Order Number 9304631

Modeling unsteady state leachate flow in a landfill using finite difference and boundary element methods

Ahmed, Shabbir, Ph.D.

City University of New York, 1992

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

A

MODELING UNSTEADY STATE LEACHATE FLOW IN A LANDFILL USING FINITE DIFFERENCE AND BOUNDARY ELEMENT METHODS

by

SHABBIR AHMED

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1992

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

8-27-92
Date

Professor Reza M. Khanbilvardi
Chair of Examining Committee

8/27/92
Date

Professor Gerard G. Lowen
Executive Officer

Professor John Fillos

Professor Ali M. Sadegh

Professor O. Lehn Franke

Dr. Paul Grosser
Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

ABSTRACT

MODELING UNSTEADY STATE LEACHATE FLOW IN A LANDFILL USING FINITE DIFFERENCE AND BOUNDARY ELEMENT METHODS

by

Shabbir Ahmed

Advisor: Professor Reza M. Khanbilvardi

The physical processes involving leachate flow in a solid waste landfill are described by the unsaturated flow through the refuse to the saturated leachate mound at the bottom of a landfill. The moisture-flow in the unsaturated zone helps build up the saturated leachate mound at the bottom of a landfill. The moisture content in the unsaturated zone is obtained by solving the two-dimensional unsaturated moisture-flow equation using numerical techniques. A two-dimensional unsteady state Flow Investigation for Landfill Leachate (FILL) model, based on the implicit finite-difference technique, has been developed to describe the leachate flow process in a landfill.

To obtain accuracy and efficiency in numerical modeling, it is important to investigate the numerical solution techniques suitable to solve the governing equations. Accuracy and efficiency of the boundary integral method over the finite-difference method has been investigated. Two approaches, direct Green's function

and perturbation Green's function formulations, have been developed to solve the unsaturated flow problem. Direct Green's function and perturbation Green's function boundary integral solutions are found to be more accurate than both the Gauss-Seidel iteration and Gauss-Jordan elimination method of finite-difference solution.

The efficiency of the boundary integral formulation for the computation of the moisture-flux is an advantage that is useful to estimate leachate accretion in a landfill. A close agreement of the internal fluxes with the exact solution shows the ability of the boundary integral methods to compute accurate recharge from the unsaturated zone to the saturated leachate mound.

To gain confidence in the finite-difference scheme for the saturated leachate flow equation, two numerical methods, Gauss-Seidel iteration and Gauss-Jordan elimination, are used to solve the finite-difference expressions. Satisfactory performance of the two methods are observed in computing leachate mound-head for leachate accretion from the unsaturated zone. The finite-difference FILL model representing the unsaturated-saturated flow has been applied to an example landfill section to demonstrate the effectiveness of the model in providing information such as evapotranspiration, surface runoff, recharge, and leachate flow rates in both the lateral and vertical directions.

ACKNOWLEDGEMENT

The present research work would not have been completed without the support of many individuals in various ways. I want to give special thanks to Professor Reza M. Khanbilvardi for the guidance, supervision, advice, and encouragement during this research work. Heartful thanks goes out to Professor John Fillos who enriched my studies with his reviews, questions, and valuable suggestions at all stages of this research work. I acknowledge his help and guidance throughout my graduate program at The City University of New York. My sincere thanks also are due to Professor Ali M. Sadegh for his advice, guidance, and help in understanding the boundary element methodology. I owe to him all that I learned about the application of boundary element technique to solve unsaturated flow problems.

Professor O. Lehn Franke, Dr. Paul Grosser and Professor Gerard G. Lowen were on my doctoral examination committee throughout this research work. I warmly acknowledge their suggestions and valuable comments that gave me better understanding of this research work. Thankful appreciation is due to Mr. Phillip J. Gleason, Director of Landfill Engineering, New York City Department of Sanitation for his valuable suggestions and help in understanding the specific problems related to the leachate flow in a landfill.

The financial support provided by the Research Foundation and the Graduate School of The City University of New York is highly appreciated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE	TITLE	PAGE

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 INTRODUCTION

Leachate is formed from that portion of precipitation that infiltrates through a solid waste landfill. The water extracts dissolved and/or suspended solids from the landfilled wastes. The resulting contaminated water is called leachate; it has the potential to pollute both groundwater and surface water resources. Because of it's importance when investigating the risks due to groundwater and surface water contamination, an estimate of the magnitude and timing of the discharged leachate is desirable.

Mathematical models are used for evaluating a variety of scenarios involved in the leachate flow processes. The complexities involved in the simulation of leachate flow process by mathematical models are a major focus of research at the present time. Of particular interest and difficulty are the simulation of leachate transport that occur in both the unsaturated and saturated zones in a landfill. To obtain accuracy and efficiency in the simulation for moisture transport, considerable research is invested for the development of high-quality numerical models.

To estimate the net infiltration into the landfill, the amount of surface runoff and evapotranspiration needs to be predicted accurately. In a landfill, side slope has an important bearing on the amount of surface runoff. However, models currently used in the field of landfills do not take into consideration the effect of slope and surface roughness (Fenn et al., 1975; Schroeder et al., 1984; Demetracopoulos et al., 1986).

Evapotranspiration depends on the climatologic and soil data of the landfill area. The modified Penman method developed by Ritchie (1972) and used in Chemical Runoff and Erosion from Agricultural Management Systems (CREAMS) model (Knisel, 1980) and Hydrologic Evaluation of Landfill Performance (HELP) model (Schroeder et al., 1984), takes into consideration the effects of climatologic (temperature and solar radiation) and soil data (soil moisture, porosity, field capacity, and wilting point) to compute evapotranspiration. This method can be used for a realistic estimate of evapotranspiration from a landfill surface.

The total precipitation minus the runoff and evapotranspiration equals the amount of water that contributes to the leachate already present within the landfill. Some of the infiltrated water, however, is retained by the soil matrix in the unsaturated zone. The moisture variation in the unsaturated zone generates leachate accretion. The saturated leachate mound is developed by the leachate accretion from the unsaturated zone. The governing partial differential equations describing the leachate flow phenomena in both the unsaturated and saturated zones of a landfill are obtained from the continuity equation and Darcy's Law. The time-history of leachate accretion is obtained by numerical solution of the partial differential equation describing the moisture-flow in the unsaturated zone in a landfill.

The numerical solution of the unsaturated moisture-flow equation can be done either in steady state or unsteady state flow condition. Since the process of the development of leachate mound-head is strictly unsteady due to changes in the hydraulic conductivity and diffusivity of the medium, it would be realistic to develop numerical models for leachate accretion rate based on the unsteady state solution of the unsaturated moisture-flow equation. The unsteady leachate flow models can be based on the solution of either one-dimensional or two-dimensional moisture-flow equation in the unsaturated zone in a landfill. The unsteady one-dimensional leachate flow model considers leachate accretion only in the vertical direction. However, the leachate accretion rate in the field changes with respect to distance and time. So, a realistic estimate of the leachate flow can be obtained by solving the two-dimensional moisture-flow equation in the vertical section of a landfill.

High-quality numerical models depend on the efficient and accurate numerical solution procedure. Finite-difference method was used extensively in porous media flow problems since 1960s (Tysen and Weber, 1963; Richtmeyer and Morton, 1967; Cooley, 1971; Prickett and Lonnquist, 1971; Remson et al., 1971, Thomas, 1973; Abnish, 1975; Abbott, 1982; Rulon et al., 1985; Demetracopoulos et al., 1986; McKeon and Chu, 1987; and Demetracopoulos, 1988). The finite-difference method requires discretization of the whole domain along with the adjustment of the geometry of the boundary to fit the finite-difference grid spacing. The method requires to establish grid size such that the variable is satisfactorily approximated everywhere in the domain.

Finite element method was used since 1970s (Neuman, 1973; Gupta and Tanji, 1976; Kuppusamy et al., 1987). This method permits to consider small elements in regions of rapidly varying flow and along the boundary to represent the physical geometry of the domain more realistically. In the finite element method, it is required to find an element size in order to approximate the partial differential equation everywhere in the domain. In such method, the internal fluxes can not be calculated directly in the solution region. The internal fluxes are important parameter to describe leachate flow problem specifically to estimate leachate accretion from the unsaturated zone in a landfill. In both the finite-difference and finite element methods, the internal fluxes can not be calculated directly in the solution domain.

Recently, researchers have begun to employ boundary integral method (also known as boundary element method) to solve problems in saturated porous media flow (Aral and Tang, 1988; Fogden et al., 1988; Lafe and Cheng, 1987). The basic idea is that through some mathematical identity the governing differential equation on the domain is transformed to an integral equation on the boundary. There is always a singularity associated with the transformation which must be calculated. In the process of transformation of the differential equation to the boundary integral equation, a three-dimensional problem is reduced to a surface integral (two-dimensional problem) and a two-dimensional problem is reduced to a line integral (one-dimensional problem). The new integral equation and boundary conditions are then solved through discretization of the boundary which leads to a system of algebraic equations. Solution of the system of equations results in information about parameters on the boundary which permits the solution of the differential equation at any point in the field.

One important feature of the boundary integral method is that one can determine the dependent variable directly from the mathematical formulation at any point in the field, whereas, in finite element or finite difference method, one can only calculate the dependent variable at node points. The velocity at the internal points is an important parameter for solving groundwater flow problems. An accurate evaluation of the velocity of the hydraulic flow is needed to solve contaminant transport problems. The knowledge of the velocities of leachate flow in the unsaturated zone in a landfill is useful to determine leachate accretion. Unlike finite-difference and finite element methods, the velocities at any point in the domain can be calculated directly using the boundary integral formulation. The ability of the boundary integral method to determine velocities at any point in the domain help estimate recharge to the saturated leachate mound in a landfill.

In this dissertation research, an unsteady two-dimensional Flow Investigation for Landfill Leachate (FILL) model is developed using the finite difference method. In the FILL model, the unsaturated and the saturated leachate flow equations are solved in a two-dimensional vertical domain using the finite difference method. A finite-difference scheme is also used for the kinematic wave equation to compute surface runoff on the upper boundary. Infiltration is a function of moisture content, hydraulic conductivity, and depth of water on the soil surface. The infiltration equation developed by Philip (1969), which considers the effect of the aforementioned parameters, is used in the present model.

The boundary integral equation method is used to investigate the leachate flow in the unsaturated zone of a landfill. Two boundary integral approaches are developed to solve the two-dimensional unsaturated leachate flow equation. In one approach, the time-dependent Green's function is used to compute the moisture content and the moisture flux in the domain. In the second approach, a perturbation Green's function boundary integral solution process is developed to compute both moisture content and moisture-flux in a landfill.

The accuracy, efficiency and representation of the physical system using finite difference and boundary element methods are compared by solving the two-dimensional leachate flow equation in the unsaturated zone of a landfill. The two methods are also evaluated with respect to the ability to determine the function and flux in the domain of a landfill.

1.2 PRESENT STATUS OF THE RESEARCH

Water balance method is one of the simple procedures to compute leachate flow from a landfill. This method was used in the Environmental Protection Agency (EPA) model developed by Fenn et al. (1975). In the water balance method, the amount of water percolating through the solid waste landfill is obtained by subtracting surface runoff, change in soil moisture of the cover soil, and evapotranspiration from the total precipitation. In this method, moisture movement through the refuse is not considered. The EPA model was used in a number of studies to estimate leachate generation rates for landfills in Cincinnati, Ohio; Orlando, Florida; and Los Angeles, California. Later models developed by Dass et al. (1977), Perrier and Gibson (1980), and Gee (1981) were also based on the water balance techniques. None of the aforementioned models include leachate flow in the unsaturated zone and computation of leachate mound-head generated due to leachate accumulation at the bottom of a landfill.

Recent mathematical models consider moisture-flow through the landfill in both steady and unsteady state flow conditions, and estimate leachate flow vertically and laterally in a landfill. Schroeder et al. (1984) developed the Hydrologic Evaluation of Landfill Performance (HELP) model, which is based on a quasi two-dimensional and quasi steady state approach for estimating leachate flow from waste disposal sites. This model is quasi two-dimensional, because it does not compute the vertical and lateral components of flow in each layer of the landfill profile simultaneously. The layers in which the hydraulic conductivity is large enough so that the flow occurs only downward and is not significantly restricted are called vertical percolation layers. According to HELP, in these layers, lateral flow components are assumed to be zero. Layers in which hydraulic conductivity is high, but the flow is restricted in the downward direction by an impervious layer, are considered lateral drainage layers. The model is called quasi steady state because the vertical flow component is computed using an unsteady moisture routing equation and the lateral flow component is obtained from the steady state solution of the Boussinesq equation.

In the HELP model, the surface runoff at the upper boundary of the landfill is computed using a soil conservation service (SCS) curve-number method, which does not consider the effect of surface roughness and slope along the side of the landfill. The modified Penman method developed by Ritchie (1972) and used in the Chemical Runoff and Erosion from Agricultural Management Systems (CREAMS) model (Knisel, 1980) is used to compute the potential evapotranspiration in the HELP model. Despite the quasi two-dimensional and quasi steady state treatment of moisture movement, the HELP model is a deterministic computer-based model that uses climatologic and soil data to compute the leachate flow rates in a landfill.

Application of flow models to simulate and estimate unsteady, unsaturated flows in solid waste landfills is still at an early stage. The governing equation used to characterize the leachate accretion on the leachate mound in the unsaturated zone is a second order partial differential equation. There are few analytical solutions for the unsaturated moisture-flow equation due to the nonlinearity and heterogeneities involved in the properties (for example hydraulic conductivity and diffusivity) of the landfill wastes. Exact analytical solutions can be obtained by making simplifying assumptions regarding soil moisture characteristics and the flow domain. Analytical solutions are described by Philip (1968, 1971), Wooding (1968), Lomen and Warrick (1974), Warrick (1975), and Batu (1982).

Since the irregularities in the domain and the complexities in the moisture-flow equation limit the representation of the real flow field by analytical models, many researchers have attempted to develop numerical models. Abbott (1982) described the numerical solution of one-dimensional unsaturated moisture-flow equation by direct explicit schemes for which a stability criteria has to be satisfied to compute the moisture content. A time-varying, one-dimensional leachate flow model was developed by Korfiatis (1984) using the finite-difference numerical technique. Demetracopoulos et al. (1986) described the implicit finite-difference scheme for the one-dimensional unsteady state moisture-flow equation which is solved to obtain the leachate accretion on the leachate mound. The implicit finite-difference expression for the moisture-flow equation was solved by the successive application of the Gaussian elimination method. Korfiatis et al. (1984) performed an experimental investigation with a small solid waste column, which was used to calibrate and verify the unsteady one-dimensional leachate flow model. The measured and predicted results were reasonably close to each other (Korfiatis et al., 1984).

McKeon and Chu (1987) described a steady state numerical model to compute suction head in unsaturated soils by the multigrid method. This method adopts an iterative solution technique that approximates the differential equation at various grid levels. The implicit finite-difference scheme for one-dimensional unsaturated flow were also described by Wright and Miller (1982). The model developed by Wright and Miller (1982) predicts suction head and leakage through landfill clay cover.

A Finite Element Model of Waste (FEMWASTE) transport was developed by Yeh and Ward (1981) that predicts the fate and transport of contaminants through saturated-unsaturated porous media. The finite element technique was also used by Yeh (1987) to solve the saturated-unsaturated flow problems through porous media. The three-dimensional Finite Element Model of Water (FEMWATER) flow through saturated-unsaturated porous media, developed by Yeh (1987) does not represent the specific flow problems related to the leachate flow in a landfill. The model does not predict surface runoff and evapotranspiration that have considerable effects on the leachate flow in a landfill.

To obtain accuracy and efficiency in numerical modeling, considerable research is being done for the development of high-quality numerical simulation techniques. In this study, the boundary integral method is employed to simulate leachate transport in the unsaturated zone of a landfill. The application of the boundary integral method to flow in porous media was first carried out by Liggett (1977), Liu and Liggett (1979), and Lennon et. al. (1980). These problems are governed by the Laplace equation for the velocity potential with a kinematic boundary condition on the free surface. The boundary integral equation for a moving interface between two fluids in porous media was solved by Liu et. al. (1981).

Taigbenu et al. (1984) used the boundary integral method to solve problems of seawater intrusion into a freshwater aquifer. Taigbenu and Liggett (1986) developed boundary integral model for contaminant transport in groundwater system. Taigbenu and Liggett (1986) also developed boundary integral method to solve Boussinesq equation for flow in an unconfined aquifer. Liggett and Liu (1979) developed two boundary integral formulations, Laplace transform solution and direct Green's function solution, to model unsteady flow in confined aquifers. Taigbenu and Liggett (1985) used the time-dependent Green's function to model groundwater flow problems considering a dimensionless diffusion equation.

Heterogeneity in the groundwater flow problems were investigated by Lafe et. al. (1981) by dividing the aquifer into a three-dimensional grid of zones. Zoning technique was employed to subdivide the domain into piecewise homogeneous regions. Cheng (1984) also developed a boundary integral equation method for Darcy's law with variable permeability. The formulation was done using the Green's function that varies with the permeability. A perturbation boundary element code was developed by Lafe and Cheng (1987) for the solution of the steady state groundwater equation in a heterogeneous media. Aral and Tang (1988) developed a numerical time integration procedure for the boundary element solution of the time-dependent anisotropic aquifer problems.

Cheng and Liggett (1984) developed boundary integral formulations for the solution of problems related to fracture propagation and consolidation of fluid-infiltrated porous media. The boundary element model for porous-elasticity was used to simulate the consolidation of partially saturated soil. The model results revealed that a small reduction in the degree of saturation would drastically alter the consolidation and pore pressure distribution of partially saturated soils. The model would be useful for simulating the problems incorporating the effects of fracture and consolidation in porous media flow.

The aforementioned applications of the boundary integral method were mostly used to solve saturated flow problems in the groundwater system. So far, the boundary integral equation method for unsteady flow of moisture transport in the unsaturated zone of a landfill has not been reported in the literature.

1.3 OBJECTIVE

The objectives of this research are:

1. To develop a two-dimensional unsteady state leachate flow model capable of simulating leachate flow rates in space and time. The model is developed with the capability to predict the unsteady variation of leachate mound-head inside the landfill;

2. To develop a runoff submodel that is incorporated into the main two-dimensional leachate flow model by considering the effect of side slope on the total runoff that occurs from a landfill;

3. To investigate the time-varying net infiltration rate that causes the build up of leachate mound at the bottom of the landfill;

4. To investigate the effect of finite-difference and boundary integral method in solving the moisture-flow equation in the unsaturated zone of a landfill. Results obtained by these techniques are compared with the simplified analytical solutions; and

5. To apply the two-dimensional unsteady state finite-difference FILL model to an example landfill section to demonstrate its effectiveness in predicting evapotranspiration, runoff, net infiltration, recharge, lateral and vertical flow rates in a landfill. Sensitivity analysis is also carried out in order to evaluate the influence of input parameters upon the model performance.

CHAPTER 2

LEACHATE FLOW PROCESS

This chapter presents the governing equations defining the leachate flow process causing the development of leachate mound in a landfill. Formulation of the FILL model along with the boundary conditions are described. The input to the upper boundary of the solid waste landfill is the precipitation less runoff and evapotranspiration. Runoff occurs at the top of the landfill and depends on the slope, roughness, hydraulic conductivity, and moisture content. The portion of the precipitation that enters into the landfill surface is the infiltration which is dependent on the depth of flow on the surface and also on the hydraulic conductivity and moisture content of the top soil. The moisture flow in the landfill basically occurs in three phases - flow through the cover, flow through the wastes, and flow through the bottom liner or virgin ground beneath the wastes as shown in Figure 2-1. The fraction of infiltration that enters into the unsaturated zone in a landfill changes the moisture content and thus leads to a build-up of leachate to form a saturated leachate mound. Then the lateral and vertical flows can be estimated when the hydraulic gradient of the leachate mound-head in the saturated zone is known.

The movement of moisture inside a landfill can be estimated by using a simple one-dimensional model. In this case, it is assumed that the flow in one vertical

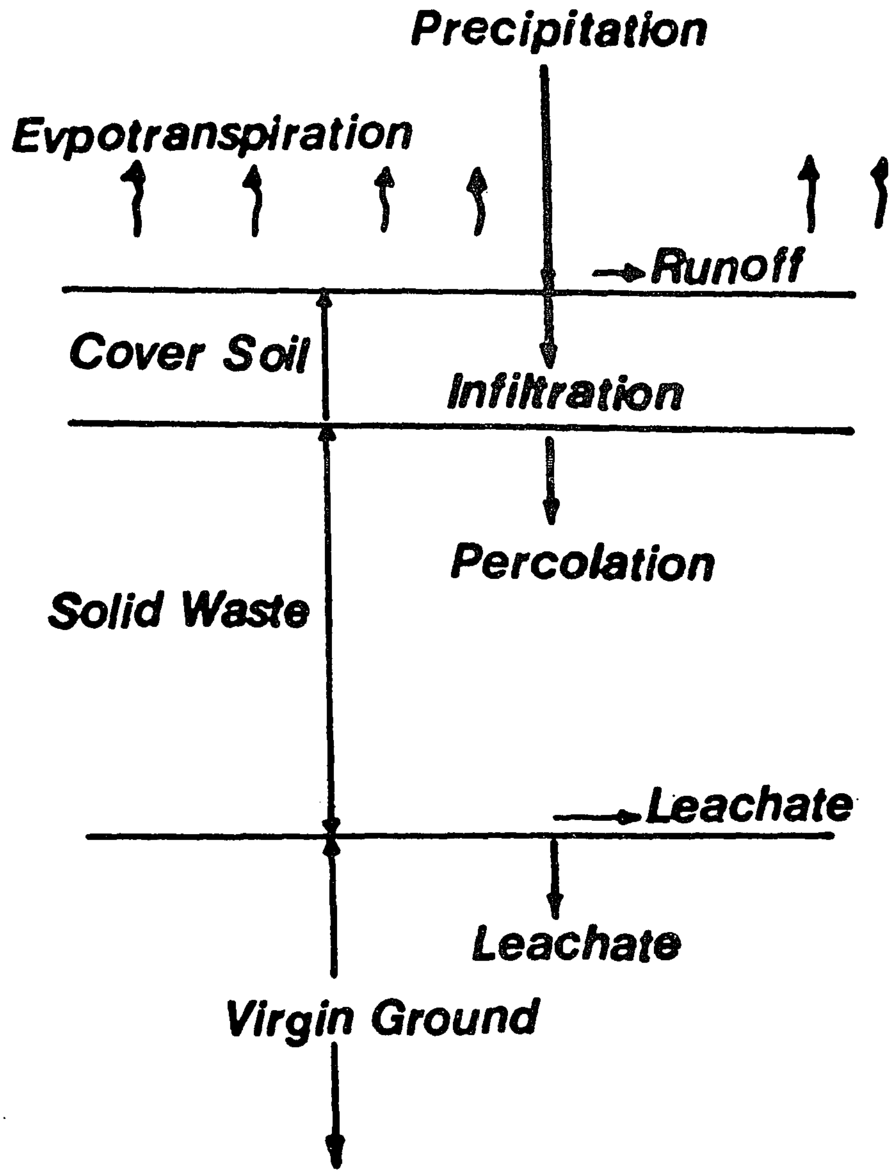


Figure 2-1. Configuration of Leachate Flow Process in a Landfill.

column of a landfill is not dependent on the changes in the moisture content in other parts of the landfill. However, moisture flow in one column depends on changes in the moisture content in the other columns of a landfill. These effects can be taken into account by considering a two-dimensional flow condition. Since the leachate flow varies in both space and time, an unsteady state two-dimensional flow with time-varying boundary conditions can be considered to predict the time-history of leachate flow inside a landfill for the real field simulation.

2.1 GOVERNING EQUATION IN THE UNSATURATED ZONE

The unsteady moisture-flow in the unsaturated zone can be formulated by using Darcy's Law and the equation of continuity. Darcy's Law was formulated for flow through a saturated porous media where hydraulic conductivity is a function of the pore properties of the medium and the properties of the fluid. Buckingham (1907) established that Darcy's Law might apply to the liquid flow in the unsaturated media in a modified form in which the hydraulic conductivity K, instead of being a constant, is regarded as a function of the volumetric moisture content. The extension of Darcy's Law was verified later by several investigators (Gardner, 1958; Childs and Collis-George, 1959; Brooks and Corey, 1964; Brutsaert, 1967; Klute, 1972; Mualem, 1976; and Clapp and Hornberger, 1978) and it is now a generally accepted theory for unsaturated liquid flow. According to Darcy's Law, the specific discharge in the unsaturated zone is:

$$q = -K(\theta)\nabla\phi \tag{2-1}$$

where,

q = volume flow rate per unit cross-sectional area normal to the direction of flow (L/T),

ϕ = total potential (L),

$K(\theta)$ = unsaturated hydraulic conductivity (L/T),

$\theta = V_w/V$ = volumetric moisture content (vol/vol),

V_w = volume of water in the soil mass (L^3), and

V = total volume of the soil mass (L^3).

