,2]:=rady+4;
    datarec.wellno[x].well_dot[5,1]:=radx+4;
    datarec.wellno[x].well_dot[5,2]:=rady;
    for y := 1 to 5 do
       for z := 1 to 2 do
          a[y,z] := datarec.wellno[x].well_dot[y,z];
    Setfillstyle(1,1);
    Setcolor(1);
    Drawpoly(5,a);
    Fillpoly(5,a);
End;  {procedure}


Procedure Set_Well(n,a,b:integer);
Begin
  datarec.wellno[n].x:=A;
  datarec.wellno[n].y:=B;
```

```
End;   (procedure)


Procedure Search(n:integer);
Var
  x,y,xx,yy    : integer;
  ends         : boolean;
  ch           : char;
  alldiff      : boolean;
Begin
   ends       := false;

   for x := 1 to n do xc[x] := datarec.wellno[x].tlx;
   for x := 1 to n do yc[x] := datarec.wellno[x].tly;
   while not ends do begin
      alldiff := true;
      if n>1 then begin
         for x := 1 to n-1 do
            for y := x+1 to n do
               if (xc[x]=xc[y]) and (yc[x]=yc[y]) then
                  alldiff := false;
      end;
      if alldiff then begin
         Exit_search:=false;
         While not exit_search do begin
            for x := 1 to n do begin
               for xx := 1 to datarec.nc do
                  for yy := 1 to datarec.nr do
                     H^[xx,yy] := Hprime^[xx,yy];
               set_well(x,xc[x],yc[x]);
               Put_well(x);
               If datarec.Plumeprob then Capture(x);
            end;

            if datarec.dewateringprob then Finite_Difference_Approximation;
            If datarec.plumeprob then assess_plume_coverage;

            Erase_well;
            xc[1] := xc[1] + 1;
            for x := 1 to n-1 do begin
               if xc[x]>datarec.wellno[x].trx then begin
                  xc[x] := datarec.wellno[x].tlx;
                  yc[x] := yc[x] + 1;
               end;
               if yc[x]>datarec.wellno[x].bry then begin
                  yc[x] := datarec.wellno[x].tly;
                  xc[x] := datarec.wellno[x].tlx;
                  xc[x+1] := xc[x+1] + 1;
               end;
            end;
            if xc[n]>datarec.wellno[n].trx then begin
               xc[n] := datarec.wellno[n].tlx;
               yc[n] := yc[n] + 1;
            end;
            if yc[n]>datarec.wellno[n].bry then begin
```

```
               ends := true;
               exit_search := true;
            end;
         end;  {while}
      end;  {if}
   end;
   if datarec.plumeprob then
      create_plume_solution_summary;
end;  {search}
```

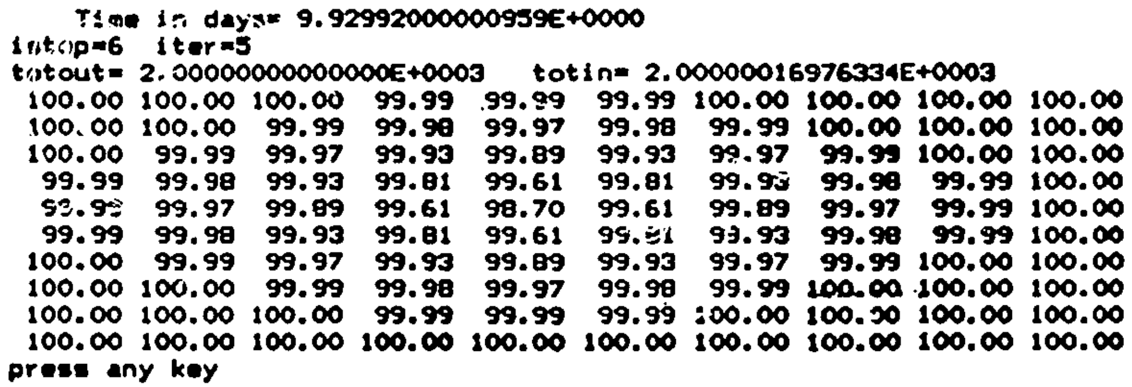
```
Procedure put_plume_area(Hspace,Vspace,nplume:Integer);
Var
  Plume_Coord : Array[1..5,1..2] of Integer;
  i           : integer;
Begin
  For i:=1 to nplume do begin
    Plume_Coord[1,1]:=trunc((datarec.plume[i].ptlx-1)*Vspace)+30;
    Plume_Coord[1,2]:=trunc((datarec.plume[i].ptly-1)*Hspace)+38;
    Plume_Coord[2,1]:=trunc((datarec.plume[i].ptrx-1)*Vspace)+30;
    Plume_Coord[2,2]:=trunc((datarec.plume[i].ptry-1)*Hspace)+38;
    Plume_Coord[3,1]:=trunc((datarec.plume[i].pbrx-1)*Vspace)+30;
    Plume_Coord[3,2]:=trunc((datarec.plume[i].pbry-1)*Hspace)+38;
    Plume_Coord[4,1]:=trunc((datarec.plume[i].pblx-1)*Vspace)+30;
    Plume_Coord[4,2]:=trunc((datarec.plume[i].pbly-1)*Hspace)+38;
    Plume_Coord[5,1]:=trunc((datarec.plume[i].ptlx-1)*Vspace)+30;
    Plume_Coord[5,2]:=trunc((datarec.plume[i].ptly-1)*Hspace)+38;
    SetFillStyle(6,1);
    SetColor(1);
    DrawPoly(5,Plume_Coord);
    FillPoly(5,Plume_Coord);
  End;  {for}
End;
```

```
Procedure put_search_areas(Hspace,Vspace,no_pumping_wells:Integer);
Var
  Search_Coord:Array[1..5,1..2] of Integer;
  i : integer;
Begin
  for i := 1 to no_pumping_wells do begin
    Search_Coord[1,1]:=trunc((datarec.wellno[i].tlx-1)*Vspace)+30;
    Search_Coord[1.2]:=trunc((datarec.wellno[i].tly-1)*Hspace)+38;
    Search_Coord[2,1]:=trunc((datarec.wellno[i].trx-1)*Vspace)+30;
    Search_Coord[2,2]:=trunc((datarec.wellno[i].try-1)*Hspace)+38;
    Search_Coord[3,1]:=trunc((datarec.wellno[i].brx-1)*Vspace)+30;
    Search_Coord[3,2]:=trunc((datarec.wellno[i].bry-1)*Hspace)+38;
    Search_Coord[4,1]:=trunc((datarec.wellno[i].blx-1)*Vspace)+30;
    Search_Coord[4,2]:=trunc((datarec.wellno[i].bly-1)*Hspace)+38;
    Search_Coord[5,1]:=trunc((datarec.wellno[i].tlx-1)*Vspace)+30;
    Search_Coord[5,2]:=trunc((datarec.wellno[i].tly-1)*Hspace)+38;
```

```
     SetFillStyle(6,1);
     SetColor(1);
     DrawPoly(5,Search_Coord);
     FillPoly(5,Search_Coord);
   end;
 End;


Procedure Draw_Search_Grid(nc,nr:integer);
Const
   gd:integer=0;
   Clipon=true;
   clipoff=false;
   gm:integer=2;
   i:integer=1;
   j:integer=1;
   ny:integer=30;
   gh:string[40]='* OPTIMUM WELL SEARCH IN PROGRESS *';
   gs:string[35]=' ';
Var
   n,x:integer;
   m : integer;
   ch : char;
   Sum_vert,Sum_horz:integer;
   Window1,Window2:Windowptr;
   pc : longint;
Begin
   Initgraph(gd,gm,'c:\tp');
   SetViewPort(0,0,639,199,clipon);
   getaspectratio(xasp,yasp);
   SetTextStyle(1,0,1);
   SetTextJustify(1,0);
   SetLineStyle(0,0,3);
   Rectangle(0,0,639,199);
   SetLineStyle(0,0,1);
   Moveto(320,15);
   OutText(gh);
   SetTextStyle(1,0,1);
   MoveTo(320,30);
   OutText(gs);
   N:=datarec.nc-2;
   m:=datarec.nr-2;
   Hspace:=(196-38)div(n+1);
   Vspace:=(600-30)div(m+1);
   Sum_Horz:=trunc((m+1)*Vspace)+30;
   Sum_Vert:=trunc((N+1)*Hspace)+38;
   For x:=1 to n+2 do
      Line(30, (x-1)*trunc(hspace)+38,Sum_Horz,(x-1)*trunc(hspace)+38);
   For x:=1 to m+2 do
      Line((x-1)*trunc(vspace)+30,38,(x-1)*trunc(Vspace)+30,Sum_Vert);
   If datarec.Plumeprob then
      Put_Plume_Area(trunc(hspace),trunc(vspace),datarec.nplume);
   If datarec.dewateringprob then
      Put_Search_Areas(trunc(hspace),trunc(vspace),datarec.no_pumping_wells);
   getimage(0,0,getmaxx,getmaxy,wellpntr^);
   datarec.maxconc := 0.0;
   If datarec.dewateringprob then begin
      Assign(doutfile,sfilename+'.OUT');
      Rewrite(doutfile);
      Writeln(doutfile,'*** Optimum Dewatering Solution Summary ***');
      Writeln(doutfile);
      Close(doutfile);
   end;
   If datarec.plumeprob then begin
      Assign(poutfile,sfilename+'.OUT');
      Rewrite(poutfile);
      Writeln(poutfile,'*** Optimum Plume Control Solution Summary ***');
      Writeln(poutfile);
      Close(poutfile);
   end;
   Search(datarec.no_pumping_wells);
   If datarec.dewateringprob then
      Create_dewatering_solution_summary;
   Closegraph;
End;


End.
```

Appendix B

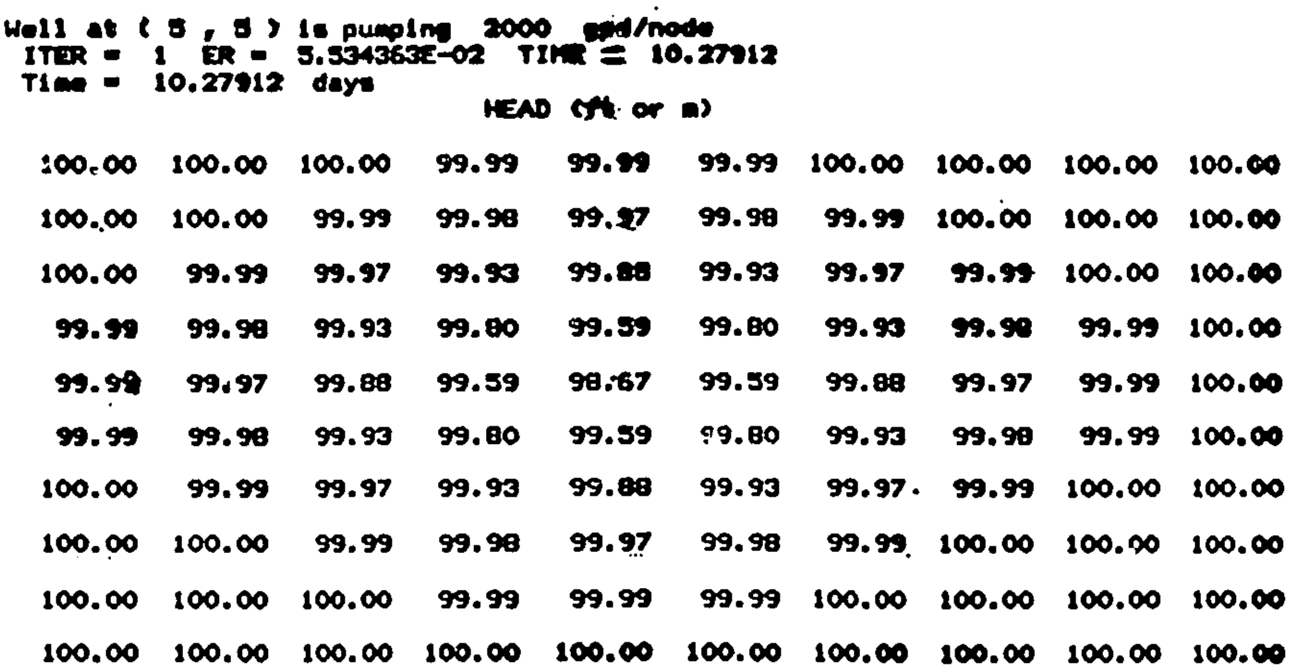
Finite Difference Program Output



```
    Time in days= 9.92992000000959E+0000
intop=6  iter=5
totout= 2.00000000000000E+0003   totin= 2.00000016976334E+0003
 100.00 100.00 100.00  99.99  99.99  99.99 100.00 100.00 100.00 100.00
 100.00 100.00  99.99  99.98  99.97  99.98  99.99 100.00 100.00 100.00
 100.00  99.99  99.97  99.93  99.89  99.93  99.97  99.99 100.00 100.00
  99.99  99.98  99.93  99.81  99.61  99.81  99.93  99.98  99.99 100.00
  99.99  99.97  99.89  99.61  98.70  99.61  99.89  99.97  99.99 100.00
  99.99  99.98  99.93  99.81  99.61  99.81  99.93  99.98  99.99 100.00
 100.00  99.99  99.97  99.93  99.89  99.93  99.97  99.99 100.00 100.00
 100.00 100.00  99.99  99.98  99.97  99.98  99.99 100.00 100.00 100.00
 100.00 100.00 100.00  99.99  99.99  99.99 100.00 100.00 100.00 100.00
 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00
press any key
```

Figure B.1

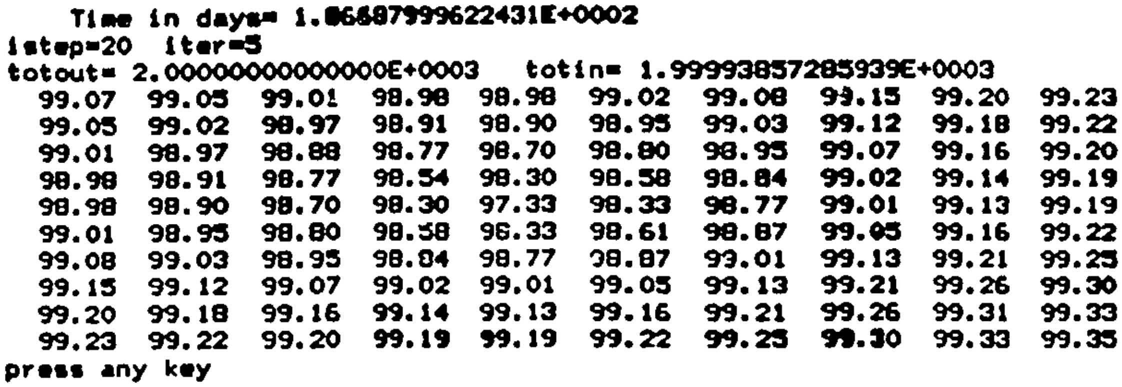
Potentiometric surface calculated by thesis Model after a simulated pumping Period of 9.9 days.



```
Well at ( 5 , 5 ) is pumping  2000  gpd/node
 ITER =  1  ER =  5.534363E-02  TIMR = 10.27912
 Time =  10.27912  days
                                          HEAD (ft or m)

  100.00  100.00  100.00   99.99   99.99   99.99  100.00  100.00  100.00  100.00

  100.00  100.00   99.99   99.98   99.97   99.98   99.99  100.00  100.00  100.00

  100.00   99.99   99.97   99.93   99.88   99.93   99.97   99.99  100.00  100.00

   99.99   99.98   99.93   99.80   99.59   99.80   99.93   99.98   99.99  100.00

   99.99   99.97   99.88   99.59   98.67   99.59   99.88   99.97   99.99  100.00

   99.99   99.98   99.93   99.80   99.59   99.80   99.93   99.98   99.99  100.00

  100.00   99.99   99.97   99.93   99.88   99.93   99.97   99.99  100.00  100.00

  100.00  100.00   99.99   99.98   99.97   99.98   99.99  100.00  100.00  100.00

  100.00  100.00  100.00   99.99   99.99   99.99  100.00  100.00  100.00  100.00

  100.00  100.00  100.00  100.00  100.00  100.00  100.00  100.00  100.00  100.00
```

Figure B.2

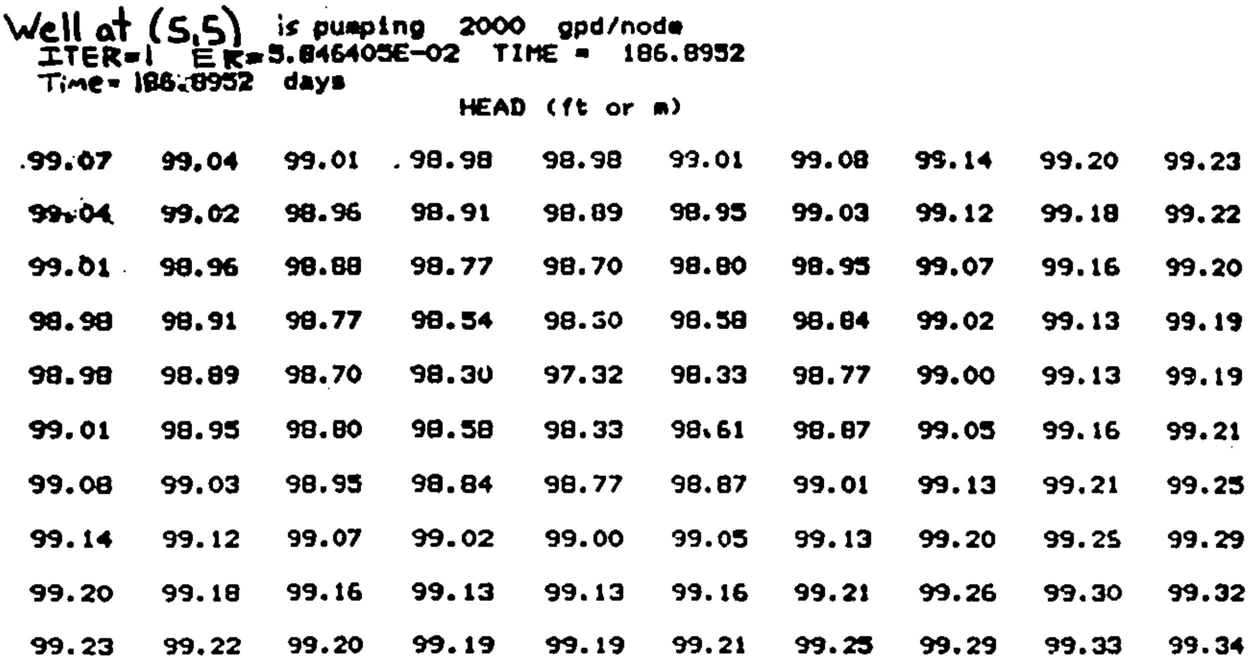
Potentiometric surface calculated by PLASMER4 Model after a simulated pumping Period of 10.3 days.