The total potential head ϕ is illustrated in Figure 2-2 which shows the negative suction head developed in the unsaturated zone. Figure 2-2 shows that the total potential head is:

$$\phi = -z - \Delta \tag{2-2}$$

For unsaturated soil Δ is the suction head and considering $\Delta = \psi$, the total potential head ϕ is:

$$\phi = -\Psi - z \tag{2-3}$$

The continuity equation can be written as (Willis and Yeh, 1987):

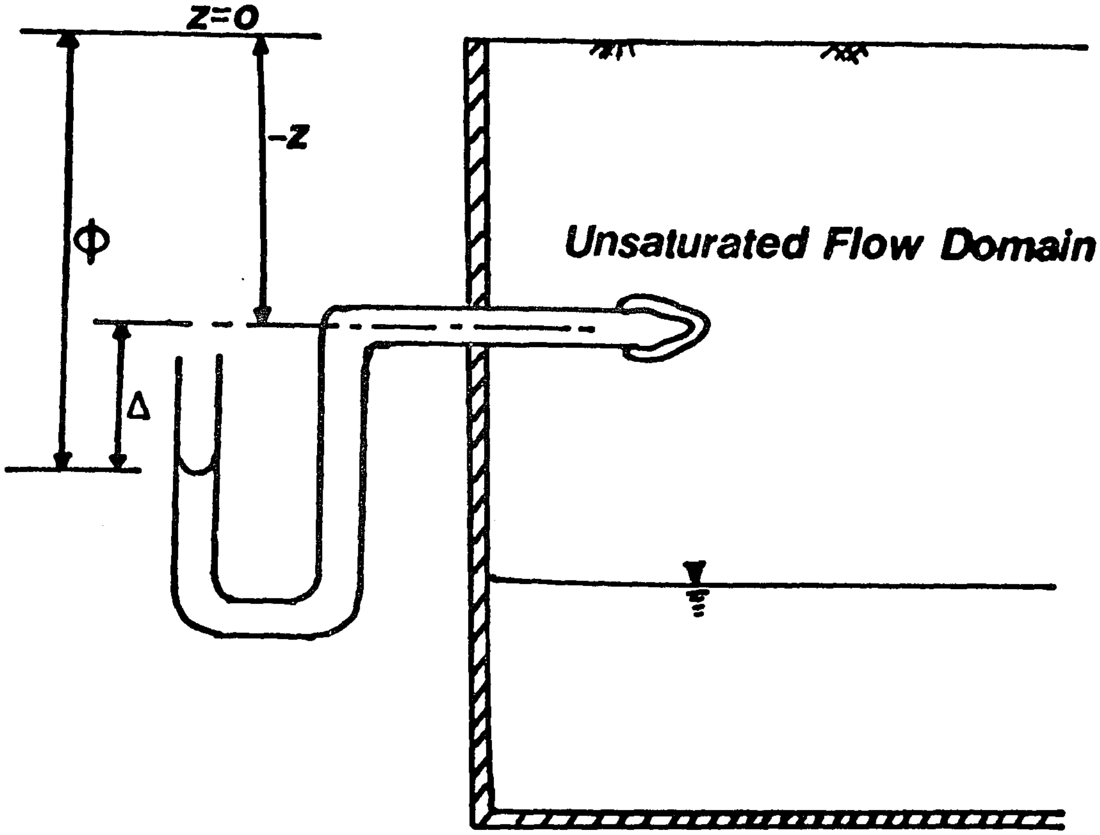


Figure 2-2. Definition Diagram for Total Potential (Bear, 1979).

$$-\nabla.(\rho q) = \frac{\partial(\rho\theta)}{\partial t} \qquad (2\text{-}4)$$

where,

ρ = density of water.

Assuming incompressible flow:

$$-\nabla.q = \frac{\partial\theta}{\partial t} \qquad (2\text{-}5)$$

From equations (2-1), (2-3), and (2-5):

$$-\nabla.K(\theta)\nabla\psi - \frac{\partial K(\theta)}{\partial z} = \frac{\partial\theta}{\partial t} \qquad (2\text{-}6)$$

Equation (2-6) can be written as:

$$\frac{\partial\theta}{\partial t} = \frac{\partial}{\partial x}\left[-K(\theta)\frac{\partial\psi}{\partial x}\right] + \frac{\partial}{\partial y}\left[-K(\theta)\frac{\partial\psi}{\partial y}\right] + \frac{\partial}{\partial z}\left[-K(\theta)\frac{\partial\psi}{\partial z}\right] - \frac{\partial K(\theta)}{\partial z} \qquad (2\text{-}7)$$

Considering a two-dimensional flow domain, equation (2-7) can be written as:

$$\frac{\partial\theta}{\partial t} = \frac{\partial}{\partial x}\left[-K(\theta)\frac{\partial\psi}{\partial x}\right] + \frac{\partial}{\partial z}\left[-K(\theta)\frac{\partial\psi}{\partial z}\right] - \frac{\partial K(\theta)}{\partial z} \qquad (2\text{-}8)$$

Assuming ψ as the single-valued function of θ, equation (2-8) can be written as:

$$\frac{\partial\theta}{\partial t} = \frac{\partial}{\partial x}\left[-K(\theta)\,\frac{\partial\psi}{\partial\theta}\,\frac{\partial\theta}{\partial x}\right] + \frac{\partial}{\partial z}\left[-K(\theta)\,\frac{\partial\psi}{\partial\theta}\,\frac{\partial\theta}{\partial z}\right] - \frac{\partial K(\theta)}{\partial z} \tag{2-9}$$

The soil moisture diffusivity $D(\theta)$ can be written as:

$$D(\theta) = -K(\theta)\,\frac{\partial\psi}{\partial\theta} \tag{2-10}$$

Therefore, the two-dimensional unsaturated moisture flow equation can be written as:

$$\frac{\partial\theta}{\partial t} = \frac{\partial}{\partial x}\left[D(\theta)\,\frac{\partial\theta}{\partial x}\right] + \frac{\partial}{\partial z}\left[D(\theta)\,\frac{\partial\theta}{\partial z}\right] - \frac{\partial K(\theta)}{\partial z} \tag{2-11}$$

This equation is solved to obtain the moisture content in the unsaturated zone of a landfill. The solution of moisture content in the unsaturated zone generates the time-history of leachate input rate on the saturated leachate mound.

2.1.1 Hydraulic Conductivity and Diffusivity Functions

The values of the unsaturated hydraulic conductivity vary with the moisture content in the soil. In the unsaturated media, the hydraulic conductivity is expressed in a modified form as a function of the volumetric moisture content. The unsaturated hydraulic conductivity as a function of moisture content θ can be expressed as (Campbell, 1974):

$$K(\theta) = K_s\left(\frac{\theta}{\theta_s}\right)^{2b+3} \tag{2-12}$$

where,

$K(\theta)$= unsaturated hydraulic conductivity as a function of moisture content (L/T),

θ_s= saturation moisture content (vol/vol),

K_s= saturated hydraulic conductivity (L/T),

b= slope of moisture characteristic curve plotted on log-log paper and is known as Campbell's parameter.

From equation (2-10), it is clear that to determine the diffusivity coefficient D(θ), the suction head has to be differentiated with respect to moisture content θ. The suction head ψ as a function of moisture content θ can be expressed as (Bristow and Williams, 1987):

$$\psi(\theta) = \psi_e\left(\frac{\theta}{\theta_s}\right)^{-b} \tag{2-13}$$

where,

$\psi(\theta)$ = suction head as a function of moisture content θ,

ψ_e = air entry suction head, and

b is the same as defined before.

Reasonable estimates for ψ_e can be obtained from the moisture characteristic curves. In practice, moisture content at air entry $\theta_e = 0.9\theta_s$ and in general, θ_e varies between $0.8\theta_s$ and θ_s (Rogowski, 1971). Therefore, air entry suction head ψ_e corresponds to the suction head value for 90 percent moisture content in the soil matrix.

From equations (2-10), (2-12), and (2-13), the diffusivity relationship in the unsaturated zone can be expressed as:

$$D(\theta) = bK(\theta)\left(\frac{\psi_e}{\theta_s}\right)\left(\frac{\theta_s}{\theta}\right)^{b+1} \tag{2-14}$$

The above relationships for unsaturated diffusivity and hydraulic conductivity are used in the unsaturated moisture-flow equation to compute the moisture content and hence the moisture-flow rate in the unsaturated zone.

2.2 BOUNDARY AND INITIAL CONDITIONS

Boundary conditions define the dependent variable at the boundaries of the domain. In the FILL model, two types of upper boundary conditions are considered, the unsaturated condition and the saturated condition. For the unsaturated flow condition, the moisture-flux is composed of precipitation less evapotranspiration and runoff. This condition at the upper boundary occurs when net precipitation intensity is less than the saturated hydraulic conductivity (Demetracopoulos et al., 1986). The precipitation intensity that produces runoff is known as the net precipitation intensity. Therefore, the input at the upper boundary for the unsaturated flow condition can be written as:

$$INPUT = P - ET - R \tag{2-15}$$

where,

P = precipitation (L),

ET = evapotranspiration (L), and

R = runoff (L).

The saturated boundary condition exists for the upper surface of the landfill when the net precipitation intensity is greater than or equal to the saturated hydraulic conductivity (Demetracopoulos et al., 1986). In this situation, the upper boundary condition is expressed as:

$$\theta = \theta_s \tag{2-16}$$

where,

θ_s = moisture content at saturation.

At the lower boundary, a free-drainage condition is considered, which is expressed as (Korfiatis et al., 1984):

$$\frac{\partial \theta}{\partial x} = 0 \qquad (2\text{-}17a)$$

$$\frac{\partial \theta}{\partial z} = 0 \qquad (2\text{-}17b)$$

The initial condition is defined as the known values of moisture content in the landfill section. Therefore, at time t=0, the initial water contents are:

$$\theta(x,z,0) = g(x,z) \qquad (2\text{-}18)$$

where g(x,z) is the known volumetric water content at all points in the x-z plane.

2.3 RUNOFF

Conservation of mass requires that the change in water level (H) with time, equals the sum of the negative of the change of flow rate with distance and difference of rainfall and infiltration. This can be expressed as [(Brass, 1990; Chow et al., 1988; and Hillel, 1980), Figure 2-3]:

$$\frac{\partial Q}{\partial x} + \frac{\partial H}{\partial t} = \overline{q} \qquad (2\text{-}19)$$

where,

Q = overland flow rate (L^3/T),

H = depth of runoff (L),

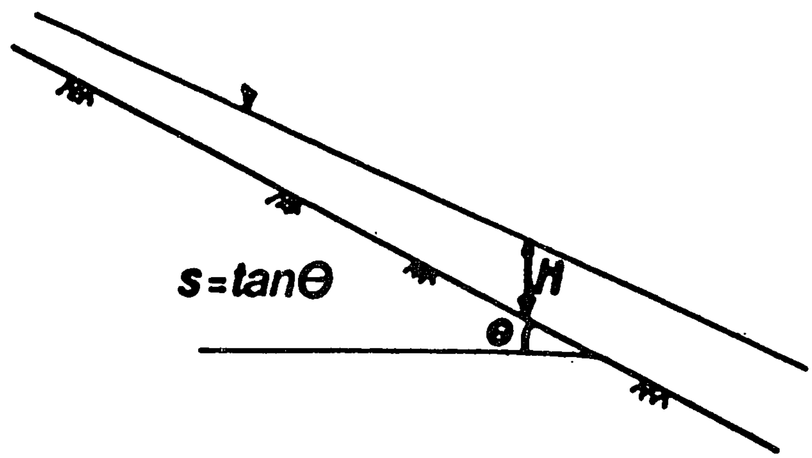


Figure 2-3. A Schematic Representation of Runoff along an Inclined Surface.

$\bar{q} = P - f$

P = rainfall rate (L/T), and

f = infiltration rate (L/T).

For overland flow Q is expressed as (Chow et al., 1988):

$$Q = \frac{1.49}{n} H^{\frac{5}{3}} S^{\frac{1}{2}} \tag{2-20}$$

where,

n = Manning's roughness coefficient, and

S = Slope of the surface.

From equation (2-20), the depth of overland flow can be written as:

$$H = \alpha Q^{\beta} \tag{2-21}$$

where,

$$\alpha = \left(\frac{n}{1.49\sqrt{S}}\right)^{\beta}$$

and

$\beta = 0.6$.

From equations (2-19) and (2-21), the kinematic wave equation is expressed as

(Chow et al., 1988):

$$\frac{\partial Q}{\partial x} + \alpha\beta Q^{\beta-1}\frac{\partial Q}{\partial t} = \overline{q} \qquad (2\text{-}22)$$

When capillary forces are dominant, the cumulative infiltration equation can be expressed as (Haverkamp et al., 1987):

$$I(t) = S_o(t)^{1/2} \qquad (2\text{-}23)$$

From which the infiltration capacity f_c can be expressed as:

$$f_c = \frac{dI}{dt} = \frac{S_o}{2}(t)^{-1/2} \qquad (2\text{-}24)$$

where,

$I(t)$= cumulative infiltration per unit surface,

S_o= sorptivity, reflecting the capacity of soil to absorb water by simple capillary forces, and

t= time.

Adjustment in the value of S_o for changing boundary condition has been made by the expression developed by Philip (1969):

$$S_o = [2K_s(\theta_s - \theta_i)(H - h_f)]^{1/2} \qquad (2\text{-}25)$$

where,

K_s= hydraulic conductivity at saturation (L/T),

θ_s= moisture content at saturation (vol/vol),

θ_i= initial moisture content (vol/vol),

H= depth of water at the soil surface (L), and

h_f= the pressure value defining wetting front, which is constant in space and time (L).

Yu-Si (1987) defined h_f as the capillary head $-h_c$ at the wetting front in the expression for sorptivity S_o. Thus, equation (2-25) can be written as:

$$S_o = [2K_s(\theta_s - \theta_i)(H + h_c)]^{1/2} \qquad (2\text{-}26)$$

This equation for sorptivity is used to compute the rate of infiltration along the upper boundary of the landfill. The value of the sorptivity from equation (2-26) is used in equation (2-24) at different time steps to compute the infiltration capacity. If rainfall intensity is higher than the infiltration capacity, the infiltration rate is considered the same as the infiltration capacity otherwise the total amount of rainfall enters into the soil surface. The infiltration rate is subtracted from the rainfall rate (as shown in equation 2-19), which is called the rainfall excess and this is the source term for the kinematic wave equation (2-22). In the kinematic wave equation (2-22), the right hand side, i.e. rainfall excess is known and the value of the depth of runoff is computed at different positions on the landfill surface considering the slope and surface roughness.

2.4 EVAPOTRANSPIRATION

The modified Penman method is used to compute potential evapotranspiration as developed by Ritchie (1972) and used in CREAMS (Knisel, 1980). The expression for the potential evapotranspiration is (Schroeder et al., 1984):

$$E_{oi} = \frac{1.28 A_i R_i}{(A_i + G)\,25.4} \qquad (2\text{-}27)$$

where,

E_{oi} = potential evapotranspiration on day i (inches),

A_i = slope of saturation vapor pressure curve on day i,

R_i = net solar radiation on day i (langleys), and

G = psychrometric constant which is assumed to remain constant at 0.68.

Evapotranspiration of available surface water occurs first. After available surface water is depleted, water from the soil column through evaporation and plant transpiration occurs. Evaporation from surface moisture is given by (Schroeder et al., 1984):

$$ESS_i = \begin{cases} E_{oi} & \text{for } E_{oi} \le SNO_{i-1} + P_i \\ SNO_{i-1} + P_i & \text{for } E_{oi} > SNO_{i-1} \end{cases} \qquad (2\text{-}28)$$

where,

ESS_i = surface water evaporation on day i (inches),

SNO_{i-1} = amount of snow at the end of day i-1, and

P_i = actual precipitation on day i (inches).

In the presence of actively transpiring plants, the potential soil evaporation is computed as follows (Schroeder et al., 1984):

$$ES_{oi} = E_{oi}e^{-0.4LAI} \qquad (2\text{-}29)$$

where,

ES_{Oi} = potential soil evaporation on day i (inches),

LAI = leaf area index on day i, and E_{Oi} as defined previously.

The leaf area index is a factor that indicates the vegetative cover for a landfill area and is defined as the ratio of the total leaf area to the soil surface area. During the nongrowing or dormant period, the dead or dormant vegetative cover reduces the potential soil evaporation. Then the potential soil evaporation for winter cover is expressed as (Schroeder et al., 1984):

$$ES_{oi} = E_{oi}e^{-0.4WCF} \qquad (2\text{-}30)$$

where,

WCF = winter cover factor.

Soil evaporation occurs in two stages. In stage-one, soil evaporation is equal to the potential soil evaporation ES_{0i}. In stage-two, evaporation is controlled by the rate at which water can be transmitted through the soil.

An upper limit is set to control the stage-one soil evaporation. The upper limit to the stage-one evaporation is expressed as (Knisel, 1980):

$$U = \frac{\left[9\,(a_s - 3)^{0.42}\right]}{25.4} \qquad (2\text{-}31)$$

where,

U = upper limit for stage-one evaporation (inches) and

a_s = soil transmissivity parameter for evaporation (mm/$\sqrt{}$day).

The total of soil evaporation less infiltration is:

$$ES1T_i = \sum_{k=m}^{i} (ES_k - IN_k) \qquad (2\text{-}32)$$

where,

ES_k = soil evaporation on day k (inches),

m = last day when $ES1T_i$ equalled zero, and

IN_k = precipitation - surface runoff - surface water evaporation (inches).

When $ES1T_i$ is greater than U, stage-one evaporation stops and stage-two evaporation starts. Stage-two evaporation is controlled by soil transmissivity.

Stage-two evaporation from the soil is estimated as (Knisel, 1980):

$$ES2_i = \frac{1}{25.4}\left[\sqrt{t_i} - \sqrt{(t_i - 1)}\right] \quad (2-33)$$

where,

$ES2_i$ = stage-two soil evaporation for day i (inches), and

t_i = days since stage-one evaporation ended.

The potential plant transpiration is computed as (Schroeder et al., 1984):

$$EP_{oi} = \frac{E_{oi} LAI_i}{3.0} \quad (2-34)$$

where,

EP_{Oi} = potential plant transpiration on day i (inches).

The actual plant transpiration demand is expressed as (Schroeder et al., 1984):

$$EPD_i = \begin{cases} EP_{oi} & \text{for } EP_{oi} + ESS_i + ES_i \le E_{oi} \\ E_{oi} - ESS_i - ES_i & \text{for } EP_{oi} + ESS_i + ES_i > E_{oi} \end{cases} \quad (2-35)$$

where,

ES_i = actual soil evaporation on day i (inches).

Using the relationship developed by Shanholtz and Lillard (1970), the actual plant transpiration is computed as:

$$EP_i = EPD_i\left[1.2 - 4EPD_i + \frac{SM_i - WP}{FC - WP}\right] \tag{2-36}$$

where,

SM_i = soil moisture on day i (inches),

WP = wilting point (VOL/VOL), and

FC = field capacity (VOL/VOL).

Field capacity is defined as the amount of soil moisture content remaining in a unit volume of soil after a prolonged period of gravity drainage. Wilting point is defined as the lowest soil moisture content that can be achieved by plant transpiration.

The total evapotranspiration (ET) is equal to the sum of the evaporation from the soil (ES) and the plant transpiration (EP):

$$ET_i = \begin{cases} ES_i + EP_i & \text{when } (ES_i + EP_i) \le (E_{oi} - ESS_i) \\ E_{oi} - ESS_i & \text{when } (ES_i + EP_i) > (E_{oi} - ESS_i) \end{cases} \tag{2-37}$$

2.5 SNOWMELT

If the mean daily temperature is below 32°F, the precipitation is stored on the surface as snow. When the temperature exceeds 32°F, snow continues to melt and contributes to the infiltration and/or runoff. The amount of snowmelt is added to the rainfall for use in equation (2-22). The amount of snowmelt without rain is computed as (Viessman et al., 1977):

$$M = K'(1-F)(0.004I_i)(1-A) + E(0.0084V)\{0.22(T_a-32) + (0.78(T_d-32)\} + 0.029F(T-32) \quad (2\text{-}38)$$

where,

M = melt, inches/day,

K'= melting coefficient, $0.9 < K' < 1.1$

F = forest cover factor,

I_i= solar radiation, langleys/day,

A = albedo (reflection coefficient),

E = exposure coefficient $0.3 < K < 1.0$,

V = wind velocity (mph),

T_a= air temperature °F, and

T_d= dew point temperature °F.

If there is rain on snow, the equation for snowmelt is (Wanielista, 1990):

$$M = 0.09 + (0.029 + 0.084EV + 0.007P)(T_a-32) \quad (2\text{-}39)$$

where,

V = wind velocity (mph),

P = rainfall (inches per day), and

T_a = air temperature, °F.

2.6 GOVERNING EQUATION IN THE SATURATED ZONE

The leachate head at the bottom of a landfill develops due to the leachate input from the unsaturated zone to the mound. The hydraulics of the leachate mound at the bottom of a landfill has been studied by Korfiatis et al. (1984), Schroeder et al. (1984) and Demetracopoulos (1988). The submodel to compute the leachate head can be formulated by applying Darcy's Law and continuity equation in the saturated zone as shown in Figure 2-4. According to Darcy's Law, the lateral drainage in the saturated zone for an inclined bottom can be written as (Bear, 1972):

$$q_L = -K_s h \frac{\partial z}{\partial x} \qquad (2\text{-}40)$$

where,

K_s = saturated hydraulic conductivity (L/T),

h = saturated thickness of the landfill section (L), and

z = elevation of water table from a reference datum (L).

Figure 2-4 shows that:

$$z = h + d$$

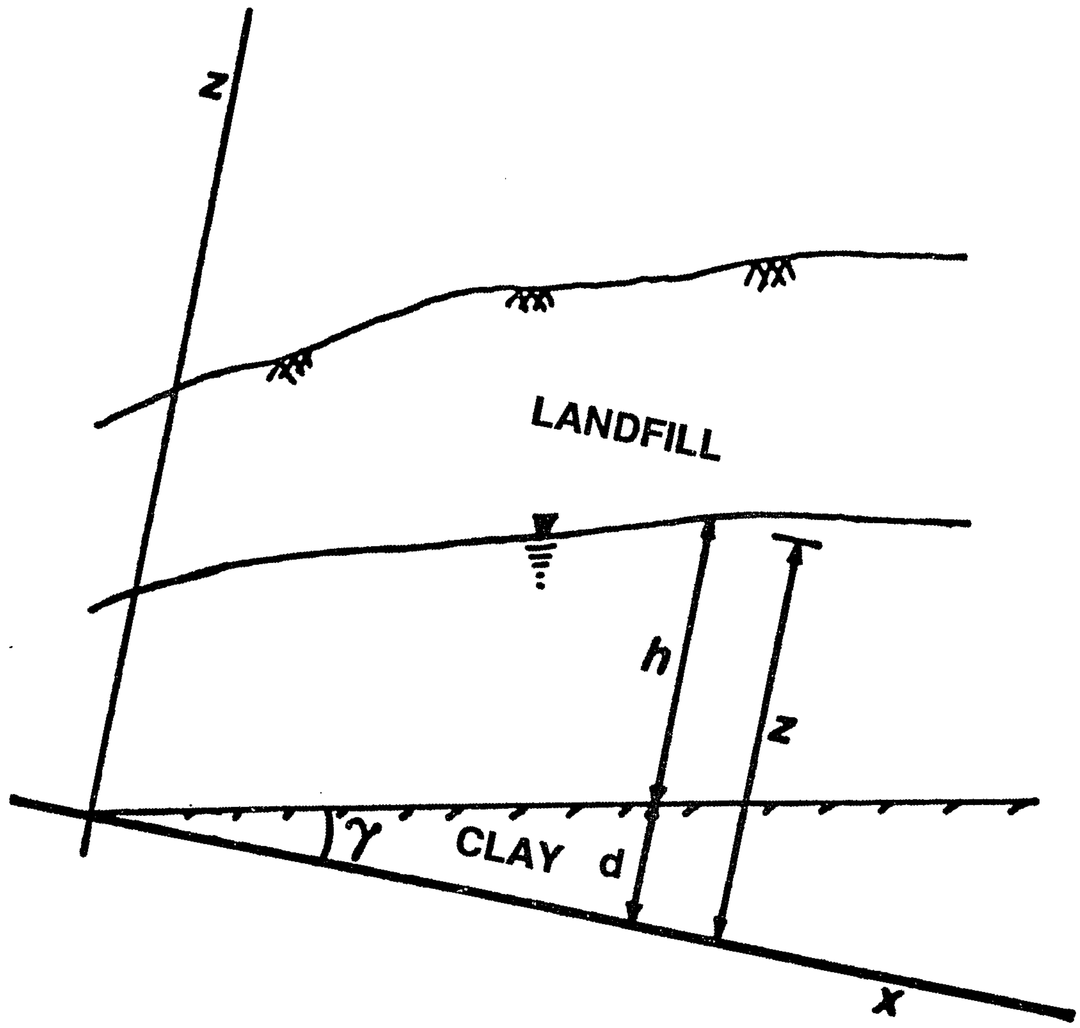


Figure 2-4. Saturated Flow at the Bottom of a Landfill.

or, $z = h + x \tan\gamma$

From which:

$$\frac{\partial z}{\partial x} = \frac{\partial h}{\partial x} + \tan\gamma$$

Therefore, the lateral drainage can be written as:

$$q_L = -K_s h\left[\frac{\partial h}{\partial x} + \tan\gamma\right] \tag{2-41}$$

Assuming a fully saturated bottom liner, the leakage rate can be expressed as (Korfiatis and Demetracopoulos, 1986):

$$q_v = K_c\left(\frac{h + d}{d}\right) \tag{2-42}$$

where,

K_c = saturated hydraulic conductivity of the clay liner (L/T), and

d = thickness of clay at the bottom of the landfill (L).

The mass conservation principle states that the sum of all flow rates into the control volume must equal the rate of volume change in the control volume. This yields (Korfiatis and Demetracopoulos, 1986):

$$-\frac{\partial q_L}{\partial x} + N_\ell - K_c\left(\frac{h+d}{d}\right) = n_e\frac{\partial h}{\partial t} \tag{2-43}$$

where,

N_l = leachate accretion onto the drainage layer (L/T),

n_e = effective porosity of the solid waste material,

Substituting equation (2-41) into equation (2-43):

$$\frac{\partial}{\partial x}\left[K_s h\left(\frac{\partial h}{\partial x} + \tan\gamma\right)\right] + N_\ell - K_c\left(\frac{h+d}{d}\right) = n_e\frac{\partial h}{\partial t} \tag{2-44}$$

Equation (2-44) is solved to obtain the leachate head at the bottom of the landfill which is used to compute lateral and vertical flow of leachate from the landfill to the perimeter or collection system.