```
   Time in days= 1.86687999622431E+0002
istep=20  iter=5
totout= 2.00000000000000E+0003   totin= 1.99993857285939E+0003
  99.07  99.05  99.01  98.98  98.98  99.02  99.08  99.15  99.20  99.23
  99.05  99.02  98.97  98.91  98.90  98.95  99.03  99.12  99.18  99.22
  99.01  98.97  98.88  98.77  98.70  98.80  98.95  99.07  99.16  99.20
  98.98  98.91  98.77  98.54  98.30  98.58  98.84  99.02  99.14  99.19
  98.98  98.90  98.70  98.30  97.33  98.33  98.77  99.01  99.13  99.19
  99.01  98.95  98.80  98.58  96.33  98.61  98.87  99.05  99.16  99.22
  99.08  99.03  98.95  98.84  98.77  98.87  99.01  99.13  99.21  99.25
  99.15  99.12  99.07  99.02  99.01  99.05  99.13  99.21  99.26  99.30
  99.20  99.18  99.16  99.14  99.13  99.16  99.21  99.26  99.31  99.33
  99.23  99.22  99.20  99.19  99.19  99.22  99.25  99.30  99.33  99.35
press any key
```

Figure B.3

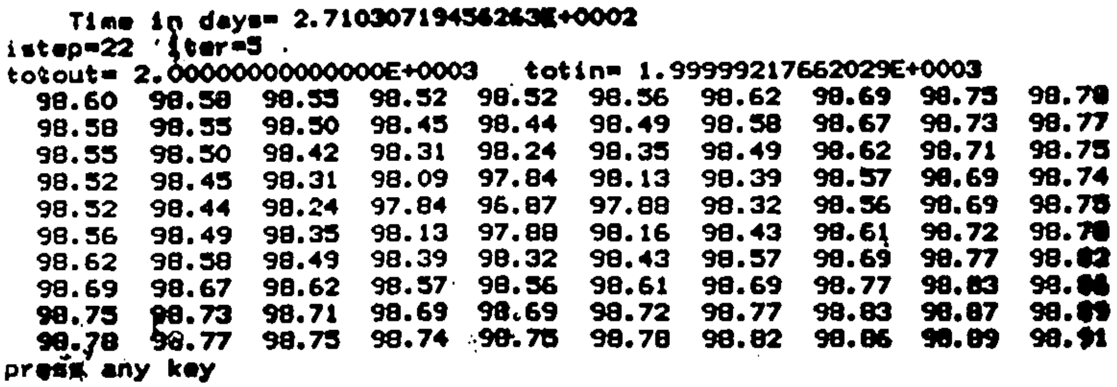
Potentiometric surface calculated by thesis Model after a simulated pumping Period of 186.7 days.



```
Well at (5,5) is pumping  2000  gpd/node
  ITER=1  ER=5.846405E-02  TIME =  186.8952
  Time= 186.8952  days
                                          HEAD (ft or m)

 99.07    99,04    99.01    98.98    98.98    99.01    99.08    99.14    99.20    99.23

 99.04    99.02    98.96    98.91    98.89    98.95    99.03    99.12    99.18    99.22

 99.01    98.96    98.88    98.77    98.70    98.80    98.95    99.07    99.16    99.20

 98.98    98.91    98.77    98.54    98.30    98.58    98.84    99.02    99.13    99.19

 98.98    98.89    98.70    98.30    97.32    98.33    98.77    99.00    99.13    99.19

 99.01    98.95    98.80    98.58    98.33    98.61    98.87    99.05    99.16    99.21

 99.08    99.03    98.95    98.84    98.77    98.87    99.01    99.13    99.21    99.25

 99.14    99.12    99.07    99.02    99.00    99.05    99.13    99.20    99.25    99.29

 99.20    99.18    99.16    99.13    99.13    99.16    99.21    99.26    99.30    99.32

 99.23    99.22    99.20    99.19    99.19    99.21    99.25    99.29    99.33    99.34
```

Figure B.4

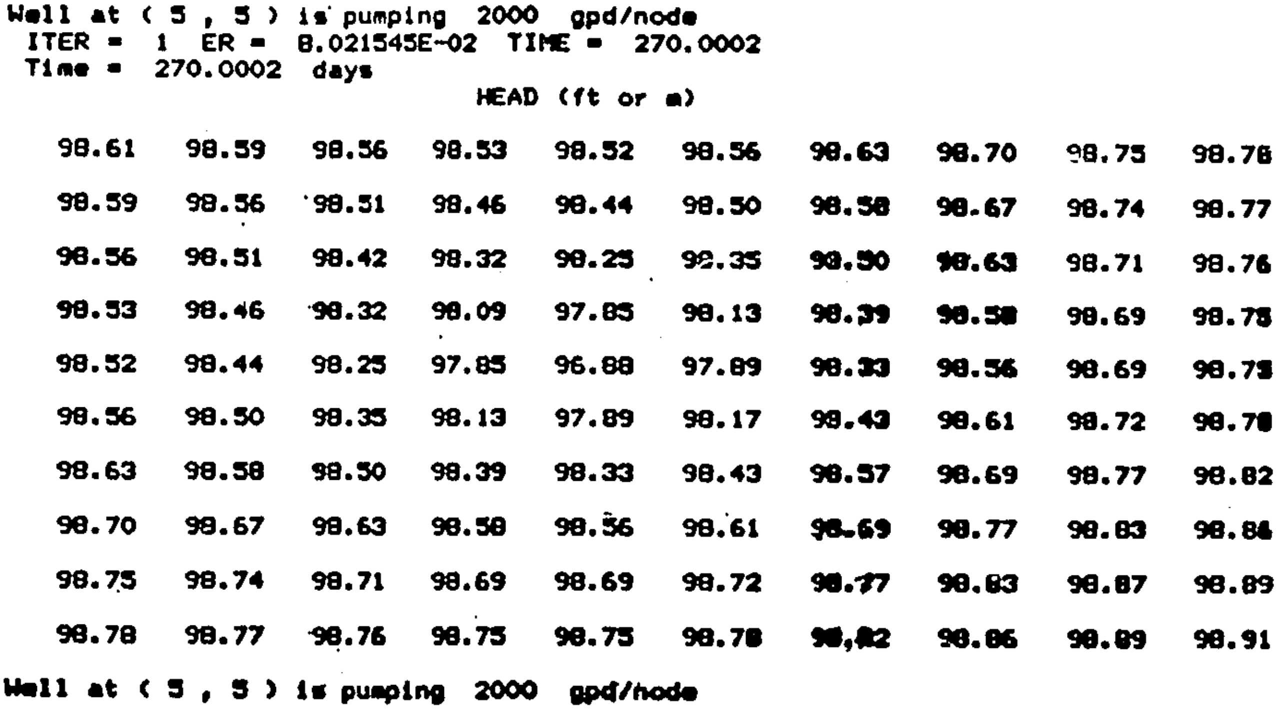
Potentiometric surface calculated by PLASMER4 Model after a simulated pumping Period of 186.9 days.



```
   Time in days= 2.71030719456263E+0002
istep=22 'iter=5 .
totout= 2.00000000000000E+0003   totin= 1.99999217662029E+0003
  98.60  98.58  98.55  98.52  98.52  98.56  98.62  98.69  98.75  98.78
  98.58  98.55  98.50  98.45  98.44  98.49  98.58  98.67  98.73  98.77
  98.55  98.50  98.42  98.31  98.24  98.35  98.49  98.62  98.71  98.75
  98.52  98.45  98.31  98.09  97.84  98.13  98.39  98.57  98.69  98.74
  98.52  98.44  98.24  97.84  96.87  97.88  98.32  98.56  98.69  98.75
  98.56  98.49  98.35  98.13  97.88  98.16  98.43  98.61  98.72  98.78
  98.62  98.58  98.49  98.39  98.32  98.43  98.57  98.69  98.77  98.82
  98.69  98.67  98.62  98.57  98.56  98.61  98.69  98.77  98.83  98.86
  98.75  98.73  98.71  98.69  98.69  98.72  98.77  98.83  98.87  98.89
  98.78  98.77  98.75  98.74  98.75  98.78  98.82  98.86  98.89  98.91
press any key
```

Figure B.5

Potentiometric surface calculated by thesis Model after a simulated pumping Period of 271 days.