After solving equation (2-11) for soil moisture content, the leachate accretion or drainage to the saturated leachate mound is computed. The governing equation for the leachate accretion is expressed as (Korfiatis et al., 1984):

$$N_\ell = K(\theta) - D(\theta)\frac{\partial\theta}{\partial z} \tag{2-45}$$

The major uncertainties in making more accurate estimate of leachate accretion to the saturated leachate mound are in the material properties such as the coefficients related to the soil moisture diffusivity $D(\theta)$ and hydraulic conductivity $K(\theta)$. For this reason, after computing the soil moisture content, the water balance is checked for change in the moisture contents by considering the net infiltration (at the boundary), inflow and outflow due to the diffusivity and the unsaturated hydraulic conductivity in the domain. The drainage rate to the saturated leachate mound computed by the governing equation (2-45) is compared with the drainage rates obtained by the water balance in the domain and by using empirical formulas described by Bras (1990). An appropriate value is estimated by assuming that the recharge to the saturated leachate mound can not exceed the total net infiltration over a period of time.

A common empirical form of the percolation rate is expressed as (Bras, 1990):

$$N_\ell = a_s \theta^{b_s} \tag{2-46}$$

where,

$a_s = K_s \psi_e$

b_s = a calibration parameter.

If water content is between field capacity and saturation, the soil moisture drainage rate is considered as (Bras, 1990):

$$N_\ell = f_c \left(1 - \frac{\theta_s - \theta_i}{\theta_s - \theta_c}\right)^3 \tag{2-47}$$

where,

f_c = minimum infiltration rate (L/T),

θ_i = initial soil moisture content (VOL/VOL), and

θ_c = field capacity moisture content (VOL/VOL).

CHAPTER 3

FINITE-DIFFERENCE FORMULATION

The governing partial differential equations incorporated into the two-dimensional leachate flow model, represent runoff at the upper boundary, moisture-flow in the unsaturated zone along with the boundary conditions, and movement of leachate at the bottom of a landfill. All these equations are discretized using the finite-difference technique to obtain a system of algebraic equations. In this method, the derivatives are approximated in the finite grid points to transform the partial differential equation into a set of simultaneous algebraic equations. These equations are solved simultaneously for the dependent variables using numerical solution methods. Application of the finite-difference techniques is described for each of the governing partial differential equations in this chapter.

3.1 UNSATURATED MOISTURE-FLOW EQUATION

In the finite-difference method, the unsaturated moisture-flow equation (2-11) is discretized in x-z-t space. The horizontal and vertical lines in x- and z-directions are divided into segments of lengths of Δx and Δz respectively. These constitute rectangular nodal areas in the vertical plane having a nodal point at the centroid of the rectangles as shown in Figure 3-1. The time is divided into intervals of duration Δt. An implicit finite-difference scheme was employed by Demetracopoulos et al. (1986) to discretize the one-dimensional unsaturated moisture-flow equation. In the present model, the same concept of the finite-difference scheme is used to discretize the two-dimensional unsaturated moisture-flow equation (2-11). According to Figure 3-1, discretization of equation (2-11) for the (i,j)th grid point yields:

$$\frac{\theta_{i,j}^{k+1}-\theta_{i,j}^{k}}{\Delta t} = \frac{1}{(\Delta x)^2}[D_{i+1/2,j}(\theta_{i+1,j}-\theta_{i,j}) - D_{i-1/2,j}(\theta_{i,j}-\theta_{i-1,j})]$$

$$+ \frac{1}{(\Delta z)^2}[D_{i,j+1/2}(\theta_{i,j+1}-\theta_{i,j}) - D_{i,j-1/2}(\theta_{i,j}-\theta_{i,j-1})]$$

$$- \frac{1}{\Delta z}(K_{i,j+1/2}-K_{i,j-1/2}) \qquad (3\text{-}1)$$

where,

k = initial time level;

$k+1$ = forward time level;

$D_{i+1/2,j}$ = diffusivity between nodes i,j and i+1,j;

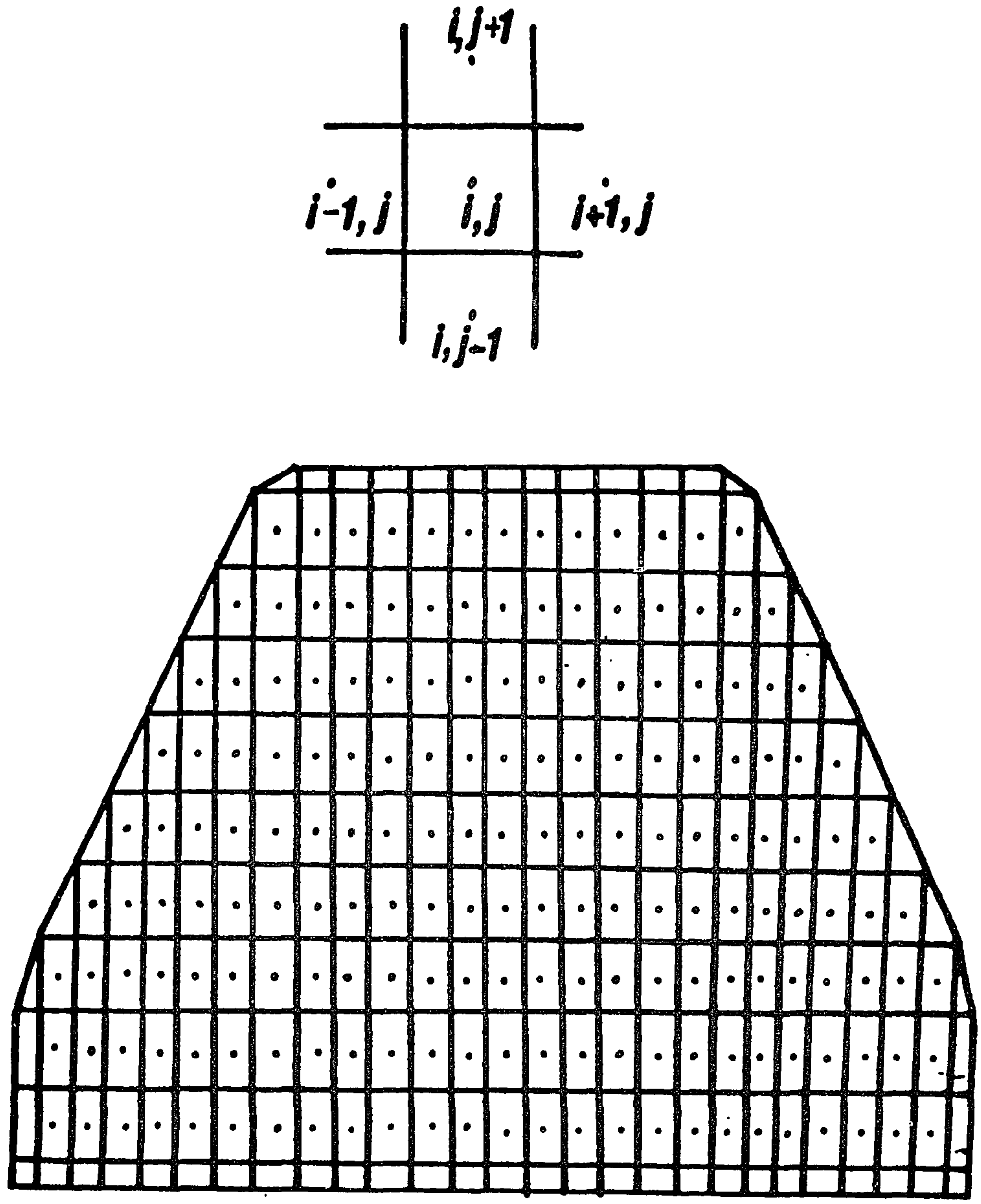
$D_{i-1/2,j}$ = diffusivity between nodes i,j and i-1,j;

$D_{i,j+1/2}$ = diffusivity between nodes i,j and i,j+1;

$D_{i,j-1/2}$ = diffusivity between nodes i,j and i,j-1;

$K_{i,j+1/2}$ = unsaturated hydraulic conductivity between nodes i,j and i,j+1; and

$K_{i,j-1/2}$ = unsaturated hydraulic conductivity between nodes i,j and i,j-1.



Figire 3-1. Finite-difference Grid Representation of a Landfill Section.

The dependent variable θ with the subscripts i,j; i+1,j; i-1,j; i,j+1; and i,j-1 defines the moisture content in the finite-difference grid-points. The diffusivity D, and hydraulic conductivity K in equation (3-1) are functions of the moisture content θ. The values of $D_{i+1/2,j}$, $D_{i-1/2,j}$, $D_{i,j+1/2}$, $D_{i,j-1/2}$, $K_{i,j+1/2}$, and $K_{i,j-1/2}$ at a particular time step are determined by the extrapolation from the previous time steps as described by Rubin and Steinhardt (1963). By evaluating $\theta_{i+1,j}$, $\theta_{i-1,j}$, $\theta_{i,j+1}$, and $\theta_{i,j-1}$ at the forward time level, the implicit finite-difference equation is obtained as follows:

$$\frac{\theta_{i,j}^{k+1}-\theta_{i,j}^{k}}{\Delta t} = \frac{1}{(\Delta x)^2}\left[D_{i+1/2,j}\left(\theta_{i+1,j}^{k+1}-\theta_{i,j}^{k+1}\right) - D_{i-1/2,j}\left(\theta_{i,j}^{k+1}-\theta_{i-1,j}^{k+1}\right)\right]$$
$$+ \frac{1}{(\Delta z)^2}\left[D_{i,j+1/2}\left(\theta_{i,j+1}^{k+1}-\theta_{i,j}^{k+1}\right) - D_{i,j-1/2}\left(\theta_{i,j}^{k+1}-\theta_{i,j-1}^{k+1}\right)\right]$$
$$- \frac{1}{\Delta z}\left(K_{i,j+1/2}-K_{i,j-1/2}\right) \qquad (3-2)$$

It is difficult to obtain a regular network of grid points having equal space intervals Δx and Δz throughout the domain of a landfill with irregular boundaries. In this case, it is necessary to develop special finite-difference formulas for irregular spacing of the grid points. Following the Taylor series expansion described by Remson et al. (1971), the implicit finite-difference expression for irregular grid points is obtained with respect to Figure 3-2 as:

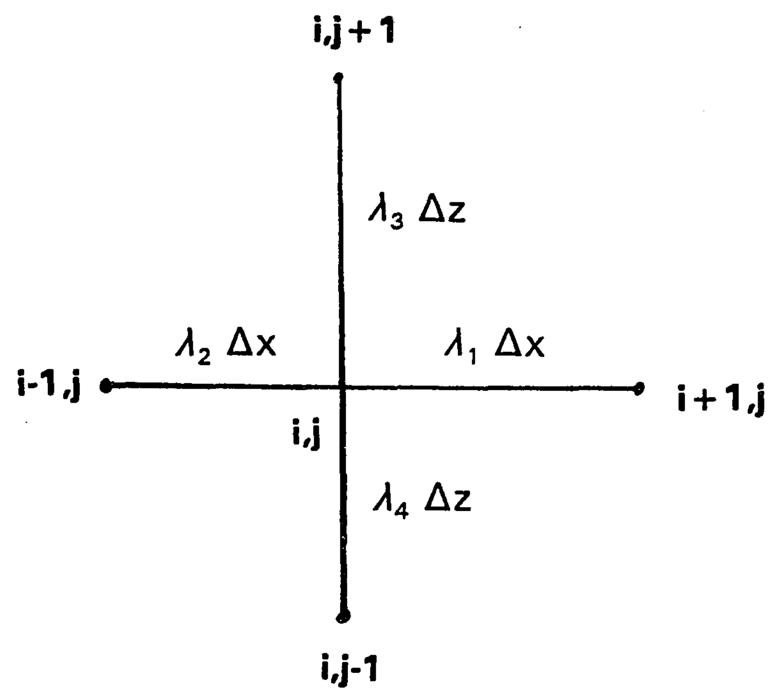


Figure 3-2. Irregular Finite-difference Grid Spacing.

$$\frac{\theta_{i,j}^{k+1}-\theta_{i,j}^{k}}{\Delta t} = \frac{2}{(\lambda_1+\lambda_2)\Delta x}\left[D_{i+1/2,j}\frac{\left(\theta_{i+1,j}^{k+1}-\theta_{i,j}^{k+1}\right)}{\lambda_1\Delta x} - D_{i-1/2,j}\frac{\left(\theta_{i,j}^{k+1}-\theta_{i-1,j}^{k+1}\right)}{\lambda_2\Delta x}\right]$$

$$+\frac{2}{(\lambda_3+\lambda_4)\Delta z}\left[D_{i,j+1/2}\frac{\left(\theta_{i,j+1}^{k+1}-\theta_{i,j}^{k+1}\right)}{\lambda_3\Delta z} - D_{i,j-1/2}\frac{\left(\theta_{i,j}^{k+1}-\theta_{i,j-1}^{k+1}\right)}{\lambda_4\Delta z}\right]$$

$$-2\left(\frac{K_{i,j+1/2}-K_{i,j-1/2}}{\lambda_3\Delta z+\lambda_4\Delta z}\right) \qquad (3-3)$$

The system of finite-difference expression developed, can be solved by the Gauss-Seidel iteration method. The iteration procedure is continued until two consecutive sets of moisture values agree within a prescribed tolerance criterion.

3.2 BOUNDARY CONDITIONS

The upper boundary of a landfill includes negligible slope at the top and steep slopes along the sides. The upper boundary condition for the boundaries at the top and at the side slope for unsaturated moisture-flow condition can be described by mathematical expression and then a finite-difference technique can be applied to get a system of algebraic equations to compute moisture values at the surface.

The upper boundary conditions at the side slope are defined by considering the flow of moisture from the boundary node (i,j) to the adjacent nodes (i+1,j) and (i,j+1) in both the horizontal and vertical directions respectively as shown in Figure 3-3.

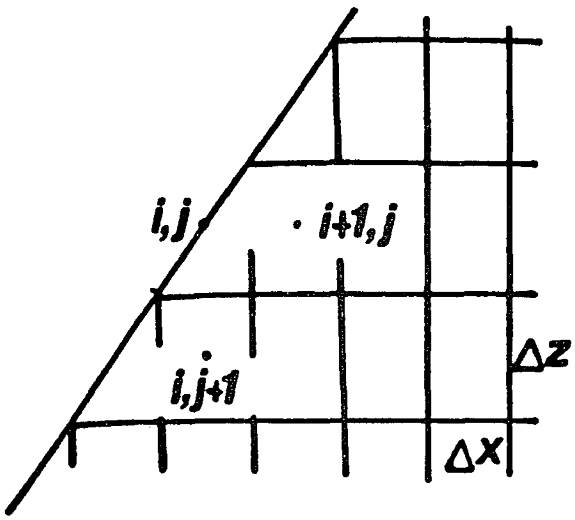


Figure 3-3. Boundary Node along the Upper Surface of a Landfill.

The unsaturated moisture flow equation (2-5) can be written as:

$$-\frac{\partial q}{\partial x} - \frac{\partial q}{\partial z} = \frac{\partial \theta}{\partial t} \tag{3-4}$$

Moisture flow in x-direction (q_x) can be written as:

$$q_x = -D(\theta)\frac{\partial \theta}{\partial x} \tag{3-5}$$

where,

$$D(\theta) = -K(\theta)\frac{\partial \psi}{\partial \theta}$$

ψ = suction head (L),

$D(\theta)$ = diffusivity (L^2/T), and

$K(\theta)$ = unsaturated hydraulic conductivity (L/T).

Equation (3-4) can be written for the boundary node (i,j) as:

$$-\frac{q_{i+1,j} - q_{i,j}}{\Delta x} - \frac{q_{i,j+1} - q_{i,j}}{\Delta z} = \frac{\theta_{i,j}^{k+1} - \theta_{i,j}^{k}}{\Delta t} \tag{3-6}$$

where,

$q_{i,j}$ = rainfall rate - (evapotranspiration rate + runoff rate)

= net infiltration (L/T).

From equation (3-5), the finite-difference expression $q_{i+1,j}$ can be written as:

$$q_{i+1,j} = - D_{i+1/2,j}\left(\frac{\theta_{i+1,j} - \theta_{i,j}}{\Delta x}\right) \tag{3-7}$$

From equation (2-1), the specific discharge in the z-direction in the unsaturated zone is:

$$q_z = K_z(\theta) - D(\theta)\frac{\partial\theta}{\partial z} \tag{3-8}$$

From which:

$$q_{i,j+1} = K_{i,j+1/2} - D_{i,j+1/2}\left(\frac{\theta_{i,j+1} - \theta_{i,j}}{\Delta z}\right) \tag{3-9}$$

Substituting (3-7) and (3-9) in (3-6), the implicit finite-difference expression for the unsaturated moisture-flow equation at the upper boundary is obtained as:

$$\begin{aligned} - \frac{\theta^{k+1}_{i,j} - \theta^{k}_{i,j}}{\Delta t} = & -\frac{1}{\Delta z}\left[K_{i,j+1/2} - D_{i,j+1/2}\left(\frac{\theta^{k+1}_{i,j+1} - \theta^{k+1}_{i,j}}{\Delta z}\right)\right] + \frac{q^{k+1}_{i,j}}{\Delta z} \\ & - \frac{1}{\Delta x}\left[- D_{i+1/2,j}\left(\frac{\theta^{k+1}_{i+1,j} - \theta^{k+1}_{i,j}}{\Delta x}\right)\right] + \frac{q^{k+1}_{i,j}}{\Delta x} \end{aligned} \tag{3-10}$$

The implicit finite-difference expression (3-10) is solved to obtain the moisture content along the upper boundary which depends on precipitation less evapotranspiration and surface runoff. The flow at the top flat surface of a landfill is considered predominantly vertical. Similar finite-difference expressions are developed for top flat surface considering moisture movement only vertically from the upper to the lower node.

At the bottom of the unsaturated zone in a landfill, where the saturated leachate mound exists, the moisture gradient along the lower boundary is postulated herein as:

$$\frac{\partial \theta}{\partial x} = 0 \tag{3-11a}$$

$$\frac{\partial \theta}{\partial z} = 0 \tag{3-11b}$$

This type of postulation has been used successfully for one-dimensional moisture flow equations (Korfiatis et al., 1984), which in fact implies free drainage from the bottom due to gravity.

3.3 RUNOFF EQUATION

The overland flow equation or the kinematic wave equation (2-22) is transformed into a system of algebraic expressions using a finite-difference scheme. The finite-difference scheme is explained for two-dimensional grid in x-t plane as shown in Figure 3-4. Different finite-difference expressions yield different numerical schemes associated with their own stability and convergence criteria. Chow et al. (1988) described the finite-difference scheme which is stable and convergent. The finite-

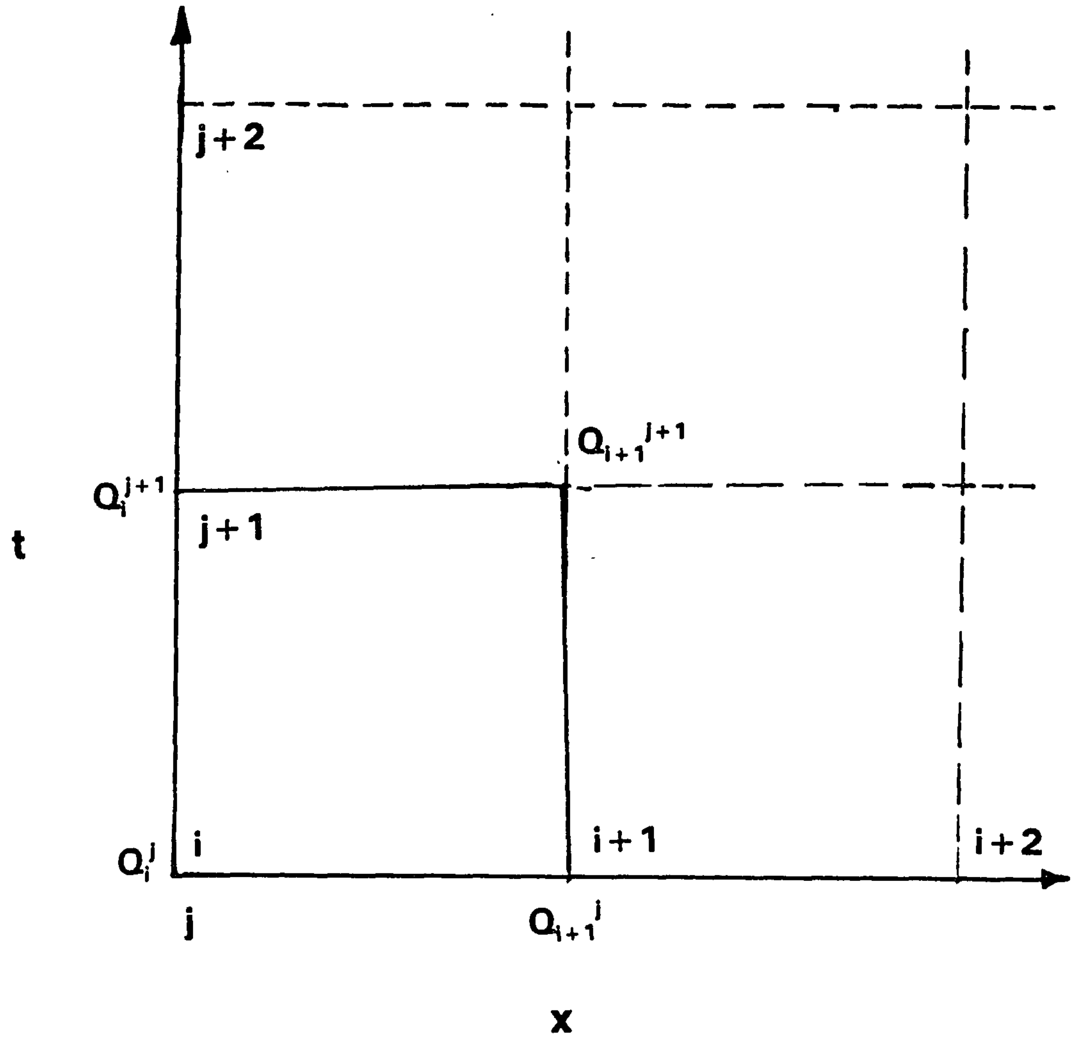


Figure 3-4. Finite-difference Scheme for the Kinematic Wave Equation.

difference expression with respect to Figure 3-4 is:

$$\frac{Q_{i+1}^{k+1} - Q_i^{k+1}}{\Delta x} + \alpha\beta\left(\frac{Q_{i+1}^{k} + Q_i^{k+1}}{2}\right)^{\beta-1}\left(\frac{Q_{i+1}^{k+1} - Q_{i+1}^{k}}{\Delta t}\right) = \bar{q} \qquad (3\text{-}12)$$

The unknown variable Q_{i+1}^{j+1} is then obtained as follows:

$$Q_{i+1}^{k+1} = \frac{\left[\frac{\Delta t}{\Delta x} Q_i^{k+1} + \alpha\beta Q_{i+1}^{k}\left(\frac{Q_{i+1}^{k} + Q_i^{k+1}}{2}\right)^{\beta-1} + \Delta t\bar{q}\right]}{\left[\frac{\Delta t}{\Delta x} + \alpha\beta\left(\frac{Q_{i+1}^{k} + Q_i^{k+1}}{2}\right)^{\beta-1}\right]} \qquad (3\text{-}13)$$

The solution procedure at each segment consists of solving for the flow rate Q at time t and then advancing the solution to forward time level t + Δt, considering the values of Q at the previous time level t as the initial condition.

3.4 SATURATED LEACHATE FLOW EQUATION

The finite-difference expression for the saturated leachate flow equation (2-44) can be obtained by discretizing it in the finite-difference grid as shown in Figure 3-5. The transformed finite-difference equation can be written as:

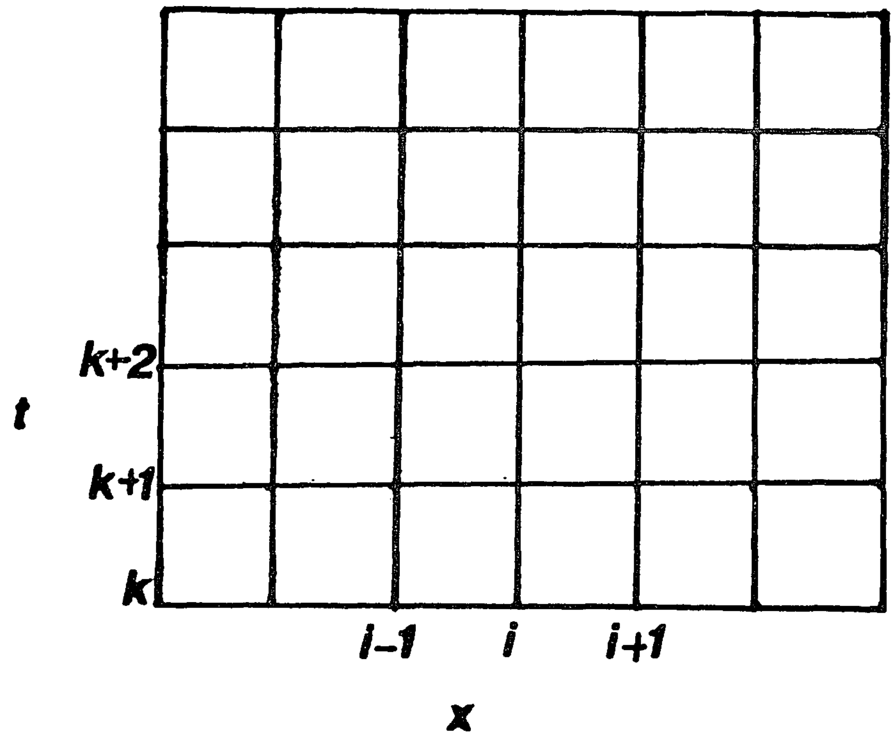


Figure 3-5. Finite-difference Scheme for the Saturated Leachate Flow Equation.

$$\frac{1}{(\Delta x)^2}[T_{i+1/2}(h_{i+1} - h_i) - T_{i-1/2}(h_i - h_{i-1})] - K_{si}\tan\gamma\left(\frac{h_{i+1} - h_{i-1}}{2\Delta x}\right)$$

$$+ N_{li} - K_{ci}\left(\frac{h_i + d_i}{d_i}\right) = n_{ei}\left(\frac{h_i^{k+1} - h_i^k}{\Delta t}\right) \qquad (3\text{-}14)$$

This system of finite-difference equations can be solved simultaneously for each grid-point by evaluating all the dependent variables at the forward time level, k+1.