```
Well at ( 5 , 5 ) is pumping  2000  gpd/node
 ITER =  1  ER =  8.021545E-02  TIME =  270.0002
 Time =  270.0002  days
                                        HEAD (ft or m)

   98.61    98.59    98.56    98.53    98.52    98.56    98.63    98.70    98.75    98.78

   98.59    98.56    98.51    98.46    98.44    98.50    98.58    98.67    98.74    98.77

   98.56    98.51    98.42    98.32    98.25    98.35    98.50    98.63    98.71    98.76

   98.53    98.46    98.32    98.09    97.85    98.13    98.39    98.58    98.69    98.75

   98.52    98.44    98.25    97.85    96.88    97.89    98.33    98.56    98.69    98.75

   98.56    98.50    98.35    98.13    97.89    98.17    98.43    98.61    98.72    98.78

   98.63    98.58    98.50    98.39    98.33    98.43    98.57    98.69    98.77    98.82

   98.70    98.67    98.63    98.58    98.56    98.61    98.69    98.77    98.83    98.84

   98.75    98.74    98.71    98.69    98.69    98.72    98.77    98.83    98.87    98.89

   98.78    98.77    98.76    98.75    98.75    98.78    98,82    98.86    98.89    98.91
Well at ( 5 , 5 ) is pumping  2000  gpd/node
```

Figure B.6

Potentiometric surface calculated by PLASMER4 Model after a simulated pumping Period of 270 days.

Appendix C

Dewatering Case Study Summary

Welland Canal Dewatering Case Study Application Summary

C.1 Introduction

Realignment of the Welland Canal between Port Robinson and Port Colborne necessitated both temporary and permanent depressurizing of an underlying aquifer (Farvolden and Nunan, 1970). The aquifer is a thin zone of fractured dolomite underlying approximately 60 to 100 ft of poorly-permeable glacial till and lacustrine sediments (Farvolden and Nunan, 1970). The confining layer permits recharge and discharge through leakage to the aquifer (Frind, 1970). In designing a dewatering system for the construction project, it became evident that one of the prime considerations would be the regional impact the dewatering operations would have upon potable water supplies of nearby residents.

C.2 Hydrogeologic Aquifer Testing

The first considerations related to the dewatering project involved determining the aquifer's hydrogeologic coefficients and boundary conditions for design of both the de-watering and monitoring systems, as well as predicting the spread of the drawdown cone (Farvolden and Nunan, 1970). Based upon the results of four aquifer pump tests, it was found that a high degree of variability existed within the aquifer. Transmissivity of the aquifer ranged in value from 2000 to 90 000 gal/day. This high degree of variability was attributed to zones of fracture where solution of the dolomite had taken place.

C.3 Regional Aquifer Response

Heavy pumping in connection with excavation was started before any attempt could be made to analyze the pump test data for regional, long term response. Within a few days, there were reports of interference from owners of private wells and it became urgent to recognize valid claims, and foresee future ones (Farvolden and Nunan, 1970). Subsequent to analyzing the pump test data, it was predicted that portions of the regional aquifer would be dewatered by as much as 20 feet.

C.4 Theoretical Aquifer Analysis

A finite difference based computer groundwater flow program was developed (Frind, 1970) and applied to the site to predict future impacts upon the regional aquifer as a result of the dewatering operations. The model was found to match field observations closely and was determined to be a useful prediction tool.

C.5 Summary

The utilization of a computer model to predict aquifer response's was considered a useful tool at the Welland Canal Site. The model allowed the consulting hydrogeologists to assess whether the aquifer's parameters were regionally representative.

Appendix D

Dewatering Case Study Program Input

Welland Canal Dewatering Case Study Application Input Data

D.1 Finite Difference Parameters

Number of columns:		10
Number of rows:		10
Grid Spacing:		10000 ft
Finite Difference Tolerance of Convergence:	0.1 ft	
Total number of time steps:		30
Initial time step:		2 days
Anisotropy factor (Txx/Tyy)		1.0

D.2 Hydrogeologic Parameters

Aquifer:		Confined
Induced Infiltration:		False
Leaky Artesian Conditions:		False
Number of Dewatering Control Points:		9
Number of pumping wells:		7

Dewatering Control Point No.	1	
X grid coordinate of control point:		4
Y grid coordinate of control point:		8
Desired control point elevation:		570 ft
Range for calculated control point elevations:		10 ft

Dewatering Control Point No.	2	
X grid coordinate of control point:		4
Y grid coordinate of control point:		7
Desired control point elevation:		570 ft
Range for calculated control point elevations:		10 ft

Dewatering Control Point No.	3	
X grid coordinate of control point:		4
Y grid coordinate of control point:		6
Desired control point elevation:		570 ft
Range for calculated control point elevations:		10 ft

Welland Canal Dewatering Case Study Application Input Data

Dewatering Control Point No.	4	
X grid coordinate of control point:		4
Y grid coordinate of control point:		5
Desired control point elevation:		570 ft.
Range for calculated control point elevations:		10 ft.
Dewatering Control Point No.	5	
X grid coordinate of control point:		5
Y grid coordinate of control point:		4
Desired control point elevation:		560 ft.
Range for calculated control point elevations:		10 ft.
Dewatering Control Point No.	6	
X grid coordinate of control point:		6
Y grid coordinate of control point:		4
Desired control point elevation:		515 ft.
Range for calculated control point elevations:		5 ft.
Dewatering Control Point No.	7	
X grid coordinate of control point:		6
Y grid coordinate of control point:		5
Desired control point elevation:		540 ft.
Range for calculated control point elevations:		5 ft.
Dewatering Control Point No.	8	
X grid coordinate of control point:		5
Y grid coordinate of control point:		7
Desired control point elevation:		525 ft.
Range for calculated control point elevations:		5 ft.
Dewatering Control Point No.	9	
X grid coordinate of control point:		5
Y grid coordinate of control point:		8
Desired control point elevation:		555 ft.
Range for calculated control point elevations:		5 ft.

Welland Canal Dewatering Case Study Application Input Data

Pumping Well No.		1
Pumping rate:		-150000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	3	
Y grid coordinate of top left corner of search area:	7	
X grid coordinate of bottom right corner of search area:	4	
Y grid coordinate of bottom right corner of search area:	8	

Pumping Well No.		2
Pumping rate:		-150000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	3	
Y grid coordinate of top left corner of search area:	5	
X grid coordinate of bottom right corner of search area:	4	
Y grid coordinate of bottom right corner of search area:	6	

Pumping Well No.		3
Pumping rate:		-150000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	4	
Y grid coordinate of top left corner of search area:	3	
X grid coordinate of bottom right corner of search area:	5	
Y grid coordinate of bottom right corner of search area:	4	

Pumping Well No.		4
Pumping rate:		2100000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	6	
Y grid coordinate of top left corner of search area:	4	
X grid coordinate of bottom right corner of search area:	6	
Y grid coordinate of bottom right corner of search area:	4	

Pumping Well No.		5
Pumping rate:		1200000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	5	
Y grid coordinate of top left corner of search area:	7	
X grid coordinate of bottom right corner of search area:	5	
Y grid coordinate of bottom right corner of search area:	7	

Welland Canal Dewatering Case Study Application Input Data

Pumping Well No.		6
Pumping rate:		500000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	6	
Y grid coordinate of top left corner of search area:	5	
X grid coordinate of bottom right corner of search area:	6	
Y grid coordinate of bottom right corner of search area:	5	
Pumping Well No.		7
Pumping rate:		500000 gpd
Maximum permissable percent dewatering	0.25	
X grid coordinate of top left corner of search area:	5	
Y grid coordinate of top left corner of search area:	8	
X grid coordinate of bottom right corner of search area:	5	
Y grid coordinate of bottom right corner of search area:	8	

TABLE D.1
Thesis Program Data Input
Dewatering Case Study Application
Welland Canal Dewatering Project

Initial Potentiometric Surface Map*

Rows	Columns									
	1	2	3	4	5	6	7	8	9	10
1	570	570	570	570	570	560	570	570	570	570
2	570	570	580	590	570	560	570	570	570	580
3	570	570	600	590	570	560	570	570	570	580
4	570	570	580	580	570	570	570	570	580	570
5	570	570	570	570	560	570	570	570	600	580
6	570	570	570	570	570	580	580	580	604	590
7	570	570	570	570	570	580	590	580	590	590
8	570	570	560	570	570	580	580	590	570	590
9	570	570	560	570	570	580	580	580	592	580
10	570	570	560	570	575	570	586	590	590	580

* All units in feet unless otherwise expressed

TABLE D.2
Thesis Program Data Input
Dewatering Case Study Application
Welland Canal Dewatering Project

Transmissivity Surface Map*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0	0	0	0	0	0	0	0	0	0
2	0	20000	20000	20000	20000	20000	20000	20000	20000	0
3	0	20000	20000	20000	5000	5000	5000	20000	20000	0
4	0	20000	20000	20000	5000	5000	5000	20000	20000	0
5	0	20000	20000	20000	5000	5000	5000	20000	20000	0
6	0	20000	20000	5000	5000	5000	5000	20000	20000	0
7	0	20000	20000	5000	5000	5000	5000	20000	20000	0
8	0	20000	20000	5000	5000	5000	5000	20000	20000	0
9	0	20000	20000	10000	10000	10000	10000	20000	20000	0
10	0	0	0	0	0	0	0	0	0	0

* All units in gallons/ft/day unless otherwise expressed

TABLE D.3
Thesis Program Data Input
Dewatering Case Study Application
Welland Canal Dewatering Project

Storativity Surface Map*

Rows	Columns									
	1	2	3	4	5	6	7	8	9	10
1	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
2	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
3	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
4	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
5	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
6	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
7	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
8	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
9	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005
10	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005	0.005

* All values are unitless

TABLE D.4
Thesis Program Data Input
Dewatering Case Study Application
Welland Canal Dewatering Project

Leakance Map*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
2	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
3	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
4	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
5	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
6	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
7	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
8	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
9	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004
10	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004	-0.004

* units are in gallons/day/cubic foot

Appendix E

Dewatering Case Study Program Output

```
*** Optimum Dewatering Solution Summary ***

Pumping rate for well # 1 = -150000.00
Grid coordinates for well 1 ( 3 , 7 )

Pumping rate for well # 2 = -150000.00
Grid coordinates for well 2 ( 3 , 5 )

Pumping rate for well # 3 = -150000.00
Grid coordinates for well 3 ( 4 , 3 )

Pumping rate for well # 4 = 2100000.00
Grid coordinates for well 4 ( 6 , 4 )

Pumping rate for well # 5 = 1200000.00
Grid coordinates for well 5 ( 5 , 7 )

Pumping rate for well # 6 =  500000.00
Grid coordinates for well 6 ( 6 , 5 )

Pumping rate for well # 7 =  500000.00
Grid coordinates for well 7 ( 5 , 8 )

Control point 1 coordinates = (4, 8)
Control point 1 desired elevation = ( 570.00 )
Control point 1 calculated elevation = ( 565.26 )
Control point 2 coordinates = (4, 7)
Control point 2 desired elevation = ( 570.00 )
Control point 2 calculated elevation = ( 563.79 )
Control point 3 coordinates = (4, 6)
Control point 3 desired elevation = ( 570.00 )
Control point 3 calculated elevation = ( 568.74 )
Control point 4 coordinates = (4, 5)
Control point 4 desired elevation = ( 570.00 )
Control point 4 calculated elevation = ( 570.30 )
Control point 5 coordinates = (5, 4)
Control point 5 desired elevation = ( 560.00 )
Control point 5 calculated elevation = ( 557.27 )
Control point 6 coordinates = (6, 4)
Control point 6 desired elevation = ( 515.00 )
Control point 6 calculated elevation = ( 519.7[illegible] )
Control point 7 coordinates = (6, 5)
Control point 7 desired elevation = ( 540.00 )
Control point 7 calculated elevation = ( 544.83 )
Control point 8 coordinates = (5, 7)
Control point 8 desired elevation = ( 525.00 )
Control point 8 calculated elevation = ( 525.13 )
Control point 9 coordinates = (5, 8)
Control point 9 desired elevation = ( 555.00 )
Control point 9 calculated elevation = ( 559.37 )
The following is the solution head map achieved
after a simulated pumping period of 44.4 days

 570.00 570.00 570.00 570.00 570.00 560.00 570.00 570.00 570.00 570.00
 570.00 577.06 578.16 576.75 570.81 565.55 567.48 570.45 573.24 574.88
 570.00 576.58 577.59 576.32 566.66 555.97 565.05 570.37 574.13 575.60
 570.00 575.00 575.46 570.97 557.27 519.72 562.91 571.12 576.71 573.67
 570.00 573.68 576.30 570.30 562.66 544.83 568.46 575.52 585.18 584.43
 570.00 571.45 571.37 568.74 564.10 571.71 578.00 581.83 590.05 591.92
 570.00 570.65 573.09 563.79 525.13 572.17 580.72 583.74 587.54 589.16
 570.00 568.28 566.94 565.26 559.37 572.42 579.53 582.91 584.57 585.08
 570.00 567.65 566.25 567.55 568.87 574.39 580.32 583.51 584.52 583.49
 570.00 568.46 563.78 568.63 571.42 573.97 582.25 585.38 586.28 580.00


Pumping rate for well # 1 = -150000.00
Grid coordinates for well 1 ( 4 , 7 )

Pumping rate for well # 2 = -150000.00
Grid coordinates for well 2 ( 3 , 5 )

Pumping rate for well # 3 = -150000.00
Grid coordinates for well 3 ( 4 , 3 )

Pumping rate for well # 4 = 2100000.00
Grid coordinates for well 4 ( 6 , 4 )

Pumping rate for well # 5 = 1200000.00
Grid coordinates for well 5 ( 5 , 7 )

Pumping rate for well # 6 =  500000.00
Grid coordinates for well 6 ( 6 , 5 )

Pumping rate for well # 7 =  500000.00
Grid coordinates for well 7 ( 5 , 8 )

Control point 1 coordinates = (4, 8)
Control point 1 desired elevation = ( 570.00 )
Control point 1 calculated elevation = ( 565.56 )
Control point 2 coordinates = (4, 7)
Control point 2 desired elevation = ( 570.00 )
Control point 2 calculated elevation = ( 568.84 )
Control point 3 coordinates = (4, 6)
Control point 3 desired elevation = ( 570.00 )
Control point 3 calculated elevation = ( 569.39 )
Control point 4 coordinates = (4, 5)
Control point 4 desired elevation = ( 570.00 )
Control point 4 calculated elevation = ( 570.35 )
Control point 5 coordinates = (5, 4)
Control point 5 desired elevation = ( 560.00 )
Control point 5 calculated elevation = ( 557.28 )
Control point 6 coordinates = (6, 4)
Control point 6 desired elevation = ( 515.00 )
Control point 6 calculated elevation = ( 519.72 )
Control point 7 coordinates = (6, 5)
Control point 7 desired elevation = ( 540.00 )
Control point 7 calculated elevation = ( 544.83 )
Control point 8 coordinates = (5, 7)
Control point 8 desired elevation = ( 525.00 )
Control point 8 calculated elevation = ( 575.84 )
Control point 9 coordinates = (5, 8)
Control point 9 desired elevation = ( 555.00 )
Control point 9 calculated elevation = ( 559.50 )
The following is the solution head map achieved
after a simulated pumping period of 44.4 days  .

 570.00 570.00 570.00 570.00 570.00 560.00 570.00 570.00 570.00 570.00
 570.00 577.06 573.16 576.75 570.81 565.55 567.48 570.45 573.24 574.88
 570.00 576.57 577 59 576.32 566.66 555.97 565.05 570.37 574.13 575.60
 570.00 574.97 575.45 570.97 557.28 519.72 562.91 571.12 576.71 573.67
 570.00 573.55 576.33 570.35 562.67 544.83 568.46 575.52 585.18 584.43
 570.00 570.96 570.87 569.39 564.25 571.73 578.01 581.83 590.05 591.92
 570.00 569.35 569.28 568.84 525.84 572.24 580.73 583.74 587.54 589.16
 570.00 567.84 566.64 565.56 559.50 572.46 579.54 582.91 584.57 585.08
 570.00 567.52 566.17 567.61 568.91 574.41 580.32 583.51 584.52 583.49
 570.00 568.41 563.75 568.55 571.43 573.98 582.25 585.39 586.28 580.00


Pumping rate for well # 1 = -150000.00
Grid coordinates for well 1 ( 3 , 8 )

Pumping rate for well # 2 = -150000.00
Grid coordinates for well 2 ( 3 , 5 )

Pumping rate for well # 3 = -150000.00
Grid coordinates for well 3 ( 4 , 3 )

Pumping rate for well # 4 = 2100000.00
Grid coordinates for well 4 ( 6 , 4 )

Pumping rate for well # 5 = 1200000.00
Grid coordinates for well 5 ( 5 , 7 )

Pumping rate for well # 6 =  500000.00
Grid coordinates for well 6 ( 6 , 5 )

Pumping rate for well # 7 =  500000.00
Grid coordinates for well 7 ( 5 , 8 )

Control point 1 coordinates = (4, 8)
Control point 1 desired elevation = ( 570.00 )
Control point 1 calculated elevation = ( 566.09 )
Control point 2 coordinates = (4, 7)
Control point 2 desired elevation = ( 570.00 )
Control point 2 calculated elevation = ( 563.32 )
Control point 3 coordinates = (4, 6)
Control point 3 desired elevation = ( 570.00 )
Control point 3 calculated elevation = ( 568.63 )
Control point 4 coordinates = (4, 5)
Control point 4 desired elevation = ( 570.00 )
Control point 4 calculated elevation = ( 570.28 )
Control point 5 coordinates = (5, 4)
Control point 5 desired elevation = ( 560.00 )
Control point 5 calculated elevation = ( 557.27 )
Control point 6 coordinates = (6, 4)
Control point 6 desired elevation = ( 515.00 )
Control point 6 calculated elevation = ( 519.72 )
Control point 7 coordinates = (6, 5)
Control point 7 desired elevation = ( 540.00 )
Control point 7 calculated elevation = ( 544.83 )
Control point 8 coordinates = (5, 7)
Control point 8 desired elevation = ( 525.00 )
Control point 8 calculated elevation = ( 525.10 )
Control point 9 coordinates = (5, 8)
Control point 9 desired elevation = ( 555.00 )
Control point 9 calculated elevation = ( 559.62 )
The following is the solution head map achieved
after a simulated pumping period of 44.4 days