Therefore, the implicit expression for the equation (3-14) can be written as:

$$\frac{1}{(\Delta x)^2}[T_{i+1/2}(h_{i+1}^{k+1} - h_i^{k+1}) - T_{i-1/2}(h_i^{k+1} - h_{i-1}^{k+1})] - K_{si}\tan\gamma\left(\frac{h_{i+1}^{k+1} - h_{i-1}^{k+1}}{2\Delta x}\right)$$

$$+ N_{li} - K_{ci}\left(\frac{h_i^{k+1} + d_i}{d_i}\right) = n_{ei}\left(\frac{h_i^{k+1} - h_i^k}{\Delta t}\right) \qquad (3\text{-}15)$$

The leachate accretion N_{li} is obtained after computing moisture values which helps build up the saturated leachate mound head. The values of leachate mound head h at each grid point are the only unknowns in equation (3-15) and are solved simultaneously at each time step using Gauss-Jordan elimination method.

3.5 NUMERICAL SOLUTION FOR UNSATURATED FLOW EQUATION

The system of finite-difference equations (3-2) derived from the partial differential equations describing moisture-flow in the unsaturated zone is solved by applying Gauss-Seidel iteration technique. The finite-difference expression (3-15) for the saturated leachate flow equation is solved using Gauss-Jordan elimination method. The iteration method is not used for the saturated leachate flow equation in the FILL model because for large values of leachate accretion from the unsaturated zone, the solution for the system of equations (3-15) does not converge rapidly. In the elimination method, the solution is obtained directly and no iteration is required.

In the unsaturated zone, iteration method is suitable to avoid storing large coefficient matrix for grid-points in the two-dimensional vertical domain. In the saturated zone, the coefficient matrix is not large and elimination method is used in the main model to avoid large central processing unit time required for iteration. However, for comparison with the analytical solution, both iteration and elimination methods are used for the solution of the finite-difference equations describing flow in both the unsaturated and saturated zones of a landfill. The two methods are used and compared with the analytical solution to establish the credibility of the model results.

To solve the system of algebraic equations for moisture content in the unsaturated zone, the relaxation scheme (Thomas, 1973) of the Gauss-Seidel iteration method is used. The values of the moisture content θ are assumed and the improved values are then calculated with the known initial and boundary conditions. A residual flow is caused by the discrepancy between the exact solution and the approximate values of the moisture content. This is gradually diminished by the process of relaxation and more accurate values of moisture content are obtained.

From equation (3-2) the residual value can be expressed as:

$$RES_{i,j} = -\frac{1}{\Delta t}\left(\theta_{i,j}^{k+1} - \theta_{i,j}^{k}\right) + \frac{D_{i+1/2,j}}{(\Delta x)^2}\left(\theta_{i+1,j}^{k+1} - \theta_{i,j}^{k+1}\right)$$
$$- \frac{D_{i-1/2,j}}{(\Delta x)^2}\left(\theta_{i,j}^{k+1} - \theta_{i-1,j}^{k+1}\right) + \frac{D_{i,j+1/2}}{(\Delta z)^2}\left(\theta_{i,j+1}^{k+1} - \theta_{i,j}^{k+1}\right)$$
$$- \frac{D_{i,j-1/2}}{(\Delta z)^2}\left(\theta_{i,j}^{k+1} - \theta_{i,j-1}^{k+1}\right) - \frac{1}{\Delta z}\left(K_{i,j+1/2} - K_{i,j-1/2}\right) \qquad (3\text{-}16)$$

Adding $\Delta\theta_{i,j}^{k+1}$ to $\theta_{i,j}^{K+1}$ in equation (3-16):

$$\frac{\theta_{i,j}^{k+1} + \Delta\theta_{i,j}^{k+1} - \theta_{i,j}^{k}}{\Delta t} = \frac{D_{i+1/2,j}}{(\Delta x)^2}\left(\theta_{i+1,j}^{k+1} - \theta_{i,j}^{k+1} - \Delta\theta_{i,j}^{k+1}\right)$$
$$- \frac{D_{i-1/2,j}}{(\Delta x)^2}\left(\theta_{i,j}^{k+1} + \Delta\theta_{i,j}^{k+1} - \theta_{i-1,j}^{k+1}\right)$$
$$+ \frac{D_{i,j+1/2}}{(\Delta z)^2}\left(\theta_{i,j+1}^{k+1} - \theta_{i,j}^{k+1} - \Delta\theta_{i,j}^{k+1}\right)$$
$$- \frac{D_{i,j-1/2}}{(\Delta z)^2}\left(\theta_{i,j}^{k+1} + \Delta\theta_{i,j}^{k+1} - \theta_{i,j-1}^{k+1}\right)$$
$$- \frac{1}{(\Delta z)}\left(K_{i,j+1/2} - K_{i,j-1/2}\right) \qquad (3\text{-}17)$$

From which:

$$\Delta\theta_{i,j}^{k+1} = \frac{RES_{i,j}}{\frac{1}{\Delta t} + \frac{1}{(\Delta x)^2}(D_{i+1/2,j} + D_{i-1/2,j}) + \frac{1}{(\Delta z)^2}(D_{i,j+1/2} + D_{i,j-1/2})} \qquad (3\text{-}18)$$

The coefficient of residual is termed relaxation coefficient. Therefore,

$$RELAX_{i,j} = \frac{1.0}{\frac{1}{\Delta t} + \frac{1}{(\Delta x)^2}(D_{i+1/2,j} + D_{i-1/2,j}) + \frac{1}{(\Delta z)^2}(D_{i,j+1/2} + D_{i,j-1/2})} \qquad (3\text{-}19)$$

Thus, the moisture content at grid-point (i,j) for the new time step t = k +1, is:

$$\theta_{i,j}(new) = \theta_{i,j}(old) + \Delta\theta_{i,j}(new) \qquad (3\text{-}20)$$

which is:

$$\theta_{i,j}(new) = \theta_{i,j}(old) + RELAX_{i,j}*RES_{i,j} \qquad (3\text{-}21)$$

Successive application of this iteration technique is continued until two consecutive sets of values agree within a prescribed tolerance criterion. A value of 10^{-5} is considered as the tolerance criterion for the convergence of the solution at a finite-difference grid-point.

3.6 NUMERICAL SOLUTION FOR SATURATED FLOW EQUATION

Finite-difference expressions (3-15) for the saturated leachate flow equation (2-44) are solved simultaneously using the Gauss-Jordan elimination method. Equation (3-15) can be rearranged as:

$$\tilde{b}h_i^{k+1} + \tilde{c}h_{i-1}^{k+1} + \tilde{d}h_{i+1}^{k+1} = \tilde{f} \qquad (3\text{-}22)$$

where,

$$\tilde{b} = -\frac{1}{(\Delta x)^2}[T_{i+1/2} + T_{i-1/2}] - \frac{K_c}{d} - \frac{K_c}{\Delta t}$$

$$\tilde{c} = \frac{T_{i-1/2}}{(\Delta x)^2} - \frac{K_s \tan\gamma}{2\Delta x}$$

$$\tilde{d} = \frac{T_{i+1/2}}{(\Delta x)^2} - \frac{K_s \tan\gamma}{2\Delta x}$$

$$\tilde{f} = -N_1 + K_c - n_e\frac{h_i^k}{\Delta t}$$

For n number of nodes, n number of equations of the type (3-22) are obtained. They may be expressed in matrix form as:

$$[A]\,[h] = [F] \qquad (3\text{-}23)$$

where,

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & . & . & . & . & a_{1n} \\ a_{21} & a_{22} & a_{23} & . & . & . & . & a_{2n} \\ a_{31} & a_{32} & a_{33} & . & . & . & . & a_{3n} \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ a_{n1} & a_{n2} & a_{n3} & . & . & . & . & a_{nn} \end{bmatrix}$$

$$[h] = \begin{bmatrix} h_1^{k+1} \\ h_2^{k+1} \\ h_3^{k+1} \\ . \\ . \\ . \\ h_n^{k+1} \end{bmatrix} \qquad [F] = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ . \\ . \\ . \\ f_n \end{bmatrix}$$

In using Gauss-Jordan elimination method, the coefficient matrix in equation (3-23) is reduced to a diagonal matrix. In this process, the ith row is used to eliminate the coefficients above and below it in the ith column. Thus at the end of the solution, the diagonal elements of [A] become unity with zero values in the remaining elements. The elimination process to make zero in the ith column of [A] with the corresponding operations on the right-hand side column vector [F] can be given as follows:

$$a_{i,j}^{(i+1)} = a_{i,j}^{(i)}/a_{i,i}^{(i)}\ , \ j=i,n$$

$$a_{k,m}^{(i+1)} = a_{k,m}^{(i)} - a_{k,i}^{(i)} * a_{i,m}^{(i)}/a_{i,i}^{(i)}$$

$$k=1,n;\ m=i,n;\ k \neq i.$$

$$f_i^{(i+1)} = f_i^{(i)}/a_{i,i}^{(i)}$$

$$f_k^{(i+1)} = f_k^{(i)} - a_{k,i} * f_i^{(i)}/a_{i,i}^{(i)},\ k=1,n \text{ and } k \neq i.$$

Rearrangement of rows and columns, called pivoting is performed when a diagonal element is nearly zero to ensure numerical stability (Dahlquist and Bjorck, 1974). The set of equations (3-22) are checked whether they retain large values of the diagonal elements at each step of the elimination.

CHAPTER 4

BOUNDARY INTEGRAL FORMULATION

The boundary integral formulations presented in this chapter are derived using the time-dependent fundamental solution of the governing partial differential equation. Two boundary integral approaches are presented. The first approach solves the unsaturated moisture-flow problem by directly applying the time-dependent Green's function. The second method is based on the expansion of the moisture content into perturbation series. The governing equation is decomposed into a moisture-flow equation without the gravity term on the right-hand side and a sequence of moisture flow equation with known gravity term on the right-hand side. The perturbation equations are then solved in succession by evaluating the gravity term of the unsaturated leachate flow equation from the preceding level of solution. The two boundary integral approaches, direct Green's function and perturbation Green's function, are closely related because the same time-dependent fundamental solution is used to formulate the boundary integral expressions. The direct Green's function formulation solves the boundary integral expression in one step. However, the perturbation Green's function approach requires repeated solution of the integral expression for homogeneous boundary conditions.

4.1 DIRECT GREEN'S FUNCTION (DGF) FORMULATION

The present boundary integral procedure for the time-dependent moisture-flow problem is similar to the technique employed by Taigbenu (1985) that was used to solve the unsteady aquifer flow problems. The method uses the time-dependent Green's function which satisfies the diffusion equation. Liggett and Liu (1979) used this method to predict the hydraulic head in a confined aquifer. To apply this method to unsaturated moisture-flow equation (2-11), it is necessary to linearize it to the form of the diffusion equation. Rubin and Steinhardt (1963) suggested that the linearization of the moisture flow equation can be done by computing the unsaturated moisture-flow characteristics by using the moisture content from the previous time steps. In this case, the linearization is also done by considering the diffusivity and the gravity term to be dependent on the moisture content computed from the previous time step.

The linearized form of the unsteady moisture-flow equation (2-11) can be written as:

$$\frac{\partial \theta}{\partial t} = \hat{D} \nabla^2 \theta - \frac{\partial K(\theta^k)}{\partial z} \tag{4-1}$$

where,

$\hat{D} = D(\theta^k)$ and

θ^k = moisture content θ at previous time step t=k.

The free-space Green's function for the present problem is used that satisfies the following equation:

$$\frac{\partial G}{\partial t} = \hat{D}\nabla^2 G + \delta(r - r_i, t - \tau) \qquad (4\text{-}2)$$

where,

$r\text{-}r_i$ = distance between any field point (x,z) and the source point (x_i,z_i), and

τ = a time variable.

Morse and Feshbach (1953) gives the solution for Green's function as:

$$G(r,t) = \frac{1}{4\pi\hat{D}(t-\tau)}\exp\left[-\frac{r^2}{4\hat{D}(t-\tau)}\right] \qquad (4\text{-}3)$$

Following the direct boundary integral formulation, which employs the Green's identity, the expression (4-1) can be written in the integral form as (Liggett and Liu, 1979):

$$\lambda\theta = \hat{D}\int_{\Gamma}\int_{t_o}^{t_1}\left[G\frac{\partial\theta}{\partial n} - \theta\frac{\partial G}{\partial n}\right]d\tau\, d\Gamma + \int_{\Omega}\theta_o G\, d\Omega - \int_{\Omega}\int_{t_o}^{t^1} G\frac{\partial K(\theta^k)}{\partial z}d\tau\, d\Omega$$

(4-4)

where,

Ω is the domain and Γ is the boundary,

θ_o = initial moisture content,

$\lambda = 1$ if $(x,z) \varepsilon \Omega$,

$\lambda = 0.5$ if $(x,z) \varepsilon \Gamma$ and Γ is smooth at (x,z), and

n = outward unit normal at the boundary.

Some of the data for θ and $\partial\theta/\partial n$ are unknown on the boundary which are obtained by solving boundary integral equation (4-4). After solution of the boundary integral equation (4-4), the values of θ and $\partial\theta/\partial n$ are known everywhere on the boundary. Once these values at the boundary are known, the solution for θ in the domain can be obtained by writing the discretized boundary integral expression for an interior node i as:

$$\theta_i = \sum_{j=1}^{N} \left\{ \left(\frac{\partial\theta}{\partial n}\right)_j \int_{\Gamma_j}\int_{t_o}^{t_1} G \, d\tau \, d\Gamma_j - \theta_j \int_{\Gamma_j}\int_{t_o}^{t_1} \frac{\partial G}{\partial n} d\tau \, d\Gamma_j \right\} + \int_{\Omega} \theta_o G \, d\Omega - \int_{\Omega}\int_{t_o}^{t_1} G \frac{\partial K(\theta^k)}{\partial z} d\tau \, d\Omega \tag{4-5}$$

where,

N = total number of boundary elements.

4.2 TIME INTEGRATION

In time integration, it is assumed that the function θ and $\partial\theta/\partial n$ remain constant on time. The time integration is performed stepwise in which the function G and $\partial G/\partial n$

vary with respect to time. The expression for ∂G/∂n is obtained by differentiating equation (4-3) as follows:

$$\frac{\partial G}{\partial n} = -\frac{1}{8\pi\hat{D}^2(t_1-\tau)^2}\exp\left[-\frac{r^2}{4\hat{D}(t_1-\tau)}\right] r\,\frac{\partial r}{\partial n}$$

According to the local coordinate system as shown in Figure 4-1:

$$\frac{\partial r}{\partial n} = \frac{\partial r}{\partial \eta}$$

$$r = \sqrt{\eta^2 + \xi^2}$$

$$\frac{\partial r}{\partial \eta} = \frac{\eta}{r}$$

Therefore,

$$\frac{\partial r}{\partial n} = \frac{\eta}{r}$$

and

$$\frac{\partial G}{\partial n} = -\frac{\eta}{8\pi\hat{D}^2(t_1-\tau)^2}\exp\left[-\frac{r^2}{4\hat{D}(t_1-\tau)}\right] \qquad (4\text{-}6)$$

Now, the integrals are evaluated in time as follows (Brebbia et al., 1984):

$$\int_{t_o}^{t_1} G\,d\tau = \frac{1}{4\pi\hat{D}}E(a) \qquad (4\text{-}7)$$

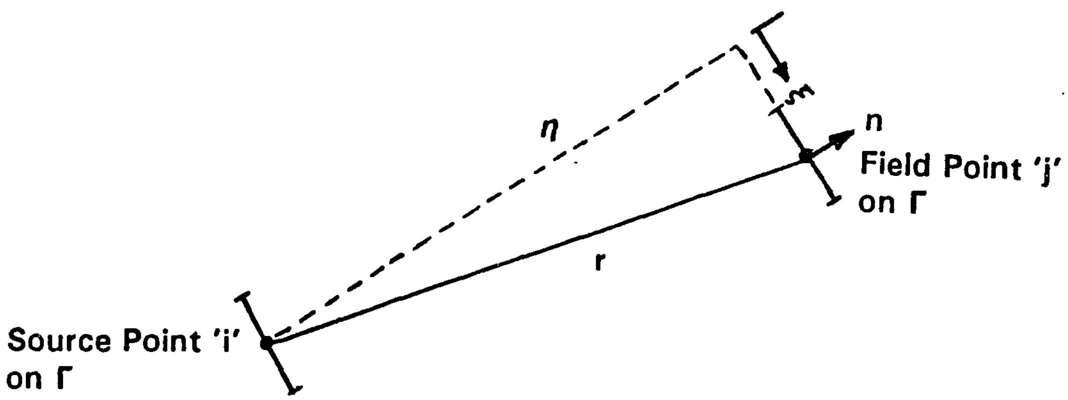


Figure 4-1. Geometrical Definition of a Local Coordinate System.

$$\int_{t_o}^{t_1} \frac{\partial G}{\partial n} = -\frac{1}{2\pi \hat{D} r^2} e^{-a} \tag{4-8}$$

where,

$$a = \frac{r^2}{4\hat{D}\Delta t}$$

E(a) = exponential-integral function.

The exponential-integral function can be expressed in the form of a series as (Brebbia et al., 1984):

$$E(z) = -C - \ln z + \sum_{n=1}^{\infty} (-1)^{n-1} \frac{z^{n-1}}{n.n!} \tag{4-9}$$

where,

C = 0.57721566

For convenience in numerical computation, the exponential-integral function is evaluated by polynomial and rational expansions as (Abramowitz and Stegan, 1970):

$$E(z) = -\ln z + a_0 + a_1 z + a_2 z^2 + a_3 z^3 + a_4 z^4 + a_5 z^5 + \varepsilon(z) \tag{4-10}$$

$$0 \le 1; \; |\varepsilon(z)| < 2*10^{-7}$$

After integration in time, the boundary integral equation (4-4) becomes:

$$E(z) = \frac{1}{ze^{z}}\left[\frac{z^4+b_1z^3+b_2z^2+b_3z+b_4}{z^4+c_1z^3+c_2z^2+c_3z+c_4}\right] \tag{4-11}$$

$$1 \le z \le \infty; \quad |\varepsilon(z)| \le 2*10^{-8}$$

$$\lambda\theta = \int_\Gamma \frac{\partial\theta}{\partial n} E(a)\frac{1}{4\pi}d\Gamma - \int_\Gamma \theta\left\{-\frac{\eta e^{-a}}{2\pi r^2}\right\}d\Gamma + \int_\Omega \theta_o G d\Omega$$

$$- \int_\Omega \frac{\partial K(\theta^k)}{\partial z}\frac{E(a)}{4\pi\hat{D}}d\Omega \tag{4-12}$$

Equation (4-12) now contains only the boundary and domain integrals which are evaluated to solve for the unknowns θ or ∂θ/∂n at the boundary. The values of the boundary and area integrals in equation (4-12) can be calculated using numerical quadrature rules. The singularities associated with the boundary and area integrals are evaluated using special quadrature formulae. After evaluating the integrals, the coefficients of the Dirichlet and Neumann boundary conditions in equation (4-12) are obtained. The whole set of equations for N number of nodes at the boundary can be written in matrix form as:

$$[\tilde{A}]\{\theta_i\} = [\tilde{B}]\left\{\left(\frac{\partial\theta}{\partial n}\right)_i\right\} + [\tilde{H}] \tag{4-13}$$

$$i = 1, 2, 3, \ldots, N$$

where,

$\tilde{A}$ and $\tilde{B}$ are the coefficient matrices, and

$\tilde{H}$ is a known column vector that corresponds to the sum of the values of the domain integrals in equation (4-12).

The system of equations (4-13) reveals that N_1 values of θ and N_2 values of $\partial\theta/\partial n$ are known and therefore, there are a set of N ($N = N_1 + N_2$) unknowns in the expression (4-13). Rearranging the equation (4-13) in such a way that all the unknowns are on the right hand side, one can write the expression in matrix form as:

$$[\tilde{C}]\,[\tilde{X}] = [\tilde{F}] \tag{4-14}$$

Here, on the left hand side $\tilde{C}$ is the known coefficient matrix, $\tilde{X}$ is the unknown (for θ or $\partial\theta/\partial n$) column vector and on the right hand side $\tilde{F}$ is the known column vector correspond to the sum of the values of the domain integrals and the value of the integrals with known boundary conditions. The system of equations (4-14) can be solved for the unknown boundary conditions using Gauss-Jordan elimination method.

4.3 BOUNDARY INTEGRATION

The boundary integrations associated with equation (4-12) are carried out by considering constant elements with mid-node as shown in Figure 4-2. In a typical element, the values of θ and $\partial\theta/\partial n$ are assumed constant and equal to the value at mid-node of the element. The boundary is divided into N elements of which N_1 belongs to Γ_1 (Dirichlet boundary) and N_2 belongs to Γ_2 (Neumann boundary). For a node 'i' at the boundary, equation (4-12) can be written as:

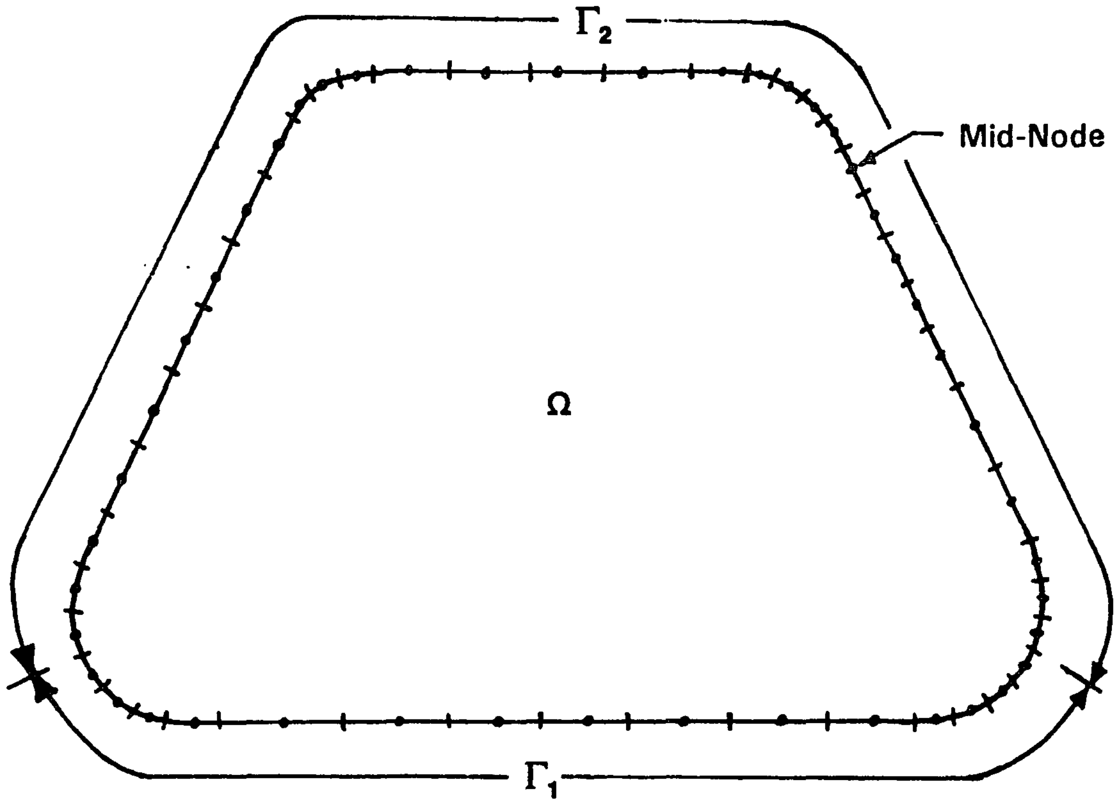


Figure 4-2. Constant Boundary Elements.

$$\lambda\theta_i = \sum_{j=1}^{N}\left(\frac{\partial\theta}{\partial n}\right)_j A_{ij} - \sum_{j=1}^{N}\theta_j B_{ij} + C_i - D_i \tag{4-15}$$

where,

$$A_{ij} = \int_\Gamma \frac{E(a)}{4\pi}\,d\Gamma \tag{4-16}$$

$$B_{ij} = \int_\Gamma \left\{-\frac{\eta}{2\pi r^2}\right\} e^{-a}\,d\Gamma \tag{4-17}$$

$$C_i = \int_\Omega \theta_o G\,d\Omega \tag{4-18}$$

$$D_i = \int_\Omega \frac{\partial K(\theta^k)}{\partial z}\,\frac{E(a)}{4\pi\hat{D}}\,d\Omega \tag{4-19}$$

The values of the integrals A_{ij} and B_{ij} are calculated using a four-point Gauss quadrature rule. In Gauss quadrature rule, the integration is performed as follows (Carnahan et al.,1972):

$$\int_{-1}^{1} f(\xi)\,d\xi = \sum_{m=1}^{n} \omega_m f(\xi_m) \tag{4-20}$$

where ξ_m are the base-point values and ω_m are the weight factors corresponding to the base-points. The values of the base-points and the weight factors are obtained from Brebbia (1978). The integrals A_{ij} and B_{ij} are of the type as follows:

$$A_{ij} = \sum_{m=1}^{4} \frac{E(a_{im})}{4\pi} \omega_m \frac{\Delta s}{2} \tag{4-21}$$

$$B_{ij} = \sum_{m=1}^{4} -\frac{\eta}{2\pi r_{im}^2} \exp(-a_{im}) \frac{\Delta s}{2} \tag{4-22}$$

where,

$$a_{im} = \frac{r_{im}^2}{4\hat{D}\Delta t}$$

r_{im} = distance between the source point and the numerical integration points.