 570.00 570.00 570.00 570.00 570.00 560.00 570.00 570.00 570.00 570.00
 570.00 577.06 578.16 576.75 570.81 565.55 567.48 570.45 573.24 574.88
 570.00 576.57 577.59 576.32 566.66 555.97 565.05 570.37 574.13 573.60
 570.00 574.97 575.44 570.96 557.27 519.72 562.91 571.12 576.71 573.67
 570.00 573.55 576.30 570.28 562.66 544.83 568.46 575.52 585.16 584.43
 570.00 571.00 570.80 568.63 564.09 571.71 578.00 581.83 590.05 591.92
 570.00 569.58 569.11 563.32 525.10 572.18 580.73 583.74 587.54 589.16
 570.00 568.85 569.49 566.09 559.62 572.49 579.54 582.91 584.57 585.08
 570.00 568.07 567.07 567.96 569.02 574.44 580.33 583.51 584.52 583.49
 570.00 568.64 564.19 568.80 571.47 573.99 582.25 585.39 586.28 580.00
```

Appendix F

Plume Case Study Summary

Emerson Electric Company Case Study Application Summary

F.1 Introduction

The Emerson Electric Company site is located in Altamonte Springs, Florida, near the city of Orlando. The Electronic and Space Division of Emerson Electric operated an electrical component manufacturing and assembly plant at this site from January 25, 1979, to the mid 1980's. The site is bordered to the south by a swampy area and on the north and west by a light industrial park. During the site's history, in addition to burying construction debris south of the plant, waste water from metal filming operations was discharged to a septic tank and tile drain on the southeast side of the building (US EPA, 1989).

F.2 Environmental Concerns

A possible contamination problem was discovered on October 20, 1981 during a site inspection by representatives of the Florida Department of Environmental Regulation (FDER). The inspectors found that wastewater from metal filming operations was being discharged to a septic tank without pre-treatment. Subsequently, an electrical conductivity survey was conducted to detect possible zones of contamination. An area of high conductance was delineated southwest of the main building on the property (US EPA, 1989).

F.3 Hydrogeological Investigation

In August 1982, four monitoring wells were installed to define the site's hydrogeology and extent of soil and groundwater contamination. The site hydrogeology consisted of a 50 ft thick water table aquifer consisting of sandy sediments, underlain by approximately 50 feet of clayey sediments.
The water samples collected from the surface water table aquifer were found to be contaminated. Critical contaminants identified were acetone, methyl ethyl ketone, chrome, and lead (US EPA, 1989).

F.4 Aquifer Remediation

The objective of the remediation was to reduce the concentrations of contaminants at the site to below regulated levels. The consulting engineers hired by Emerson designed an extraction system consisting of five pumping wells, with a total system pumping rate of 30 gallons per minute. The FDER consented to allow the collected waste water to be discharged directly into the municipal sanitary sewer network. This water would eventually be treated by the Altamonte Springs water treatment plant using normal treatment methods. The consultant predicted that the system would properly purge the aquifer clean in nine months (US EPA, 1989).

F.5 Summary

The remediation program continued for 22 months before aquifer concentrations of the critical contaminants fell below regulated limits. In January, 1989, the site was taken off of the State Action Site List and was considered properly remediated (US EPA, 1989).

Appendix G

Plume Case Study Program Input

Emerson Electric Company Case Study Application Input Data

G.1 Finite Difference Parameters

Number of columns:	10
Number of rows:	10
Grid Spacing:	50 ft
Finite Difference Tolerance of Convergence:	0.1 ft
Total number of time steps:	20
Initial time step:	2 days
Anisotropy factor (Txx/Tyy)	1.0

G.2 Hydrogeologic Parameters

Aquifer:	Unconfined
Induced Infiltration:	False
Leaky Artesian Conditions:	False
Average site gradient:	0.01
Average aquifer thickness	50 ft.
Number of pumping wells:	2

Pumping Well No.		1
Pumping rate:		7000 US gpd
Maximum permissible percent dewatering	0.25	
X grid coordinate of top left corner of search area:	3	
y grid coordinate of top left corner of search area:	7	
X grid coordinate of bottom right corner of search area:	4	
y grid coordinate of bottom right corner of search area:	8	

Pumping Well No.		2
Pumping rate:		7000 US gpd
Maximum permissible percent dewatering	0.25	
X grid coordinate of top left corner of search area:	7	
y grid coordinate of top left corner of search area:	7	
X grid coordinate of bottom right corner of search area:	8	
y grid coordinate of bottom right corner of search area:	8	

Emerson Electric Company Case Study Application Input Data

G.3 Plume Parameters

No. of Shapes required to represent the plume	1
X grid coordinate of top left corner of plume area:	2
y grid coordinate of top left corner of plume area:	5
X grid coordinate of bottom right corner of plume area:	8
y grid coordinate of bottom right corner of plume area:	8

TABLE G.1
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Aquifer Base Elevation Map*

Rows	Columns									
	1	2	3	4	5	6	7	8	9	10
1	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
2	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
3	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
4	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
5	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
6	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
7	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
8	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
9	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
10	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100

* units are in feet

TABLE G.2
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Aquifer Hydraulic Conductivity Map*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	80	80	80	80	80	80	80	80	80	80
2	80	80	80	80	80	80	80	80	80	80
3	30	80	80	80	80	80	80	80	80	80
4	80	80	80	80	80	80	80	80	80	80
5	80	80	80	80	80	80	80	80	80	80
6	80	80	80	80	80	80	80	80	80	80
7	80	80	80	80	80	80	80	80	90	80
8	80	80	60	80	80	80	80	80	80	80
9	80	30	80	80	80	80	80	80	80	80
10	80	80	80	80	80	80	80	80	80	80

* units are in square feet/day

TABLE G.3
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Contaminant Concentration Map*

Rows	Columns									
	1	2	3	4	5	6	7	8	9	10
1	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2
3	2	2	2	2	2	2	2	2	2	2
4	2	2	2	2	2	2	2	2	2	2
5	18	18	25	25	25	25	25	18	18	18
6	25	25	25	50	50	50	25	18	18	18
7	25	25	30	50	50	50	25	25	25	18
8	25	25	30	50	50	50	25	25	25	2
9	25	25	25	30	30	30	25	25	2	2
10	25	25	25	25	25	25	25	25	2	2

* units are in μg/Litre

TABLE G.4
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Potentiometric Surface Elevation Map*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	78	79	79	79	80	80	81	81	81	81
2	78	79	79	79	79	80	80	81	81	81
3	78	78	79	79	79	80	80	80	81	81
4	78	78	78	78	79	80	80	80	80	81
5	78	78	78	78	79	79	79	80	80	80
6	78	78	78	78	79	79	79	80	80	80
7	78	78	78	78	78	79	79	80	80	80
8	78	78	78	78	78	78	78	79	79	80
9	78	78	78	78	78	78	78	78	78	80
10	78	78	78	78	78	78	78	78	78	79

* units are in feet

TABLE G.5
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Leakance Map*

Rows	*Columns*									
	1	2	3	4	5	6	7	8	9	10
1	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
3	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
4	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
5	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
6	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
7	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
8	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
9	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
10	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2

* units are in gal/day/cubic foot

TABLE G.6
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Storativity Map*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
3	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
4	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
8	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5

* values are unitless

TABLE G.7
Thesis Program Data Input
Plume Case Study Application
Emerson Electric Company Aquifer Remediation Project

Transmissivity Map*

Rows	*Columns*									
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
2	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
3	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
4	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
5	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
6	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
7	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
8	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
9	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000
10	4000	4000	4000	4000	4000	4000	4000	4000	4000	4000

* units are in US gallons/day foot

Appendix H

Plume Case Study Program Output

```
*** Optimum Plume Control Solution Summary ***


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of  4.4 days

  78.24  78.21  78.16  78.12  78.08  78.06  78.04  78.03  78.02  78.01
  78.38  78.34  78.26  78.18  78.13  78.09  78.06  78.04  78.03  78.02
  78.69  78.61  78.46  78.33  78.24  78.17  78.12  78.07  78.05  78.04
  79.01  78.91  78.75  78.56  78.44  78.33  78.22  78.14  78.09  78.07
  79.38  79.23  79.05  78.87  78.74  78.59  78.38  78.24  78.15  78.11
  79.72  79.55  79.32  79.07  78.98  78.82  78.53  78.38  78.26  78.19
  80.05  79.78  79.42  78.87  79.16  79.03  78.43  78.55  78.41  78.31
  80.35  80.14  79.78  79.49  79.50  79.39  79.11  78.89  78.64  78.49
  80.56  80.43  80.17  79.87  79.76  79.64  79.44  79.18  78.93  78.73
  80.67  80.57  80.39  80.15  79.93  79.78  79.60  79.39  79.17  78.96


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of  7.3 days

  78.35  78.32  78.27  78.21  78.16  78.12  78.09  78.06  78.05  78.04
  78.46  78.42  78.35  78.28  78.21  78.16  78.12  78.08  78.06  78.05
  78.68  78.62  78.52  78.42  78.32  78.25  78.18  78.13  78.10  78.08
  78.95  78.88  78.75  78.61  78.49  78.38  78.27  78.20  78.15  78.12
  79.25  79.15  78.99  78.81  78.68  78.54  78.39  78.29  78.23  78.19
  79.54  79.41  79.19  78.93  78.85  78.69  78.45  78.39  78.33  78.29
  79.82  79.65  79.34  78.75  78.97  78.82  78.27  78.49  78.47  78.42
  80.09  79.94  79.68  79.40  79.32  79.17  78.91  78.81  78.69  78.60
  80.30  80.18  79.98  79.75  79.60  79.44  79.25  79.07  78.91  78.80
  80.41  80.32  80.15  79.94  79.76  79.59  79.41  79.22  79.05  78.93
```

```
Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 10.7 days

   78.44  78.41  78.36  78.30  78.24  78.19  78.15  78.11  78.09  78.08
   78.52  78.49  78.43  78.36  78.29  78.23  78.18  78.14  78.11  78.10
   78.69  78.64  78.56  78.47  78.38  78.30  78.23  78.18  78.15  78.13
   78.90  78.84  78.74  78.62  78.51  78.40  78.31  78.25  78.21  78.19
   79.14  79.06  78.92  78.75  78.63  78.51  78.38  78.32  78.28  78.26
   79.38  79.27  79.07  78.81  78.73  78.59  78.38  78.38  78.37  78.36
   79.62  79.48  79.19  78.60  78.81  78.66  78.14  78.44  78.49  78.48
   79.85  79.73  79.51  79.23  79.14  78.98  78.76  78.72  78.68  78.65
   80.03  79.94  79.76  79.56  79.41  79.24  79.07  78.95  78.86  78.80
   80.14  80.05  79.90  79.72  79.55  79.38  79.22  79.08  78.96  78.89


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 14.9 days

   78.51  78.48  78.43  78.38  78.32  78.26  78.21  78.17  78.15  78.14
   78.57  78.54  78.49  78.42  78.35  78.29  78.24  78.19  78.17  78.15
   78.69  78.65  78.58  78.50  78.42  78.34  78.28  78.23  78.20  78.19
   78.85  78.80  78.71  78.60  78.50  78.41  78.33  78.28  78.25  78.24
   79.03  78.96  78.83  78.68  78.58  78.47  78.36  78.33  78.31  78.30
   79.22  79.12  78.93  78.68  78.62  78.49  78.32  78.36  78.38  78.39
   79.41  79.29  79.01  78.43  78.65  78.52  78.04  78.38  78.48  78.50
   79.60  79.50  79.30  79.03  78.95  78.81  78.62  78.63  78.64  78.63
   79.76  79.68  79.52  79.34  79.20  79.05  78.91  78.83  78.79  78.76
   79.85  79.77  79.64  79.48  79.33  79.18  79.04  78.94  78.87  78.83
```

```
Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 19.9 days

  78.55   78.53   78.49   78.43   78.37   78.32   78.27   78.23   78.21   78.19
  78.60   78.57   78.52   78.46   78.40   78.34   78.29   78.25   78.22   78.21
  78.68   78.65   78.59   78.51   78.44   78.37   78.31   78.27   78.25   78.24
  78.79   78.74   78.66   78.57   78.48   78.40   78.34   78.30   78.28   78.28
  78.91   78.85   78.74   78.60   78.51   78.42   78.34   78.32   78.32   78.33
  79.05   78.96   78.78   78.55   78.50   78.40   78.25   78.32   78.37   78.39
  79.19   79.08   78.82   78.25   78.49   78.38   77.93   78.31   78.44   78.47
  79.35   79.25   79.07   78.82   78.75   78.64   78.48   78.53   78.57   78.58
  79.48   79.40   79.27   79.10   78.98   78.86   78.75   78.71   78.69   78.69
  79.55   79.49   79.37   79.23   79.10   78.97   78.87   78.80   78.76   78.74


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 25.8 days

  78.57   78.55   78.51   78.46   78.41   78.36   78.31   78.28   78.26   78.25
  78.60   78.57   78.53   78.48   78.42   78.37   78.32   78.29   78.27   78.26
  78.65   78.62   78.56   78.50   78.44   78.38   78.33   78.30   78.28   78.27
  78.71   78.67   78.60   78.52   78.44   78.38   78.33   78.30   78.30   78.30
  78.79   78.73   78.63   78.50   78.43   78.36   78.29   78.30   78.31   78.32
  78.87   78.79   78.63   78.41   78.38   78.30   78.17   78.26   78.33   78.36
  78.97   78.87   78.62   78.07   78.33   78.24   77.82   78.22   78.37   78.42
  79.09   79.00   78.83   78.60   78.55   78.46   78.33   78.41   78.47   78.50
  79.19   79.13   79.00   78.85   78.75   78.66   78.58   78.57   78.58   78.58
  79.25   79.19   79.09   78.97   78.86   78.76   78.69   78.65   78.64   78.63
```

```
Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 33.0 days

   78.55  78.54  78.50  78.46  78.42  78.37  78.34  78.31  78.29  78.28
   78.57  78.55  78.51  78.46  78.41  78.37  78.33  78.31  78.29  78.28
   78.59  78.56  78.52  78.46  78.41  78.36  78.32  78.30  78.29  78.29
   78.62  78.58  78.52  78.45  78.38  78.33  78.29  78.28  78.29  78.29
   78.65  78.60  78.50  78.39  78.33  78.27  78.23  78.25  78.28  78.29
   78.69  78.61  78.46  78.25  78.24  78.18  78.08  78.18  78.26  78.30
   78.75  78.65  78.41  77.88  78.15  78.09  77.69  78.11  78.27  78.33
   78.83  78.75  78.59  78.37  78.35  78.28  78.17  78.28  78.36  78.39
   78.91  78.85  78.73  78.61  78.53  78.46  78.40  78.41  78.44  78.46
   78.95  78.90  78.81  78.71  78.62  78.55  78.50  78.49  78.49  78.50


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 41.6 days

   78.50  78.49  78.46  78.42  78.39  78.35  78.32  78.30  78.29  78.29
   78.50  78.48  78.45  78.41  78.38  78.34  78.31  78.29  78.28  78.28
   78.50  78.48  78.44  78.39  78.35  78.31  78.29  78.27  78.27  78.27
   78.50  78.47  78.41  78.35  78.30  78.26  78.23  78.23  78.24  78.25
   78.49  78.45  78.36  78.26  78.21  78.17  78.14  78.17  78.21  78.23
   78.50  78.42  78.28  78.09  78.09  78.04  77.96  78.08  78.17  78.22
   78.52  78.42  78.19  77.67  77.97  77.92  77.54  77.98  78.15  78.22
   78.57  78.49  78.34  78.14  78.14  78.09  78.00  78.12  78.21  78.26
   78.63  78.57  78.47  78.36  78.30  78.24  78.21  78.24  78.28  78.31
   78.66  78.61  78.54  78.45  78.38  78.33  78.30  78.30  78.32  78.34

Well Number 1
```

```
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 51.9 days

  78.41  78.39  78.37  78.34  78.31  78.29  78.27  78.26  78.25  78.25
  78.40  78.38  78.35  78.32  78.29  78.27  78.25  78.24  78.24  78.23
  78.38  78.36  78.32  78.28  78.25  78.22  78.21  78.20  78.21  78.21
  78.35  78.32  78.27  78.22  78.18  78.15  78.14  78.15  78.16  78.17
  78.32  78.27  78.19  78.10  78.07  78.04  78.02  78.06  78.11  78.13
  78.29  78.22  78.08  77.90  77.91  77.88  77.81  77.94  78.04  78.09
  78.28  78.19  77.97  77.46  77.77  77.74  77.37  77.82  78.01  78.08
  78.31  78.23  78.09  77.91  77.92  77.88  77.81  77.94  78.05  78.10
  78.34  78.29  78.20  78.10  78.06  78.02  78.00  78.05  78.10  78.13
  78.37  78.33  78.26  78.18  78.13  78.10  78.08  78.10  78.13  78.15


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 64.3 days

  78.26  78.25  78.23  78.21  78.19  78.17  78.16  78.16  78.15  78.15
  78.25  78.23  78.21  78.19  78.16  78.15  78.14  78.13  78.13  78.13
  78.21  78.20  78.17  78.14  78.11  78.09  78.08  78.09  78.09  78.10
  78.17  78.14  78.10  78.05  78.02  78.00  77.99  78.01  78.03  78.05
  78.11  78.07  77.99  77.91  77.89  77.87  77.86  77.91  77.96  77.99
  78.06  77.99  77.86  77.69  77.71  77.69  77.63  77.77  77.88  77.93
  78.03  77.94  77.73  77.23  77.55  77.53  77.17  77.63  77.82  77.90
  78.04  77.97  77.83  77.66  77.68  77.66  77.59  77.74  77.85  77.90
  78.06  78.01  77.93  77.84  77.81  77.78  77.78  77.83  77.89  77.92
  78.08  78.04  77.98  77.91  77.87  77.85  77.85  77.88  77.92  77.94


Well Number 1
Pumping rate =    7000.00
```

```
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 79.2 days

  78.07  78.06  78.05  78.03  78.02  78.00  78.00  78.00  78.00  78.00
  78.05  78.04  78.02  78.00  77.98  77.97  77.97  77.97  77.97  77.98
  78.01  77.99  77.97  77.94  77.92  77.91  77.91  77.91  77.92  77.93
  77.95  77.92  77.88  77.84  77.82  77.80  77.81  77.83  77.85  77.87
  77.88  77.84  77.77  77.69  77.67  77.66  77.65  77.71  77.77  77.80
  77.81  77.74  77.62  77.45  77.48  77.47  77.41  77.56  77.67  77.73
  77.76  77.67  77.47  76.97  77.30  77.29  76.94  77.41  77.60  77.68
  77.75  77.69  77.56  77.39  77.42  77.40  77.35  77.50  77.61  77.67
  77.77  77.72  77.64  77.56  77.54  77.52  77.52  77.58  77.65  77.68
  77.78  77.75  77.69  77.63  77.60  77.59  77.59  77.63  77.67  77.70