Δs = length of field element,

ω_m = weight factor for the mth integration point.

The integral A_{ij} has a logarithmic singularity in the exponential-integral function as $j \to i$. The integral A_{ij} for singularity is evaluated as (Brebbia et al., 1982):

$$A_{ij} = \frac{\Delta s}{4\pi}\left[-C - \ln\alpha + 2 + \sum_{m=1}^{\infty} (-1)^{m-1} \frac{\alpha^m}{m(2m+1)m!}\right] \quad \text{for } i=j \tag{4-23}$$

where,

$$\alpha = \frac{(\Delta s)^2}{16\hat{D}\Delta t}$$

4.4 AREA INTEGRATION

The integrals defined by equations (4-18) and (4-19) are evaluated by dividing the domain into a series of triangular cells as shown in Figure 4-3. On each cell the integration can be performed as (Brebbia, 1978):

$$C_i = \sum_{m'=1}^{n_e} \left\{ \sum_{m=1}^{n} \omega_m (\theta_o G)_m \right\} A_{m'} \tag{4-24}$$

$$D_i = \sum_{m'=1}^{n_e} \left\{ \sum_{m=1}^{4} \omega_m \left[\frac{\partial K(\theta^k)}{\partial z} \frac{E(a)}{4\pi\hat{D}} \right]_m \right\} A_{m'} \tag{4-25}$$

where,

n_e = number of elements,

n = number of integration points on each element,

ω_m = weighting function, and

$A_{m'}$ = area of a cell or an element.

In this case, the integration points are considered as the vertices of triangular elements. The weighting functions for the integration points are used from Tables given by Zienkiewicz and Taylor (1989).

The singularity arises for the domain integral D_i when the source point coincides with one of the vertices of the triangular element. The logarithmic singularity in D_i due to the exponential-integral function is evaluated using the special quadrature

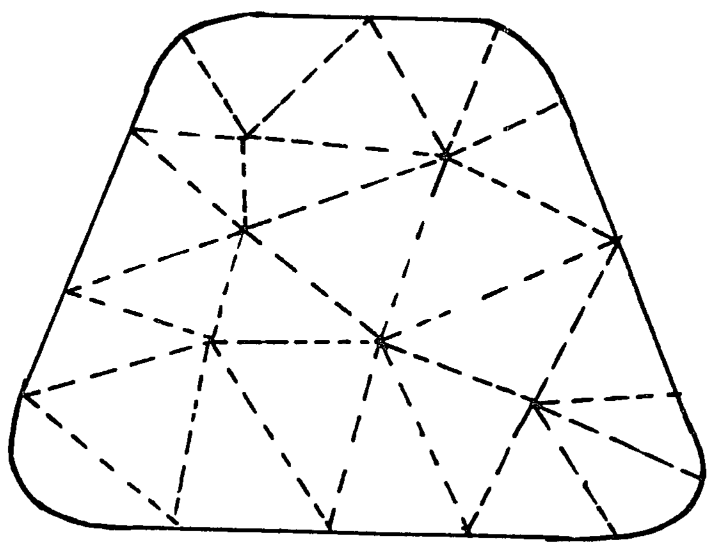


Figure 4-3. Triangular Elements for Numerical Integration in the Domain.

rule suggested by Liggett (1982). A general situation for singularity is shown at the origin (point 1) in Figure 4-4. The special quadrature formula for the integral due to logarithmic singularity (I_s) associated with the exponential-integral function in D_i is expressed as (Liggett, 1982):

$$I_s = -\tilde{D}\eta_s^2\left[(\tilde{u}-\tilde{v})\left(\ln\eta_s - \frac{3}{2}\right) + \frac{\tilde{u}}{2}\ln(1+\tilde{u}^2) - \frac{\tilde{v}}{2}\ln(1+\tilde{v}^2) + \tan^{-1}\tilde{u} - \tan^{-1}\tilde{v}\right] \quad (4\text{-}26)$$

where,

$$\tilde{D} = \frac{1}{4\pi\hat{D}}\frac{\partial K(\theta^k)}{\partial z}$$

u = slope of line 1-3 as shown in Figure 4-4,

v = slope of line 1-2 as shown in Figure 4-4, and

η_s = perpendicular distance from point 1 to line 2-3 as shown in Figure 4-4.

In the series expansion (4-9) for exponential-integral function, the log singularity associated with the integral D_i is separated to evaluate using the above expression. The rest of the series in integral D_i is evaluated using the four-point Gauss quadrature rule.

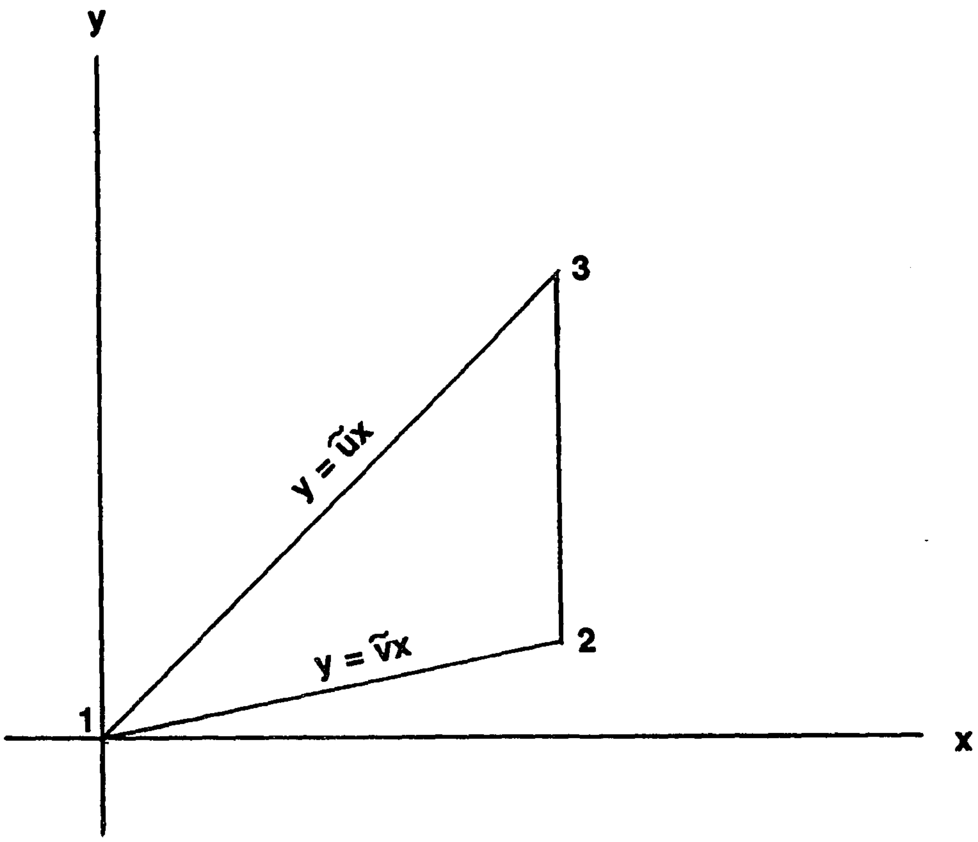


Figure 4-4. Definition Sketch of a Triangle for Singular Integral Computation (Liggett, 1982).

4.5 INTERNAL FLUXES

The velocities of the moisture flow in the unsaturated zone is of particular interest to predict leachate flow in a landfill. The moisture flux is used to compute leachate accretion onto the saturated leachate mound. The recharge to the saturated leachate mound depends on the moisture-flux in the unsaturated zone of a landfill. In the boundary integral formulation, the point value of the moisture-flux is calculated by differentiating the boundary integral formulation of the dependent variable. The moisture flow velocity in the x-direction is obtained by differentiating equation (4-5) as:

$$\frac{\partial\theta}{\partial x} = \hat{D}\int_{\Gamma}\int_{t_o}^{t_1}\frac{\partial G}{\partial x}\frac{\partial\theta}{\partial n}\,d\tau\,d\Gamma - \hat{D}\int_{\Gamma}\int_{t_o}^{t_1}\theta\frac{\partial^2 G}{\partial x\,\partial n}\,d\tau\,d\Gamma + \int_{\Omega}\theta_o\frac{\partial G}{\partial x}\,d\Omega$$

$$- \int_{\Omega}\int_{t_o}^{t^1}\frac{\partial G}{\partial x}\frac{\partial K(\theta^k)}{\partial z}\,d\tau\,d\Omega \qquad (4\text{-}27)$$

Similarly in the z-direction, the moisture-flux is expressed as:

$$\frac{\partial\theta}{\partial z} = \hat{D}\int_{\Gamma}\int_{t_o}^{t_1}\frac{\partial G}{\partial z}\frac{\partial\theta}{\partial n}\,d\tau\,d\Gamma - \hat{D}\int_{\Gamma}\int_{t_o}^{t_1}\theta\frac{\partial^2 G}{\partial z\,\partial n}\,d\tau\,d\Gamma + \int_{\Omega}\theta_o\frac{\partial G}{\partial z}\,d\Omega$$

$$- \int_{\Omega}\int_{t_o}^{t^1}\frac{\partial G}{\partial z}\frac{\partial K(\theta^k)}{\partial z}\,d\tau\,d\Omega \qquad (4\text{-}28)$$

After time integration, the expressions (4-27) and (4-28) contains the boundary and area integrals and are expressed as:

$$\frac{\partial\theta}{\partial x}=\int_{\Gamma}\left\{-\frac{e^{-a}}{2\pi r}\frac{\partial r}{\partial x}\right\}\frac{\partial\theta}{\partial n}d\Gamma-\int_{\Gamma}\left\{\frac{\eta(1+a)e^{-a}}{\pi r^{3}}\frac{\partial r}{\partial x}-\frac{e^{-a}}{2\pi r^{2}}\frac{\partial\eta}{\partial x}\right\}\theta d\Gamma$$

$$+\int_{\Omega}\theta_{o}\left\{\frac{-re^{-a}}{8\pi\hat{D}^{2}(\Delta t)^{2}}\frac{\partial r}{\partial x}\right\}d\Omega-\int_{\Omega}\frac{\partial K(\theta^{k})}{\partial z}\left\{\frac{e^{-a}}{2\pi r\hat{D}}\frac{\partial r}{\partial x}\right\}d\Omega \quad (4-29)$$

$$\frac{\partial\theta}{\partial z}=\int_{\Gamma}\left\{-\frac{e^{-a}}{2\pi r}\frac{\partial r}{\partial z}\right\}\frac{\partial\theta}{\partial n}d\Gamma-\int_{\Gamma}\left\{\frac{\eta(1+a)e^{-a}}{\pi r^{3}}\frac{\partial r}{\partial z}-\frac{e^{-a}}{2\pi r^{2}}\frac{\partial\eta}{\partial z}\right\}\theta d\Gamma$$

$$+\int_{\Omega}\theta_{o}\left\{\frac{-re^{-a}}{8\pi\hat{D}^{2}(\Delta t)^{2}}\frac{\partial r}{\partial z}\right\}d\Omega-\int_{\Omega}\frac{\partial K(\theta^{k})}{\partial z}\left\{\frac{e^{-a}}{2\pi r\hat{D}}\frac{\partial r}{\partial z}\right\}d\Omega \quad (4-30)$$

The boundary and area integrations are done as explained before. Once the unknown boundary values θ and $\partial\theta/\partial n$ are determined, they are substituted into equations (4-29) and (4-30) to obtain internal fluxes.

4.6 PERTURBATION-GREEN'S FUNCTION (PGF) FORMULATION

A perturbation technique similar to Lafe and Cheng (1987) is developed to solve equation (4-1) by evaluating the gravity term $\partial K(\theta)/\partial z$ at the current time level. It is assumed that the volumetric moisture content θ in the x-z-t space can be expressed as:

$$\theta=\theta_{o}+\theta_{1}+\theta_{2}+\theta_{3}+\ldots\ldots\ldots \quad (4-31)$$

where,

$\theta_o >> \theta_1 >> \theta_2 >> \ldots\ldots.$

It is required that the leading term in the series satisfies the moisture-flow equation with no source term. A sequence of moisture-flow equations with known right-hand sides are written. Therefore, the following set of equations can be written as:

$$\frac{\partial \theta_o}{\partial t} - \hat{D}\nabla^2\theta_o = 0 \tag{4-32}$$

$$\frac{\partial \theta_i}{\partial t} - \hat{D}\nabla^2\theta_i = -\frac{\partial K(\theta_{i-1})}{\partial z} \tag{4-33}$$

$$i = 1, 2, 3, 4, 5, \ldots\ldots$$

Once equation (4-32) is solved for θ_o, then the right-hand side of equation (4-33) indicating the gravity term $\partial K(\theta_o)/\partial z$ is known. Finally, equation (4-33) is solved for θ_1, θ_2, θ_3, ..., using the value of θ from the preceding level of solution.

In the perturbation quantities, it is required that the leading term absorb the entire boundary conditions:

$$\theta_o = \bar{\theta} \qquad \text{on } \Gamma_1$$

$$\frac{\partial \theta_o}{\partial n} = i_n \qquad \text{on } \Gamma_2$$

$$\theta_i = 0 \qquad \text{on } \Gamma_1$$

$$\frac{\partial \theta_i}{\partial n} = 0 \qquad \text{on } \Gamma_2$$

$$i = 1, 2, 3, 4, \ldots\ldots$$

Therefore, a new system of equations is obtained from the original boundary value problem. The new system consists of solving the homogeneous equation with the original boundary and initial conditions. A sequence of nonhomogeneous equations is solved with homogeneous boundary conditions and known right-hand side obtained from the preceding level of solution. Successive solutions are required to find the final result.

The boundary integral formulation is the same as for the DGF formulation explained before. In the boundary integral formulation for the homogeneous equation (4-32), the domain integral D_i defined by equation (4-19) does not exist. The boundary and domain integrations for the nonhomogeneous equations (4-33) are the same as described in direct Green's function formulation. In the domain integral D_i for the nonhomogeneous equation (4-33), the gravity term is evaluated as $\partial K(\theta_{i-1})/\partial z$, where θ_{i-1} is the value of θ at the preceding level of solution.

CHAPTER 5

COMPARISON OF NUMERICAL MODELS WITH ANALYTICAL SOLUTION

This chapter evaluates the performance of the finite-difference and boundary integral solution techniques to solve the unsaturated leachate flow equation. The finite-difference numerical solution technique is compared with the analytical solution for the saturated leachate flow equation. Comparisons of the numerical solution techniques are made with the analytical solution of the governing partial differential equations in simplified forms for idealized flow conditions with regular boundaries of the domain. Although the comparisons of the numerical models are made for idealized flow conditions, confidence is gained for the applicability of the models for complex real field conditions if the models reproduce agreeable true solution.

Analytical solution is derived for the unsaturated leachate flow equation for a rectangular domain defined by both Dirichlet and Neumann boundary conditions. The finite-difference solution of the moisture-flow equation using Gauss-Seidel iteration and Gauss-Jordan elimination methods are compared with the exact solution. The direct Green's function and perturbation Green's function boundary integral solutions are also compared with the finite-difference and exact solutions.

The analytical solution for the saturated leachate flow equation is obtained for a drainage layer with known leachate accretion onto it. The two numerical solution techniques, Gauss-Seidel iteration and Gauss-Jordan elimination are used to solve the simplified form of the saturated leachate flow equation. The numerical solution results are compared with the exact solutions.

5.1 ANALYTICAL SOLUTION

Analytical solutions are obtained for both the governing partial differential equations in the unsaturated and saturated zones of a landfill. The analytical solution for the unsaturated flow is done on a two-dimensional rectangular domain considering unsteady Dirichlet boundary condition. The steady Neumann boundary condition is assumed for both the unsaturated and saturated leachate flow equation.

5.1.1 Unsaturated Flow Equation

Analytical solution is difficult to obtain for the complex moisture-flow equation including all the flow components active in a landfill. However, analytical solutions can be obtained for the unsteady moisture-flow equation with the help of some simplifying assumptions. The assumptions made in the derivation of the two-dimensional moisture flow equation are :

- A well-defined rectangular domain is considered with Dirichlet and Neumann boundary conditions;

- The diffusivity is a function of moisture content. It is estimated using the moisture content from the previous time step; and

- The flow due to gravity is obtained from the moisture content at the previous time-step. This condition is obtained by defining the gravity term as $\partial K(\theta^k)/\partial z$, where θ^k is the moisture content at the previous time step t=k.

With the above simplifying assumptions, the unsaturated moisture-flow equation (2-11) becomes:

$$\frac{\partial \theta}{\partial t} = \hat{D}\left(\frac{\partial^2 \theta}{\partial x^2} + \frac{\partial^2 \theta}{\partial z^2}\right) - p \qquad (5\text{-}1)$$

where,

$$p = \frac{\partial K(\theta^k)}{\partial z}$$

Analytical solution for the above equation is obtained using the method of separation of variables with respect to Figure 5-1a for the following boundary conditions:

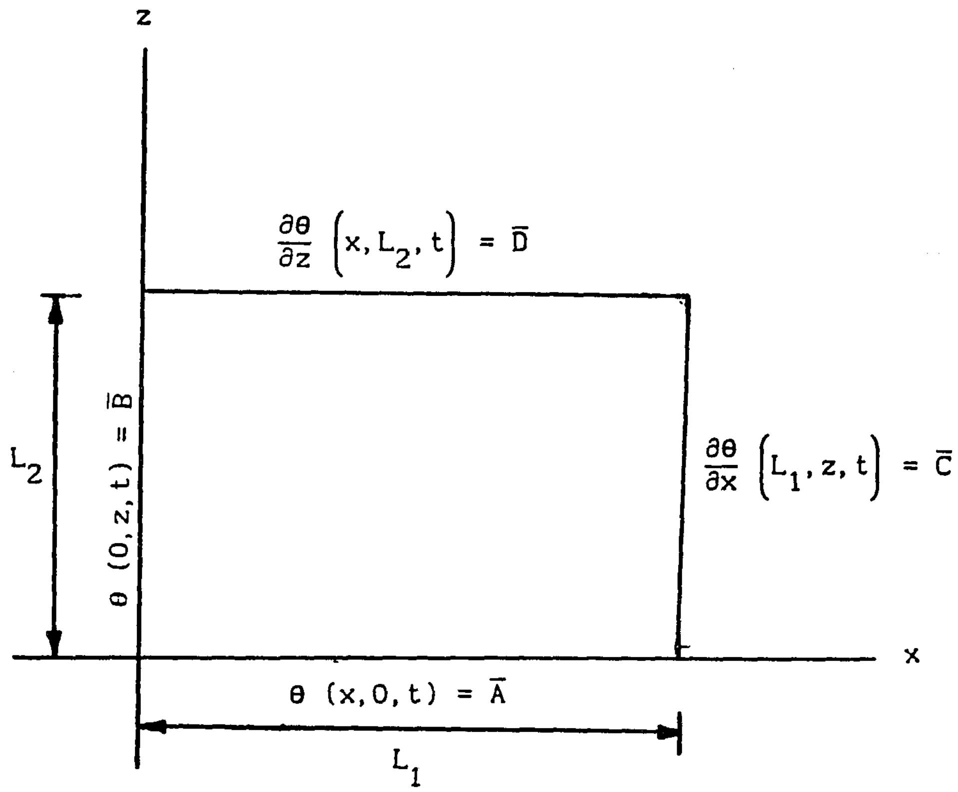


Figure 5-1a. Computational Domain for Analytical Solution.

$$\theta(x,0,t) = \bar{A}$$

$$\theta(0,z,t) = \bar{B}$$

$$\frac{\partial\theta}{\partial x}(L_1,z,t) = \bar{C}$$

$$\frac{\partial\theta}{\partial z}(x,L_2,t) = \bar{D}$$

The initial condition is as follows:

$$\theta(x,z,0) = \theta_o$$

The solution for the moisture content θ, obtained for the above boundary and initial conditions is expressed as:

$$\theta = \sum_{n=0}^{\infty}\left[\sum_{m=0}^{\infty} C_{mn}(t)\,\mathrm{Sin}\frac{(2m+1)\pi x}{2L_1} + \beta_n + \frac{x^2}{2L_1}\hat{C}\right]\mathrm{Sin}\frac{(2n+1)\pi z}{2L_2} + \bar{A} + \frac{z^2}{2L_2}\bar{D} \tag{5-2}$$

where,

$$C_{mn}(t) = \frac{t_{mn}}{\hat{\alpha}} + \hat{\gamma}e^{-\hat{\alpha}t}$$

$$t_{mn} = \frac{4}{(2m+1)\pi}[t_{mn1} - t_{mn2}]$$

$$t_{mn1} = \overline{t}_n + \alpha_c - \alpha_\beta$$

$$t_{mn2} = \frac{\hat{a}}{L_1}\left\{\frac{2L_1^2}{(2m+1)\pi}\mathrm{Sin}\frac{(2m+1)\pi}{2} - \left[\frac{2L_1}{(2m+1)\pi}\right]^2\right\}$$

$$\overline{t}_n = \frac{4}{(2n+1)\pi}[\alpha_D - p]$$

$$\alpha_D = \frac{\hat{D}\overline{D}}{L_2}$$

$$\alpha_c = \frac{\hat{D}\hat{C}}{L_1}$$

$$\hat{C} = \frac{4\overline{C}}{(2n+1)\pi}$$

$$\alpha_\beta = \hat{D}\left\{\frac{(2n+1)\pi}{2L_2}\right\}^2\beta_n$$

$$\beta_n = \frac{4}{(2n+1)\pi}\{\beta_{n1} - \beta_{n2}\}$$

$$\beta_{n1} = \overline{B} - \overline{A}$$

$$\beta_{n2} = \frac{\overline{D}}{L_2}\left\{\frac{2L_2^2}{(2n+1)\pi}\mathrm{Sin}\frac{(2n+1)\pi}{2} - \left[\frac{2L_2}{(2n+1)\pi}\right]^2\right\}$$

$$\hat{a} = \hat{D}\hat{C}\left\{\frac{(2n+1)\pi}{2L_2}\right\}^2$$

The detail derivation of the above analytical solution is given in Appendix A.

The analytical solution for the unsaturated moisture-flow equation (5-1) can be obtained in one-dimensional form by disregarding the gravity term $\partial K(\theta^k)/\partial z$. The initial and boundary conditions are shown in Figure 5-1b and can be expressed as follows:

$$\theta = \theta_{in} \qquad z \leq 0,\ t = 0$$

$$\theta = \theta_{oo} \qquad z = 0,\ t > 0$$

$$\theta \text{ finite for } z \rightarrow -\infty,\ t > 0$$

The solution for the one-dimensional moisture-flow problem with the given initial and boundary conditions is expressed as (Eagleson, 1970):

$$\theta(z,t) = \theta_{oo} + (\theta_{in} - \theta_{oo})\,\mathrm{erf}\left[\frac{|z|}{2(\hat{D}t)^{1/2}}\right] \qquad (5\text{-}3)$$

The unsaturated moisture content can be obtained using the above one-dimensional analytical solution. The boundary integral solution for moisture content in a soil column can be compared with the exact solution expressed in equation (5-3).

5.1.2 Saturated Flow Equation

Analytical solution for the saturated leachate flow equation is derived by considering the following assumptions:

- The top of the underlying bottom clay (Figure 2-4) is considered horizontal;

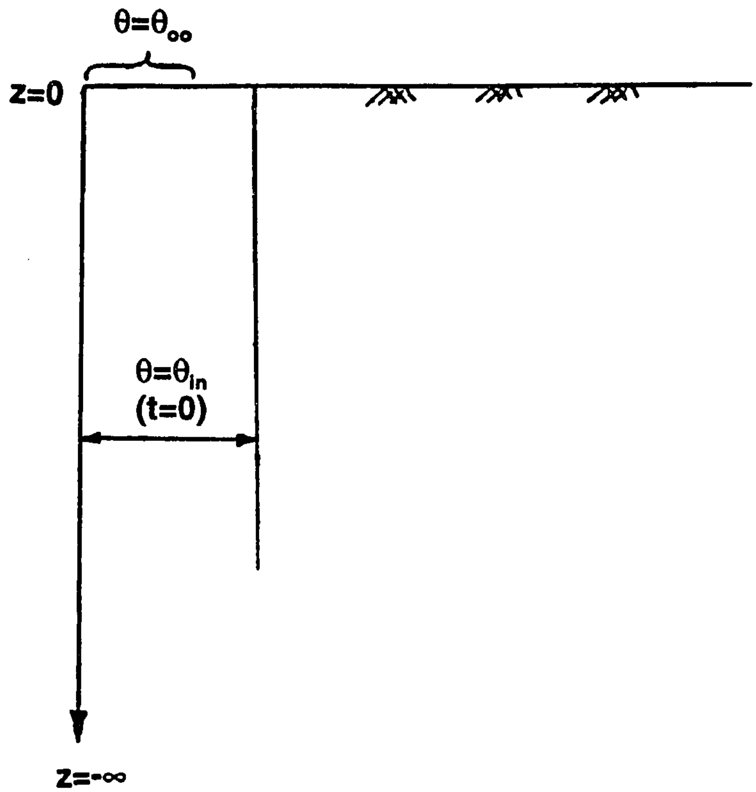


Figure 5-1b. Boundary and Initial Conditions for a One-Dimensional Moisture-Flow Problem (Willis and Yeh, 1987).

- The transmissivity is computed at a certain time level as $T=K_s h^k$, where h^k is the hydraulic head computed at previous time step t=k. For this consideration, transmissivity becomes independent of the dependent variable at any time level of computation; and

- The steady state Dirichlet and Neumann boundary conditions are considered.