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 97.0 days

  77.83  77.82  77.81  77.79  77.78  77.78  77.77  77.78  77.78  77.78
  77.80  77.79  77.78  77.76  77.75  77.74  77.74  77.75  77.75  77.76
  77.75  77.74  77.72  77.69  77.68  77.67  77.67  77.68  77.70  77.71
  77.68  77.66  77.62  77.58  77.57  77.56  77.56  77.59  77.62  77.64
  77.60  77.56  77.49  77.42  77.41  77.40  77.40  77.46  77.52  77.55
  77.52  77.46  77.33  77.17  77.21  77.20  77.15  77.30  77.42  77.47
  77.46  77.38  77.17  76.69  77.02  77.01  76.66  77.14  77.34  77.41
  77.45  77.38  77.26  77.09  77.13  77.12  77.07  77.22  77.34  77.40
  77.46  77.41  77.34  77.26  77.24  77.23  77.24  77.30  77.37  77.41
  77.47  77.43  77.38  77.32  77.30  77.29  77.30  77.34  77.39  77.42


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
```

```
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 118.4 days

  77.52  77.52  77.51  77.50  77.49  77.48  77.48  77.49  77.49  77.50
  77.49  77.49  77.47  77.46  77.45  77.45  77.45  77.46  77.46  77.47
  77.44  77.43  77.41  77.39  77.37  77.37  77.38  77.39  77.41  77.42
  77.37  77.34  77.31  77.27  77.26  77.25  77.26  77.29  77.32  77.34
  77.28  77.24  77.17  77.11  77.09  77.09  77.09  77.16  77.22  77.25
  77.19  77.13  77.01  76.85  76.89  76.88  76.84  76.99  77.11  77.16
  77.13  77.04  76.84  76.35  76.69  76.68  76.34  76.82  77.02  77.10
  77.10  77.04  76.92  76.75  76.79  76.79  76.74  76.90  77.02  77.08
  77.11  77.07  76.99  76.92  76.90  76.90  76.90  76.97  77.04  77.08
  77.12  77.09  77.03  76.98  76.96  76.95  76.97  77.01  77.06  77.09


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 144.1 days

  77.15  77.14  77.13  77.13  77.12  77.12  77.12  77.12  77.13  77.14
  77.12  77.11  77.10  77.09  77.08  77.08  77.08  77.09  77.10  77.11
  77.06  77.05  77.03  77.01  77.00  77.00  77.01  77.02  77.04  77.05
  76.99  76.97  76.93  76.90  76.88  76.88  76.89  76.92  76.95  76.97
  76.89  76.86  76.79  76.73  76.72  76.71  76.72  76.78  76.84  76.88
  76.80  76.74  76.62  76.46  76.50  76.50  76.46  76.61  76.73  76.79
  76.73  76.65  76.45  75.97  76.30  76.30  75.96  76.44  76.64  76.72
  76.71  76.65  76.52  76.36  76.40  76.40  76.36  76.51  76.64  76.70
  76.71  76.67  76.60  76.52  76.51  76.51  76.52  76.59  76.66  76.70
  76.72  76.69  76.64  76.59  76.57  76.56  76.58  76.63  76.68  76.70


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 174.9 days

  76.70  76.69  76.68  76.68  76.67  76.67  76.67  76.68  76.69  76.69
  76.67  76.66  76.65  76.64  76.63  76.63  76.63  76.64  76.65  76.66
  76.61  76.60  76.58  76.56  76.55  76.55  76.56  76.58  76.59  76.60
  76.53  76.51  76.48  76.44  76.43  76.43  76.44  76.47  76.50  76.52
  76.44  76.40  76.34  76.27  76.26  76.26  76.27  76.33  76.39  76.43
  76.34  76.28  76.16  76.01  76.05  76.05  76.00  76.16  76.28  76.33
  76.27  76.19  75.99  75.51  75.84  75.84  75.50  75.98  76.18  76.26
  76.25  76.19  76.06  75.90  75.94  75.94  75.90  76.06  76.18  76.24
  76.25  76.21  76.13  76.06  76.05  76.05  76.06  76.13  76.20  76.24
  76.25  76.22  76.17  76.12  76.10  76.10  76.12  76.17  76.22  76.25


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 211.9 days

  76.15  76.15  76.14  76.13  76.13  76.13  76.13  76.14  76.15  76.15
  76.12  76.12  76.11  76.10  76.09  76.09  76.09  76.10  76.11  76.12
  76.07  76.06  76.04  76.02  76.01  76.01  76.02  76.03  76.05  76.06
  75.99  75.97  75.93  75.90  75.89  75.88  75.90  75.93  75.96  75.98
  75.89  75.85  75.79  75.72  75.72  75.71  75.72  75.79  75.85  75.89
  75.79  75.74  75.62  75.46  75.50  75.50  75.46  75.61  75.73  75.79
  75.72  75.64  75.44  74.96  75.30  75.30  74.96  75.44  75.64  75.72
  75.70  75.64  75.51  75.35  75.39  75.39  75.35  75.51  75.63  75.69
  75.70  75.66  75.58  75.51  75.50  75.50  75.51  75.58  75.65  75.69
  75.70  75.67  75.62  75.57  75.55  75.55  75.57  75.62  75.67  75.70


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001
```

```
Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 256.2 days

  75.50  75.50  75.49  75.48  75.47  75.47  75.48  75.49  75.49  75.50
  75.47  75.46  75.45  75.44  75.44  75.44  75.44  75.45  75.46  75.47
  75.41  75.40  75.38  75.36  75.36  75.35  75.36  75.38  75.40  75.41
  75.33  75.31  75.28  75.24  75.23  75.23  75.24  75.27  75.31  75.33
  75.23  75.20  75.13  75.07  75.06  75.06  75.07  75.13  75.20  75.23
  75.14  75.08  74.96  74.80  74.85  74.85  74.80  74.96  75.08  75.14
  75.07  74.99  74.78  74.30  74.64  74.64  74.30  74.78  74.98  75.06
  75.04  74.98  74.86  74.70  74.74  74.74  74.69  74.85  74.98  75.04
  75.04  75.00  74.93  74.85  74.84  74.84  74.85  74.93  75.00  75.04
  75.04  75.01  74.96  74.92  74.90  74.90  74.92  74.96  75.01  75.04


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 309.5 days

  74.71  74.71  74.70  74.69  74.69  74.69  74.69  74.70  74.71  74.71
  74.68  74.68  74.67  74.66  74.65  74.65  74.66  74.67  74.68  74.68
  74.63  74.62  74.60  74.58  74.57  74.57  74.58  74.60  74.61  74.63
  74.54  74.52  74.49  74.46  74.45  74.45  74.46  74.49  74.52  74.54
  74.45  74.41  74.35  74.28  74.27  74.27  74.28  74.35  74.41  74.45
  74.35  74.29  74.17  74.02  74.06  74.06  74.02  74.17  74.29  74.35
  74.28  74.20  74.00  73.52  73.85  73.85  73.51  74.00  74.20  74.28
  74.25  74.19  74.07  73.91  73.95  73.95  73.91  74.07  74.19  74.25
  74.25  74.21  74.14  74.07  74.05  74.05  74.07  74.14  74.21  74.25
  74.26  74.23  74.18  74.13  74.11  74.11  74.13  74.18  74.23  74.26


Well Number 1
Pumping rate =    7000.00
Grid coordinates for well  1 ( 4 , 7 )
Capture width in the X direction =  8.7500000000000E+0001
Capture width in the Y direction =  8.7500000000000E+0001
```

```
Well Number 2
Pumping rate =    7000.00
Grid coordinates for well  2 ( 7 , 7 )
Capture width in the X direction =  8.75000000000000E+0001
Capture width in the Y direction =  8.75000000000000E+0001

The following is the solution head map
achieved after a simulated pumping period of 373.4 days

  73.77  73.77  73.76  73.75  73.75  73.75  73.75  73.76  73.77  73.77
  73.74  73.74  73.72  73.71  73.71  73.71  73.71  73.72  73.73  73.74
  73.68  73.67  73.65  73.64  73.63  73.63  73.64  73.65  73.67  73.68
  73.60  73.58  73.55  73.52  73.50  73.50  73.52  73.55  73.58  73.60
  73.50  73.47  73.41  73.34  73.33  73.33  73.34  73.40  73.47  73.50
  73.41  73.35  73.23  73.07  73.12  73.12  73.07  73.23  73.35  73.41
  73.34  73.26  73.05  72.57  72.91  72.91  72.57  73.05  73.25  73.34
  73.31  73.25  73.12  72.96  73.01  73.01  72.96  73.12  73.25  73.31
  73.31  73.27  73.20  73.12  73.11  73.11  73.12  73.19  73.27  73.31
  73.31  73.28  73.23  73.18  73.16  73.16  73.18  73.23  73.28  73.31
```

VITA AUCTORIS

Robert J. MacLean Hyde was born in Wiarton, Ontario on November 1, 1963. In 1982, he enrolled in Geological Engineering at the University of Windsor and obtained a B.A.Sc in 1987. Currently, he is a candidate for a Master's degree in Geological Engineering at Windsor specializing in environmental engineering and hydrogeology. For the past two years, Robert has been working as a consulting engineer in Mississauga, Ontario, where he has been actively involved in computer modelling impact assessments, landfill site design and monitoring, hydrogeologic site investigations, environmental audits, and site remediations for consulting projects in Ontario, Nova Scotia, and the United States.