Saturated leachate flow equation (2-44) is written in a simplified form as:

$$\frac{\partial h}{\partial t} = \alpha \frac{\partial^2 h}{\partial x^2} + \beta h - p \qquad (5\text{-}4)$$

where,

$$\alpha = \frac{T}{n_e}$$

$$\beta = -\frac{K_c}{n_e d} \text{ and}$$

$$\overline{p} = \frac{-N_\ell + K_c}{n_e}$$

Equation (5-3) is solved for the following initial and boundary conditions:

where,

L = length of the saturated leachate mound from the origin at x=0.

$$h(x, t=0) = h_i$$

$$\frac{\partial h}{\partial x}(x=0, t) = i_o$$

$$h(x=L, t) = h_L$$

The method of separation of variables is used to obtain the solution for equation (5-3) for the given boundary and initial conditions. The hydraulic head h(x,t) at any distance x and at any time t is expressed as:

$$h(x,t) = \sum_{n=0}^{\infty} C_n(t) \mathrm{Cos}\left[\frac{(2n+1)\pi x}{2L}\right] + i_o x\left(1 - \frac{x}{L}\right) + \frac{x^2 h_L}{L^2} \qquad (5\text{-}5)$$

where,

$$C_n(t) = \frac{\Phi_n}{\vartheta} + g_s e^{-\vartheta t}$$

$$\Phi_n = \frac{4\alpha u_o}{L^2}\left[-i_o + \frac{h_L}{L}\right] + \frac{2}{L}\left[-\bar{p}u_o + \beta i_o u_1 - \frac{\beta i_o}{L}u_2 + \frac{\beta h_L}{L^2}u_2\right]$$

$$u_o = \frac{2L}{(2n+1)\pi}(-1)^n$$

$$u_1 = \frac{2L^2}{(2n+1)\pi}\left[(-1)^n - \frac{2}{(2n+1)\pi}\right]$$

$$u_2 = \frac{2L}{(2n+1)\pi}\left[(-1)^n L^2 - \frac{8(-1)^n}{\pi^2}\left(\frac{L}{2n+1}\right)^2\right]$$

$$g_s = \frac{2}{L}\left[h_i u_o - i_o u_1 + \frac{i_o}{L} u_2 - \frac{h_L}{L^2} u_2\right]$$

$$\hat{\vartheta} = \alpha\left[\frac{(2n+1)\pi}{2L}\right]^2 - \beta$$

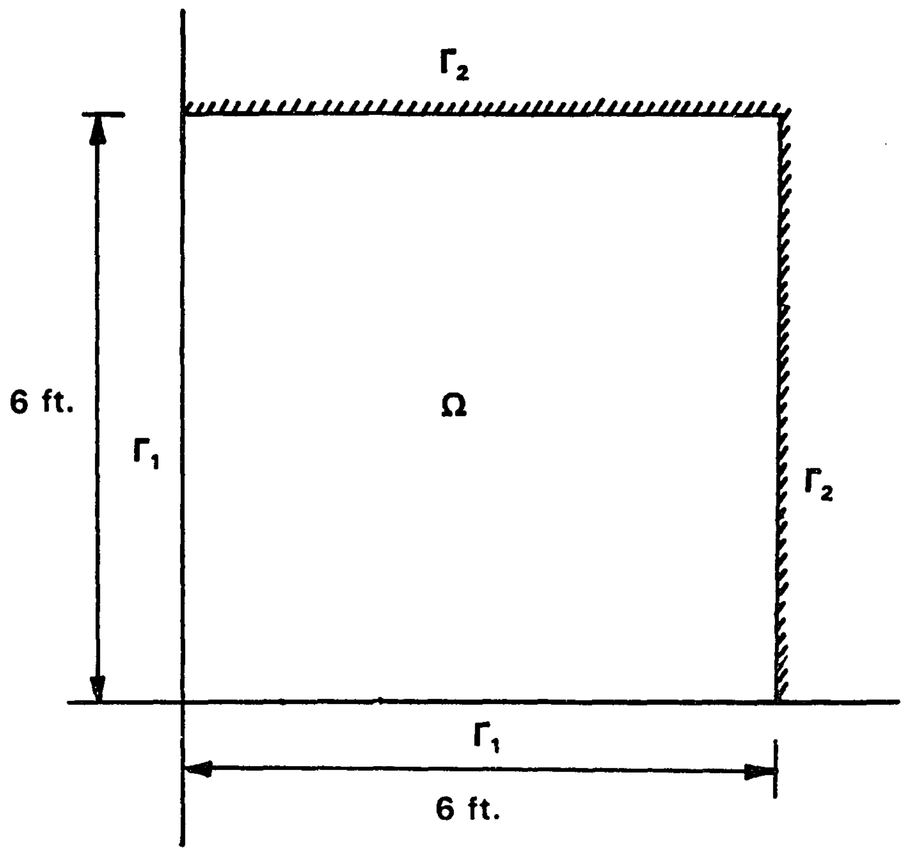
The detail derivation of the above analytical solution is given in Appendix B.

5.2 COMPARISON OF ANALYTICAL AND NUMERICAL SOLUTION

The numerical calculations for the unsaturated leachate flow problem are done on a two-dimensional square domain of size 6 ft. as shown in Figure 5-2. The analytical solution is formulated using simplifying assumptions for the problem defined by Neumann and Dirichlet boundary conditions. The numerical solutions are also obtained for the saturated flow equation. The results for both the unsaturated and saturated flow problems are compared with the exact solutions.

5.2.1 Unsaturated Flow Problem

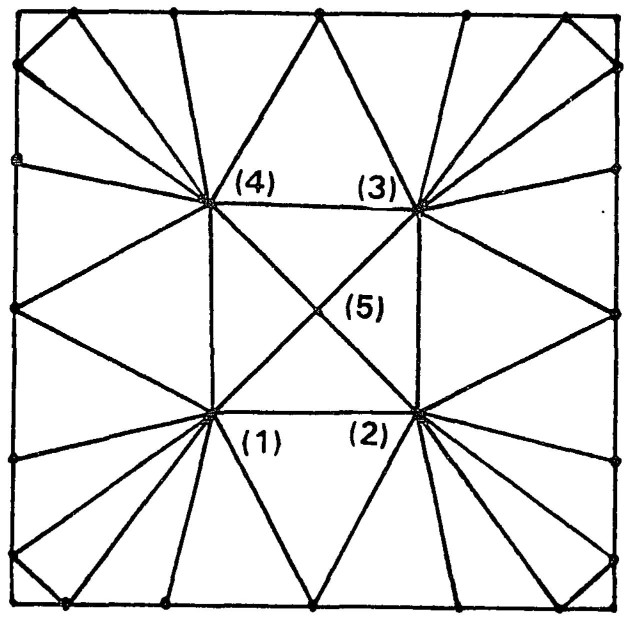
The square region is divided into a series of triangular cells for domain integration in the boundary integral formulation as shown in Figure 5-3. The computational region for the boundary integral solution consists of 20 boundary elements, 28 triangular elements, and 5 internal nodes.



Γ_1 : Dirichlet Boundary Condition

Γ_2 : Neumann Boundary Condition

Figure 5-2. Computational Domain for Comparison of Analytical and Numerical Solution.



(1) : Internal Node No. 1.

Figure 5-3. Computational Domain for Boundary Integral Solution.

For the finite-difference solution, the area is discretized into a network of squares as shown in Figure 5-4. The Neumann boundary condition is considered steady over the computational period. The Dirichlet boundary condition is considered unsteady and the variation is shown Figure 5-5. An initial moisture value of 0.373 is assumed. The other necessary information for the model run are given in Table 5-1.

The direct Green's function and perturbation Green's function boundary integral solutions are compared with the exact solution for node numbers 1 and 3 as shown in Figures 5-6a and 5-6b respectively. Figure 5-6a shows the comparison at x = 2.0 ft. and z = 2.0 ft. and Figure 5-6b shows the comparison at x = 4.0 ft. and z = 4.0 ft. From these Figures, it is observed that the boundary integral solutions obtained by direct Green's function and perturbation Green's function formulations are in close agreement with the exact solution. The two boundary integral techniques give similar results.

The assumption for computing the gravity term $\partial K(\theta)/\partial z$ using the moisture content from the previous time step is made in the analytical solution. In the direct green's function solution the same assumption is made to compute the gravity term. However, the gravity term in the perturbation boundary integral method is obtained from the previous higher order solution in the perturbation quantities. The assumption of considering constant gravity term at a certain time step is not possible to use in the perturbation technique. But agreeable results with the direct

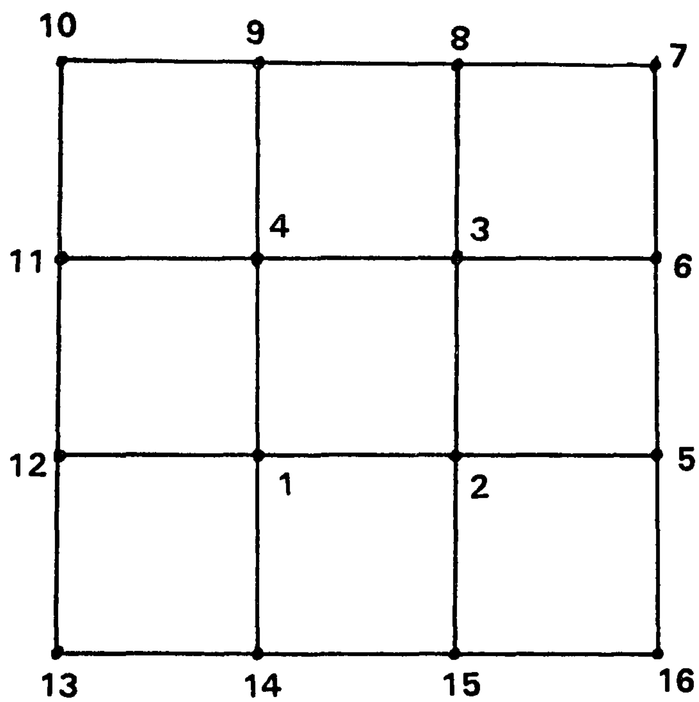


Figure 5-4. Computational Domain for Finite-Difference Solution.

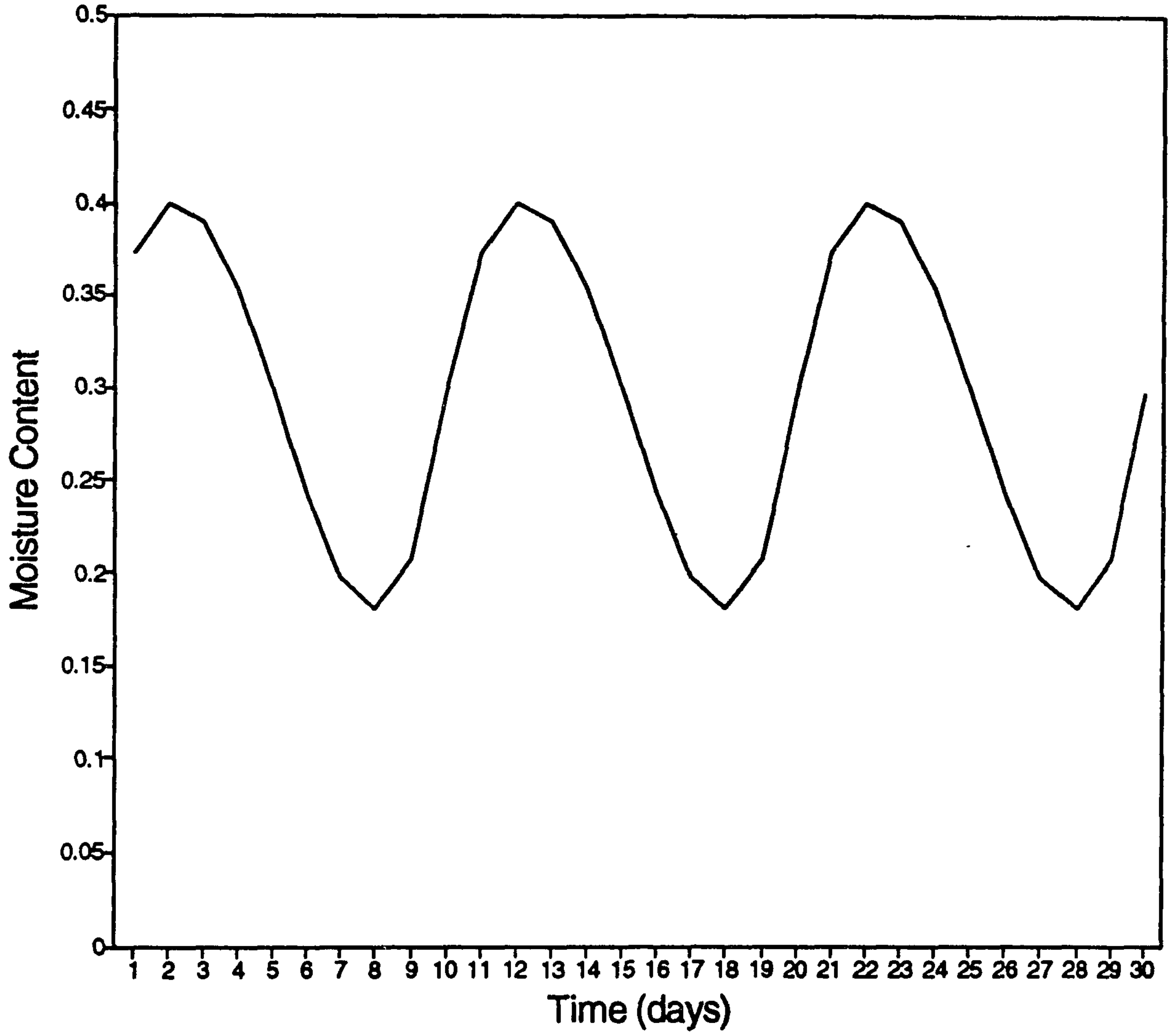


Figure 5-5. Unsteady Dirichlet Boundary Condition.

TABLE 5-1

DATA USED TO SOLVE THE UNSATURATED FLOW EQUATION

Hydraulic conductivity	28 ft./day
Porosity	0.417
Air entry suction head	-2.57 ft.
Campbell's parameter (b)	4.0

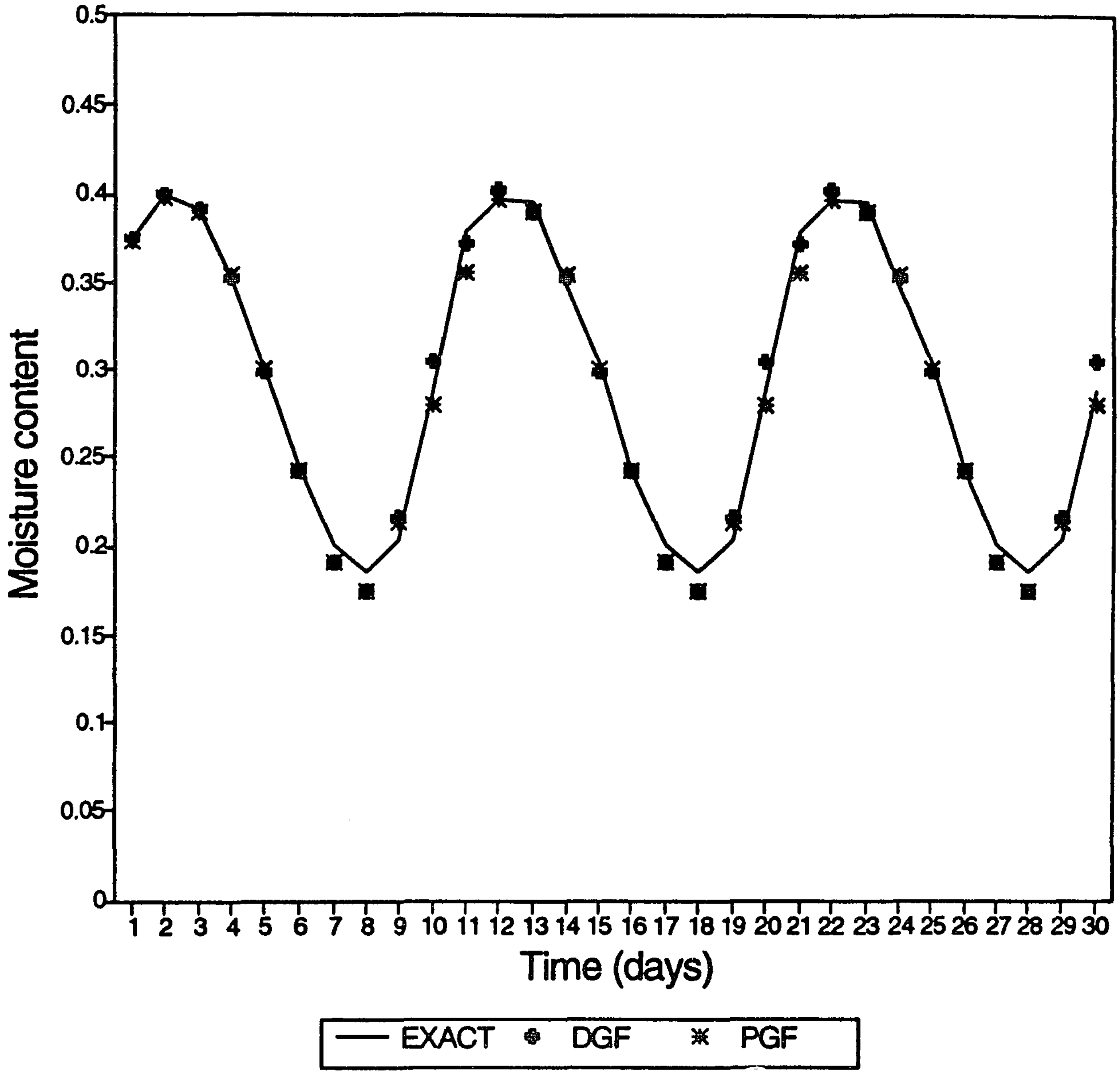


Figure 5-6a. Comparison of Analytical and Boundary Integral Solution at Node No. 1.

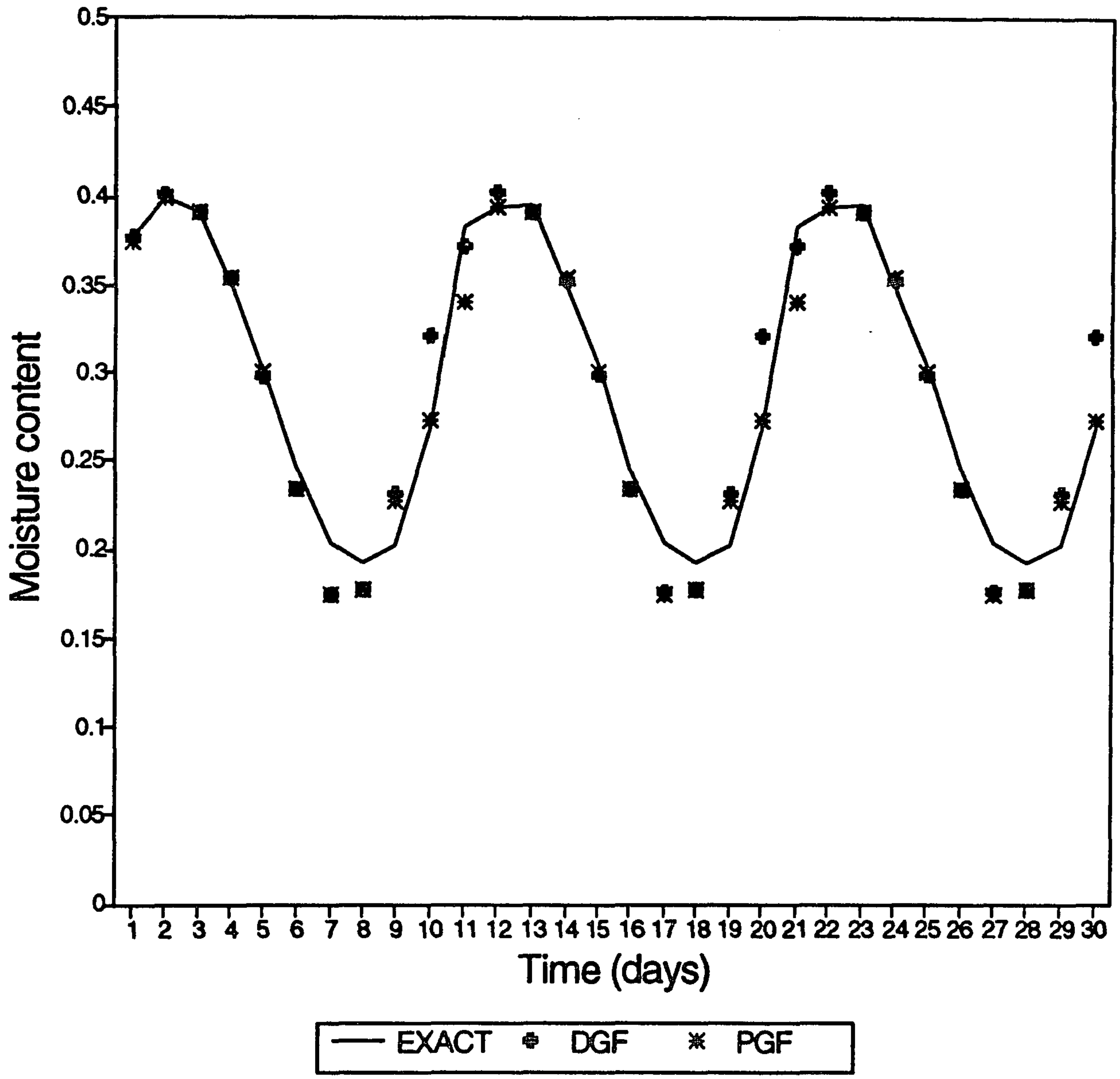


Figure 5-6b. Comparison of Analytical and Boundary Integral solution at Node No. 3.

Green's function and exact solutions are observed.

The finite-difference solution is obtained using the Gauss-Seidel iteration and Gauss-Jordan elimination methods. The direct Green's function boundary integral solution and Gauss-Seidel iteration solution are compared with the analytical solution for internal nodes 1 and 3 as shown in Figures 5-7a and 5-7b respectively. A maximum absolute deviation of 0.04855 from the analytical solution is observed in the case of the Gauss-Seidel iteration method of finite-difference solution at node number 1. However, a maximum absolute deviation of 1.53 percent in moisture content is obtained in the direct Green's function boundary integral solution. The deviations at node number 3 is significant for Gauss-Seidel iteration method of finite-difference solution. A maximum absolute deviation of 0.10779 occurs in the finite-difference solution by Gauss-Seidel iteration method against a maximum absolute deviation of 0.05259 by direct Green's function boundary integral solution at node number 3.

The direct Green's function boundary integral solutions are also compared with the solution obtained by the Gauss-Jordan elimination method as shown in Figures 5-8a and 5-8b. The Gauss-Seidel iteration and Gauss-Jordan elimination methods of finite-difference solution give similar results. The similar results shown by the iteration and elimination methods and the agreeable solution by both the methods with the analytical solution establish the credibility of the finite-difference scheme used for the governing equation. However, the direct Green's function boundary

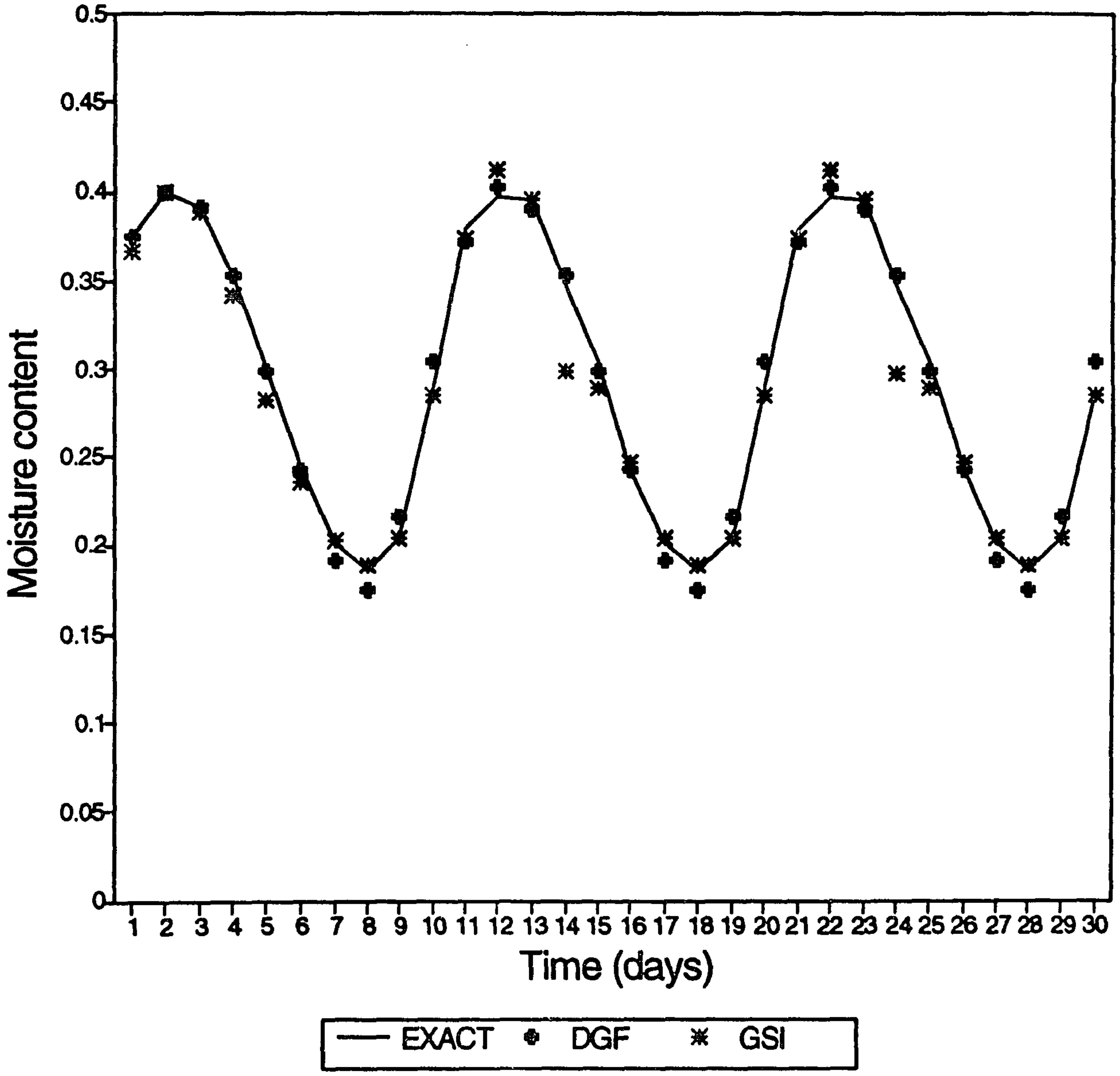


Figure 5-7a. Comparison of Direct Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Seidel Iteration Method at Node No. 1.

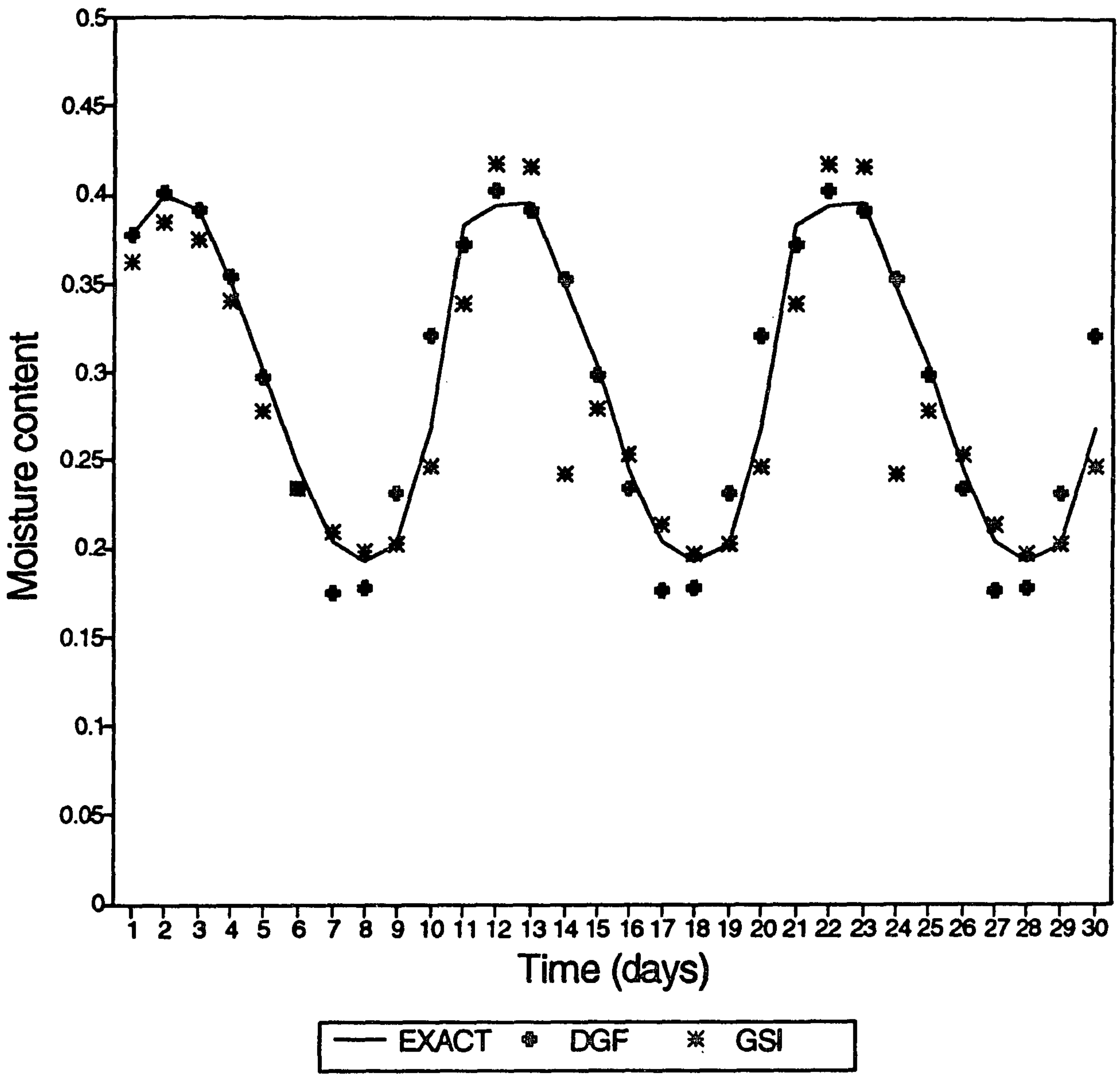


Figure 5-7b. Comparison of Direct Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Seidel Iteration Method at Node No. 3.

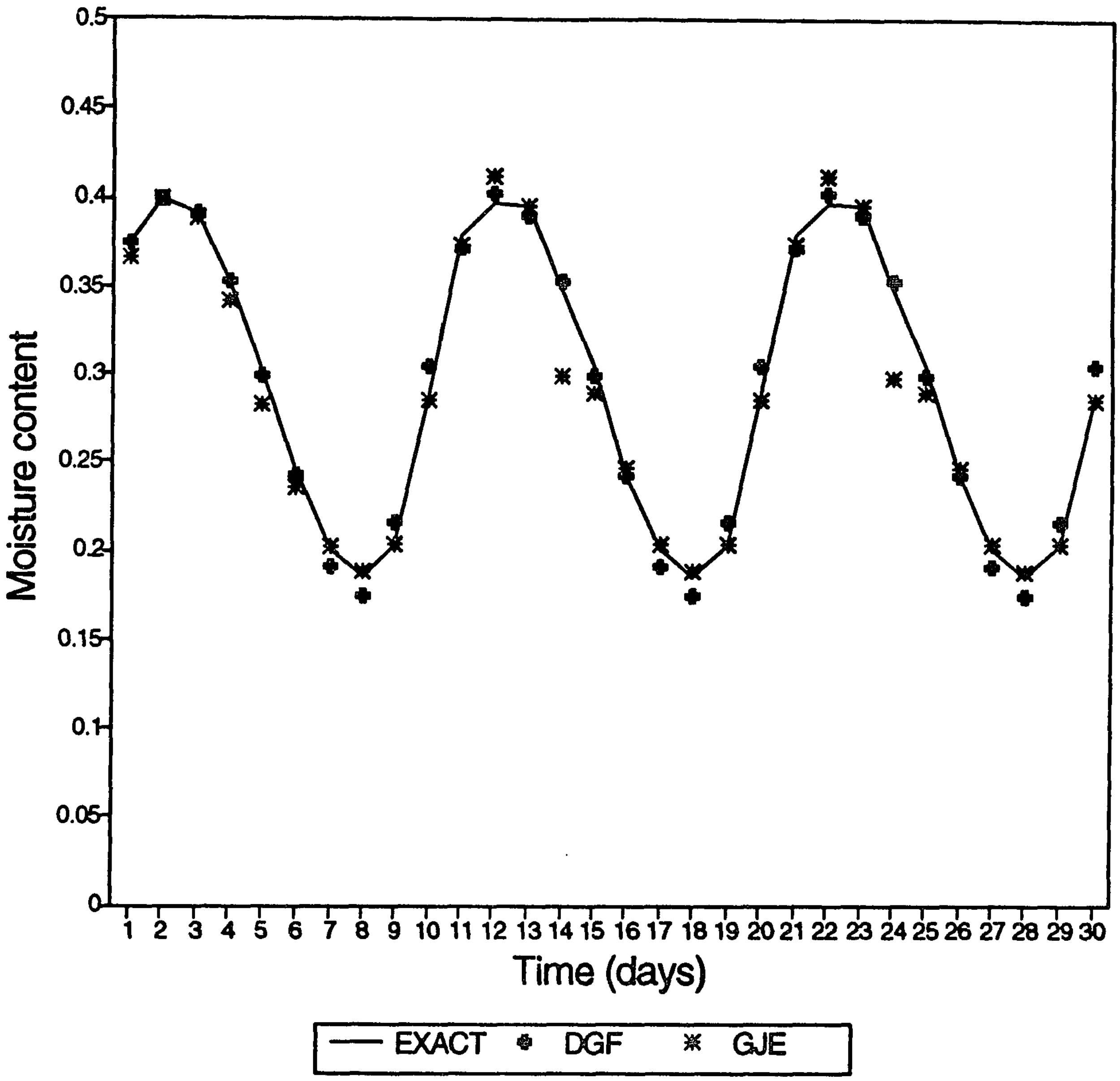


Figure 5-8a. Comparison of Direct Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Jordan Elimination method at Node No. 1.

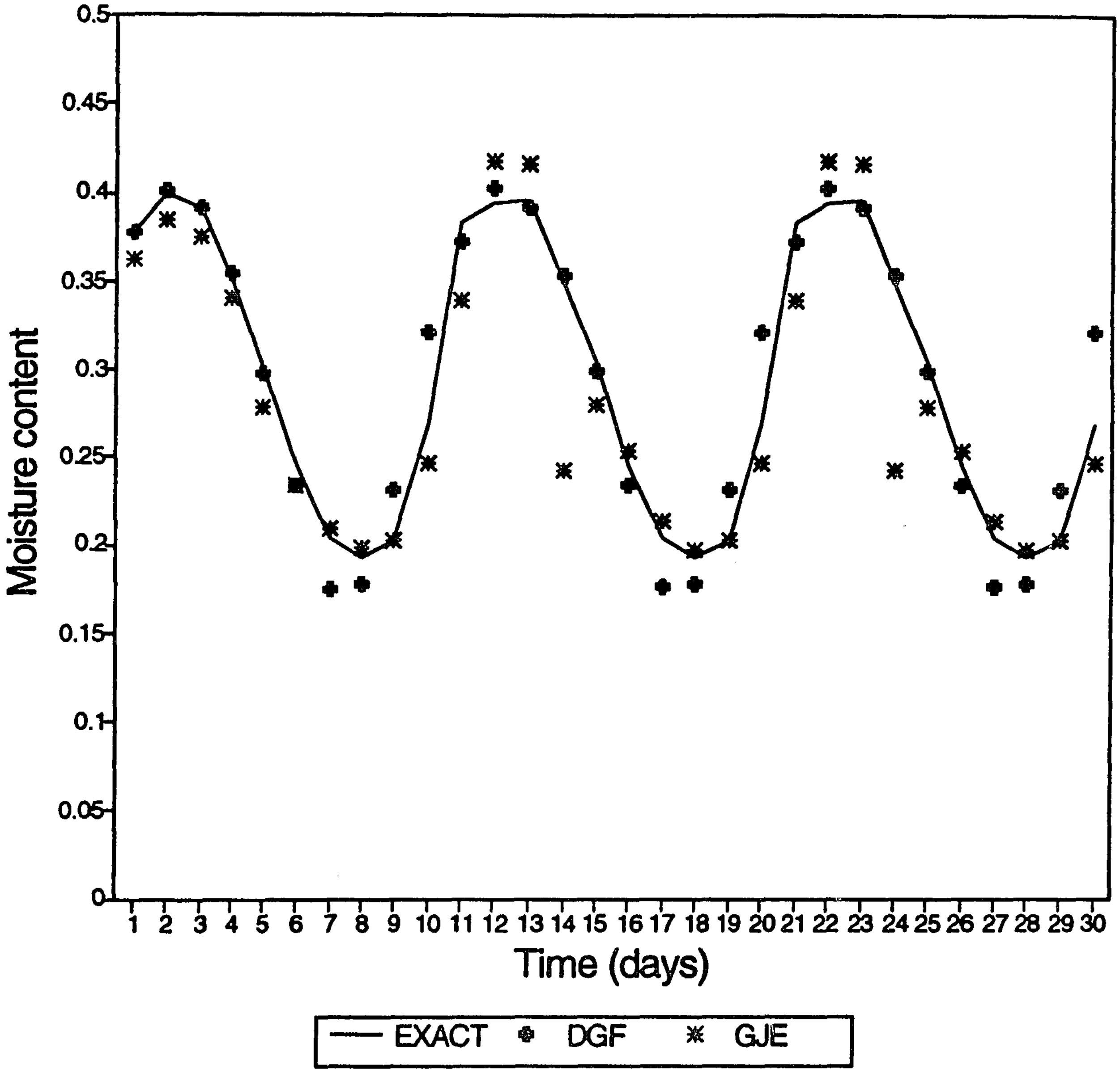


Figure 5-8b. Comparison of Direct Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Jordan Elimination method at Node No. 3.

integral method shows more agreeable results than the finite-difference solution obtained by both the Gauss-Seidel iteration and Gauss-Jordan elimination methods as shown in Figures 5-7 and 5-8. This implies that the boundary integral solution gives more accurate results than the finite-difference method for moisture content in the unsaturated zone.

The results of perturbation boundary integral solution and finite-difference solution using Gauss-Seidel iteration method are shown in Figures 5-9a and 5-9b respectively. A maximum absolute deviation of 2.27 percent of moisture content from the analytical solution is found at node number 1 in the case of perturbation Green's function boundary integral solution. At node number 3, the maximum absolute deviation from analytical solution is small in perturbation Green's function solution, compared to the Gauss-Seidel iteration method of finite-difference solution. The maximum absolute deviation of 0.04368 from analytical solution is observed in perturbation Green's function solution against the absolute deviation of 0.10779 in the Gauss-Seidel iteration method of the finite-difference solution.

The perturbation Green's function boundary integral solution is also compared with the finite-difference solution using Gauss-Jordan elimination method. The comparisons with respect to the analytical solution at node numbers 1 and 3 are shown in Figures 5-10a and 5-10b respectively. The deviation in results by Gauss-Jordan elimination method is similar to the deviation in results obtained by Gauss-Seidel iteration method. The computation of gravity term in the perturbation

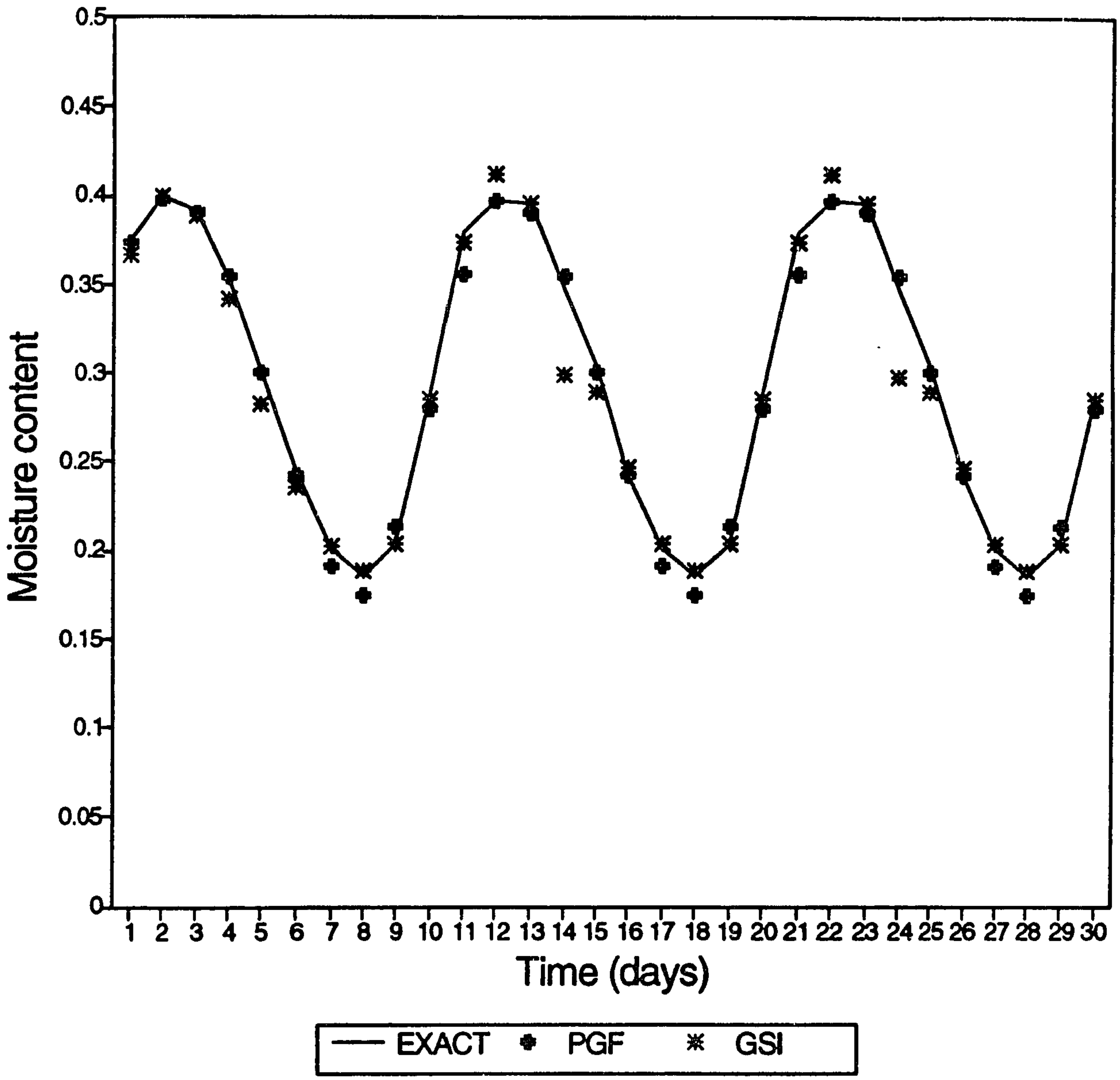


Figure 5-9a. Comparison of Perturbation Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Seidel Iteration Method at Node No. 1.

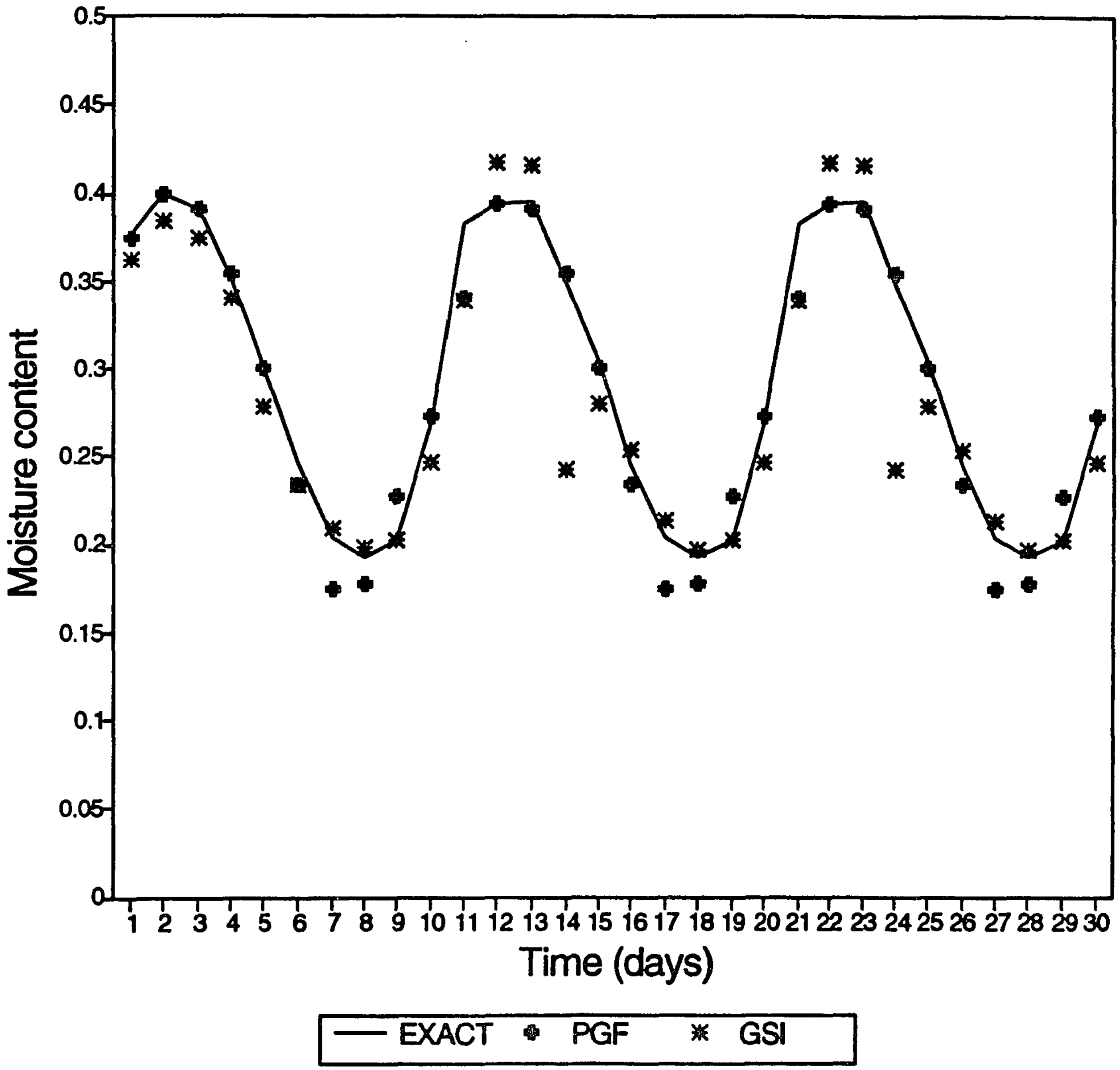


Figure 5-9b. Comparison of Perturbation Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Seidel Iteration Method at Node No. 3.

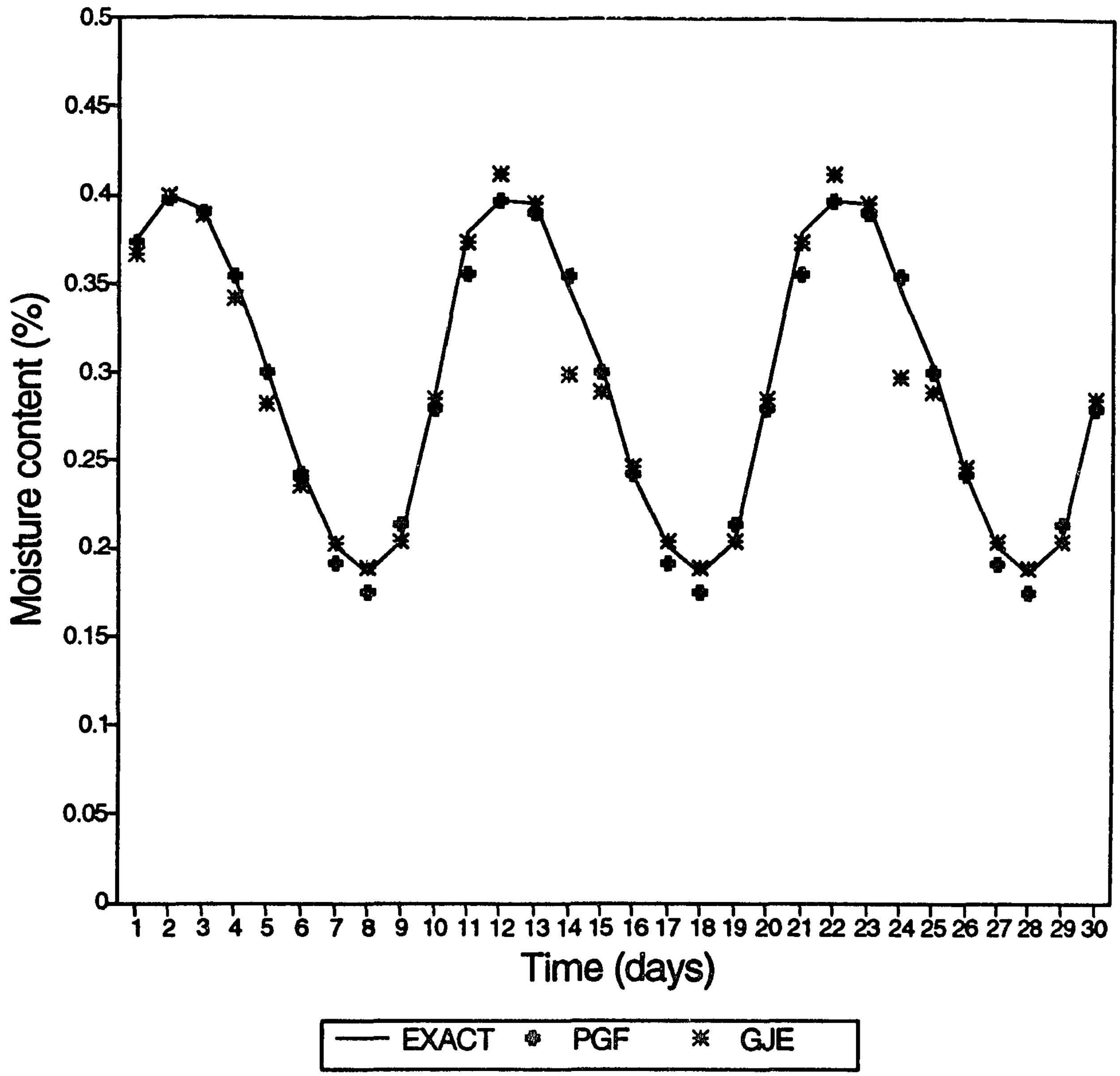


Figure 5-10a. Comparison of Perturbation Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Jordan Elimination Method at Node No. 1.

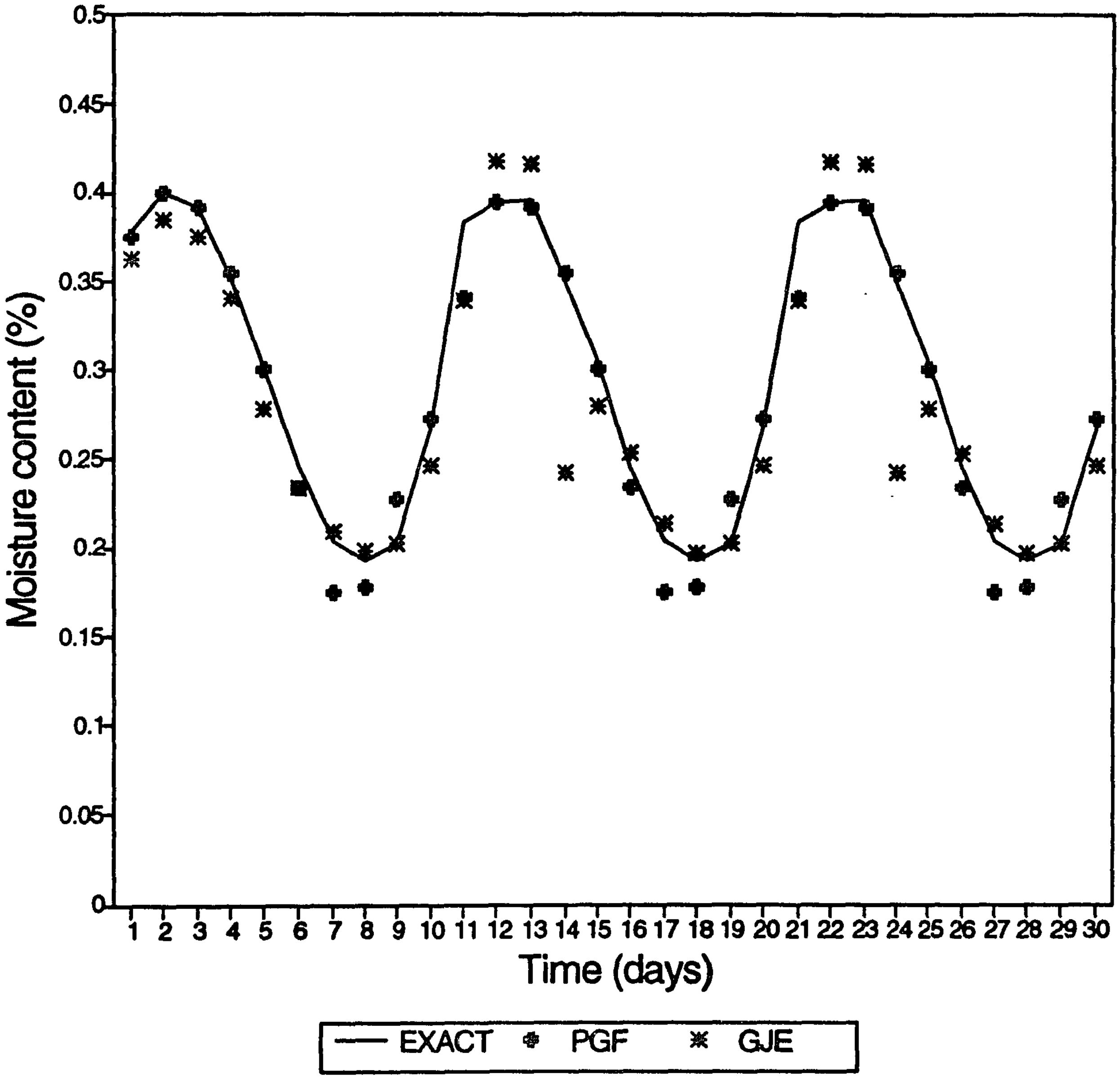
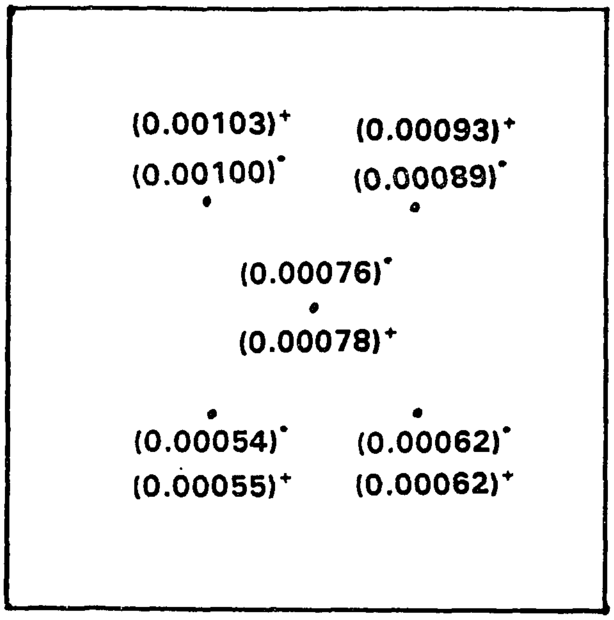


Figure 5-10b. Comparison of Perturbation Green's Function Boundary Integral Solution and Finite-difference Solution by Gauss-Jordan Elimination Method at Node No. 3.

boundary integral solution is done using perturbation series at the current time level. This gives a better representation of the physical system by considering moisture content from the higher order solution of the perturbation series at the current time level to compute the gravity term. It is observed that the perturbation Green's function solution gives more agreeable results compared to the finite-difference solution. Therefore, it can be concluded that the boundary integral method represents the moisture variation in the unsaturated zone of the subsurface more accurately than the finite-difference method.

Unlike the finite-difference method, the velocity at internal points can be computed by boundary integral formulation, once the Dirichlet and Neumann boundary conditions are solved for the boundary nodes. The internal velocities in x- and z-directions are obtained using the expressions (4-29) and (4-30). The internal velocities in x- and z-directions are compared with the analytical solution as shown in Figures 5-11a and 5-11b respectively. Close agreement between the boundary integral and analytical solution shows the ability of the boundary integral method to compute leachate accretion more accurately in the unsaturated zone of a landfill.

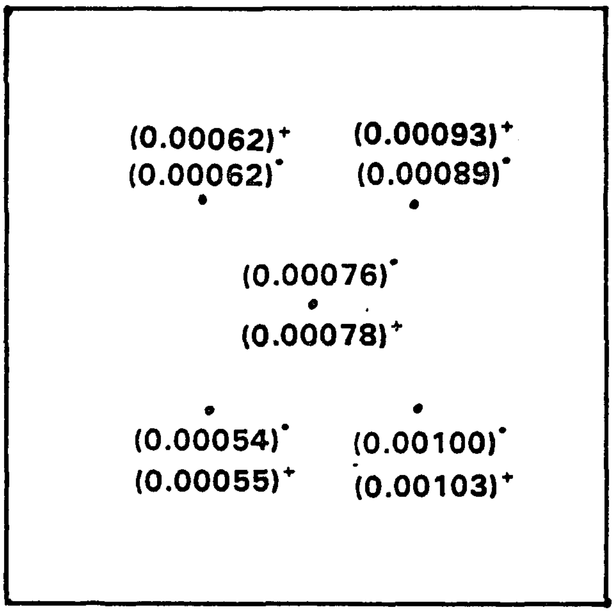
It is valuable to find the differences in the solution obtained by finite-difference and boundary integral techniques from the exact solution. The mean of the absolute values of the deviations between numerical and analytical solutions at each time step is computed for the whole domain. The variation of mean values of the



* Boundary Integral Solution

\+ Exact Solution

Figure 5-11a. Internal Flux in x-direction.



* Boundary Integral Solution

+ Exact Solution

Figure 5-11b. Internal Flux in z-direction.

absolute error is shown in Figure 5-12. It is observed that the errors in the direct Green's function and the perturbation Green's function boundary integral solutions are less compared to the finite-difference solution.

A further investigation of the boundary integral solution is done by considering 40 boundary elements as shown in Figure 5-13. The domain integral for the direct Green's function boundary integral solutions are evaluated by discretizing the domain using 56 triangular cells and 40 nodes at the boundary as shown in Figure 5-13. The analytical and boundary integral solutions at node numbers 1 and 5 are shown in Figures 5-14 and 5-15 respectively. The boundary integral solutions are in close agreement with the exact solutions as shown in both the Figures 5-14 and 5-15 for node numbers 1 and 5 respectively.

The deviation of the boundary integral solution from the exact solution decreases by 1.34 percent using 40 boundary elements compared to the results using 20 boundary elements. This shows that the deviations of the boundary integral solution from the exact solution do not decrease significantly by increasing the number of elements from 20 to 40 at the boundary.

The present two-dimensional direct Green's function boundary integral model is also used to simulate the one-dimensional moisture-flow problem in a soil column with the Dirichlet (Γ_1) and Neumann (Γ_2) boundary conditions as shown in Figure 5-16. The soil column is descretized using 44 boundary elements and 80 triangular

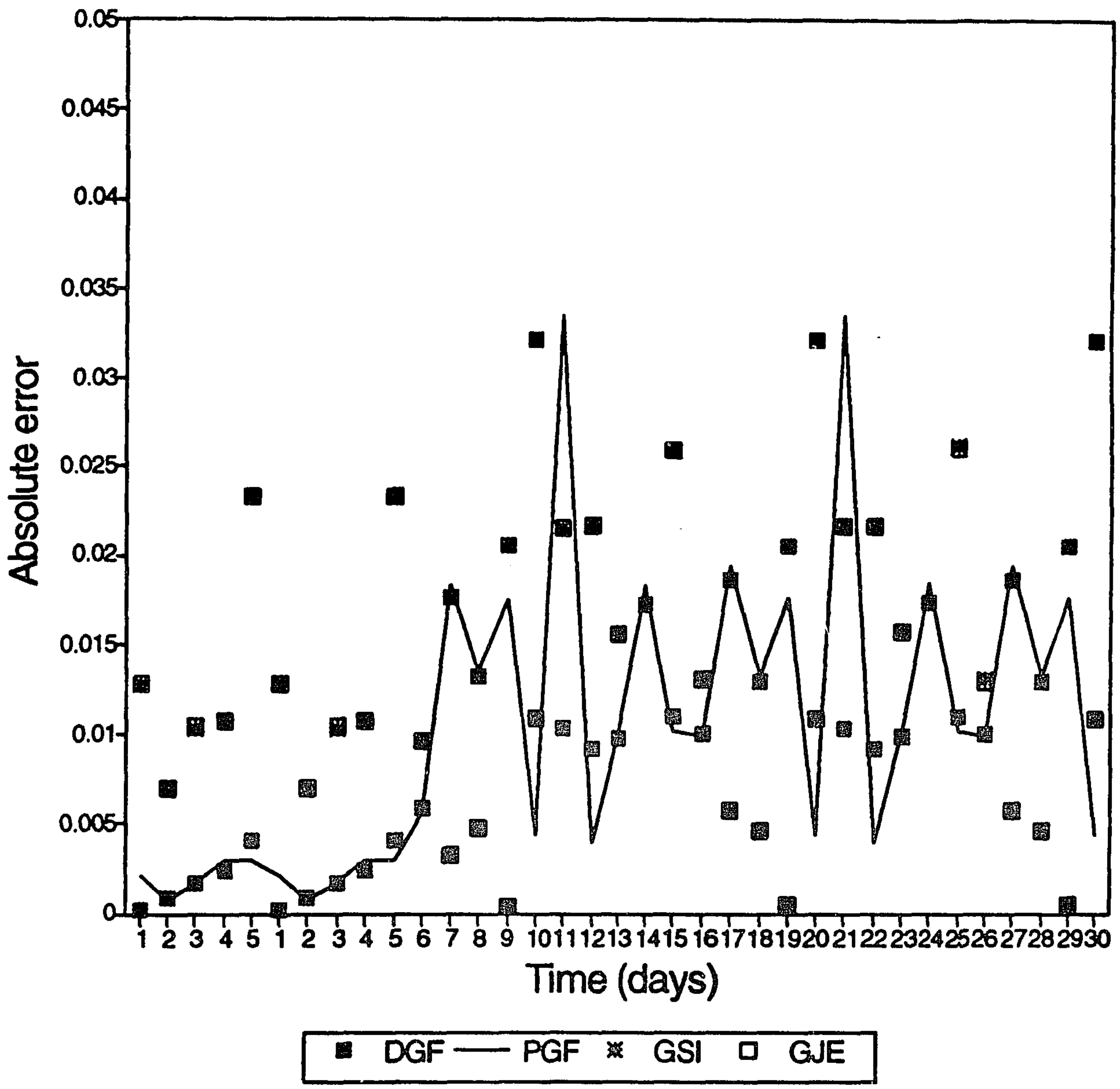
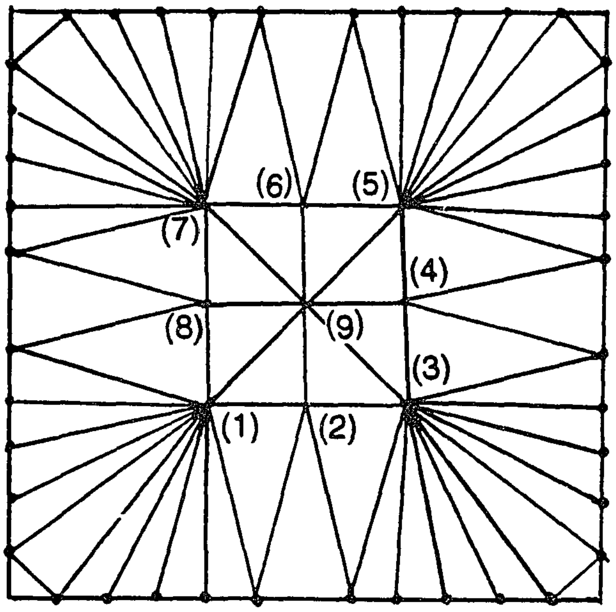


Figure 5-12. Variation of Mean Absolute Error in Boundary Integral and Finite-difference Solution.



(1) : Internal Node No. 1.

Figure 5-13. Computational Domain for Boundary Integral Solution with 40 Elements at the Boundary.

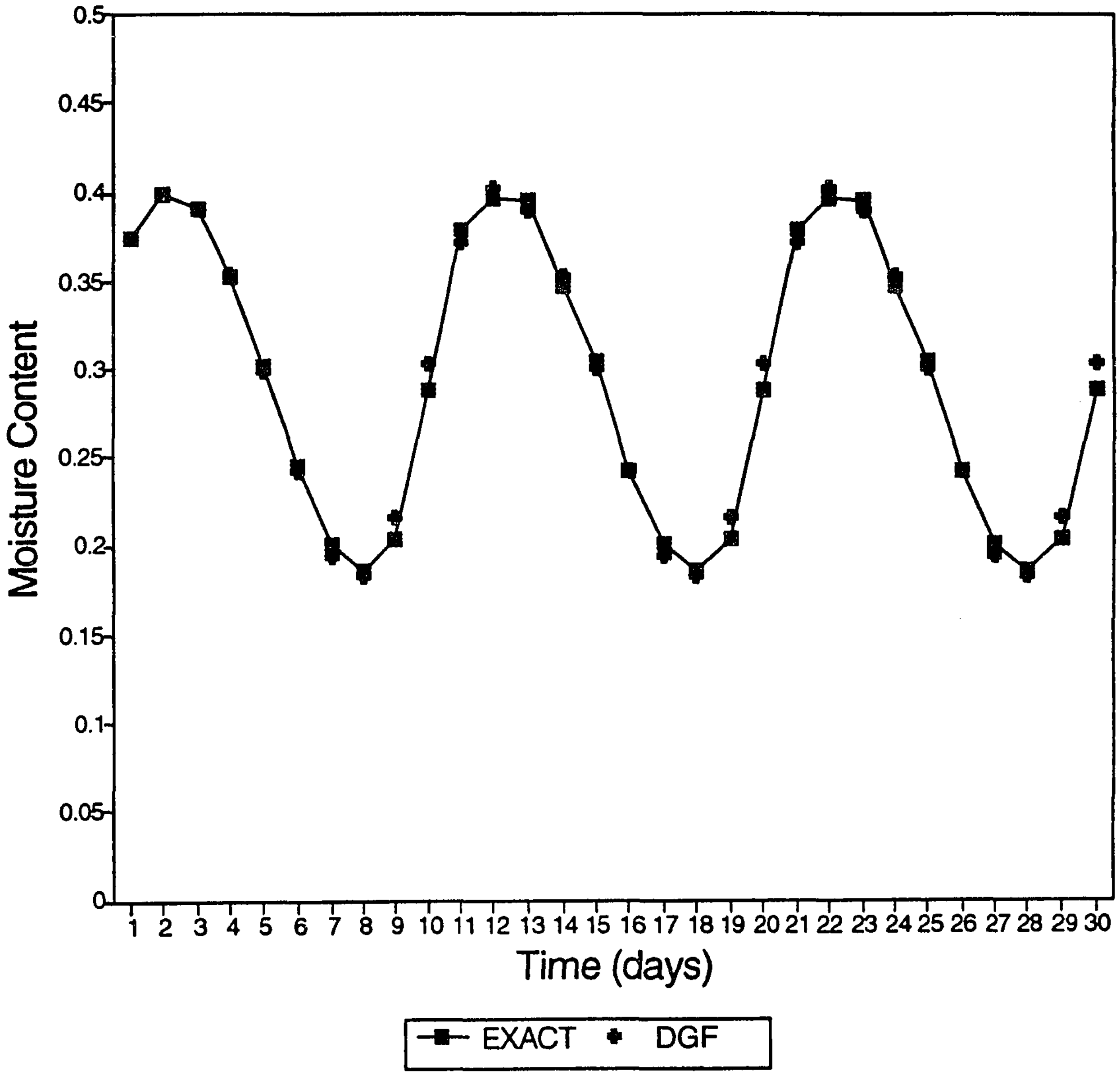


Figure 5-14. Comparison of Analytical and Boundary Integral Solution at Node Number 1 with 40 Elements at the Boundary.

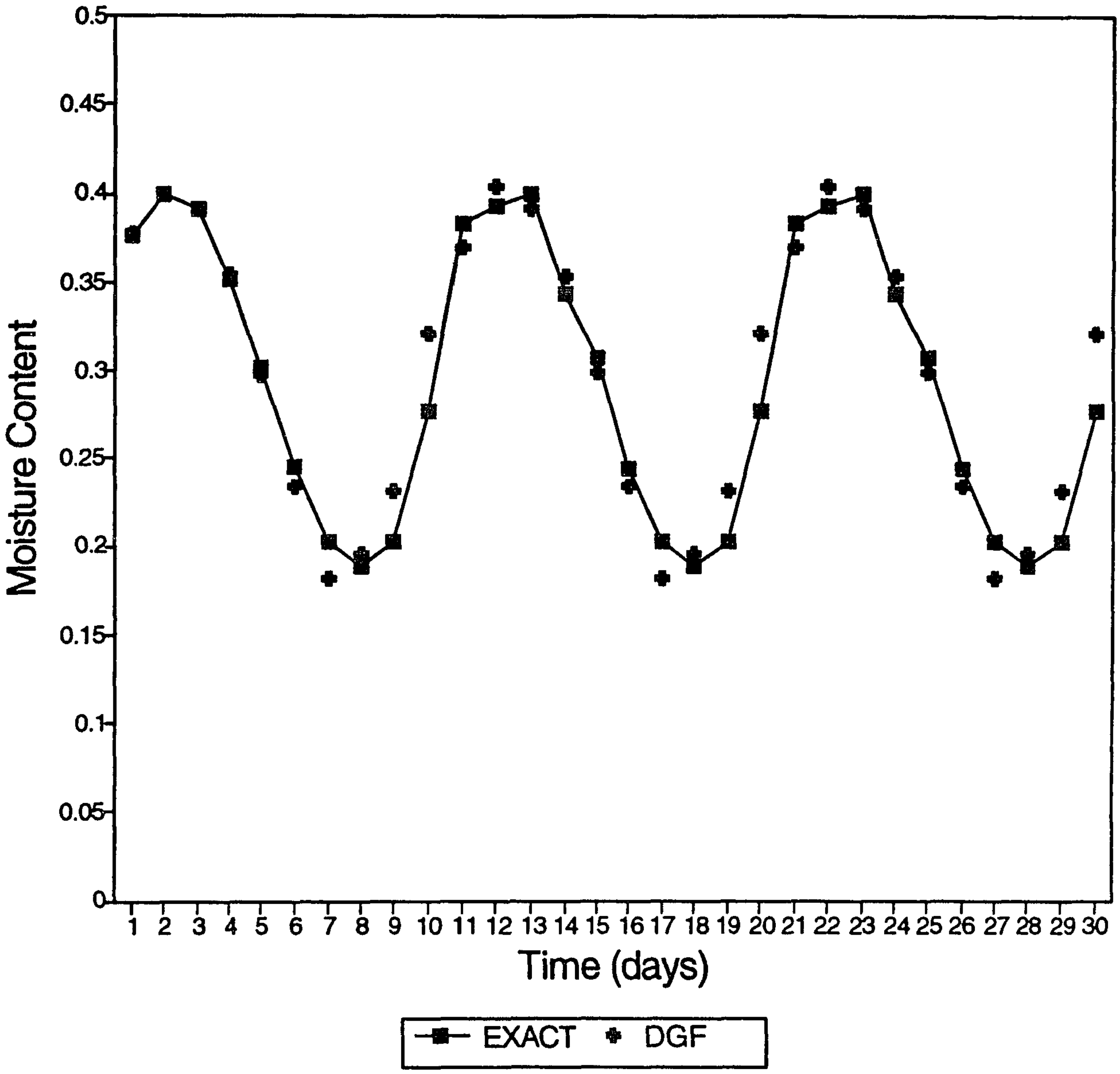
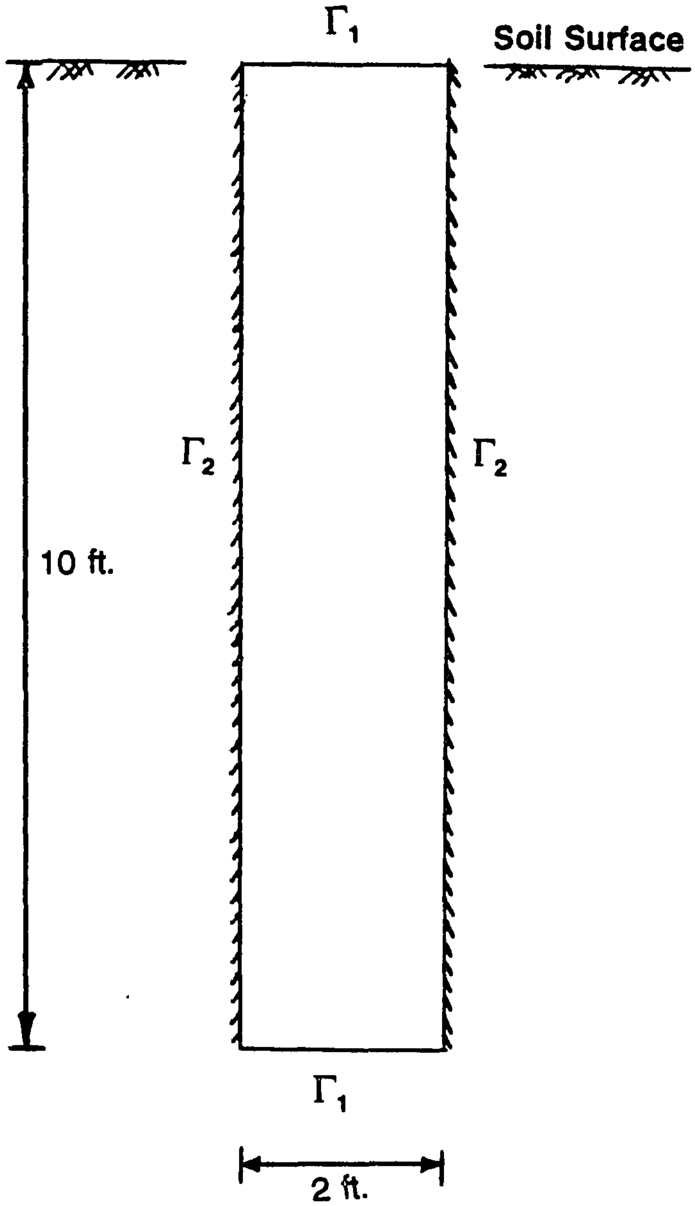


Figure 5-15. Comparison of Analytical and Boundary Integral Solution at Node Number 5 with 40 Elements at the Boundary.



Γ_1 : Dirichlet Boundary Condition

Γ_2 : Neumann Boundary Condition

Figure 5-16. Computational Domain for Comparison of Analytical and Boundary Integral Solution for One-Dimensional Moisture-Flow Problem.

cells as shown in Figure 5-17.

The exact solution in the soil column is obtained using the one-dimensional form of the moisture-flow equation (5-1) by disregarding the gravity term. The one-dimensional analytical solution is expressed in equation (5-1). The analytical and direct Green's function boundary integral solutions are obtained for 19 internal nodes in the computational domain. The comparisons of the direct Green's function boundary integral and exact solutions are made for node numbers 10 and 16 as shown in Figures 5-18 and 5-19 respectively. The variation of the moisture contents predicted by the direct Green's function boundary integral technique for the one-dimensional moisture-flow problem is in close agreement with the variation obtained by the exact solution.

A comparison of the central processing unit time corresponding to the finite-difference and boundary integral solutions is made by running the finite-difference and boundary integral models in an IBM compatible 386 25 MHz personal computer. It is observed that for perturbation Green's function boundary integral solution the central processing unit (CPU) time required is equal to 3 times the CPU time required in direct Green's function boundary integral solution. The more CPU time required in the perturbation Green's function solution is due to the repeated solution required to compute the perturbation quantities.

The boundary integral method of solution is observed to be more time consuming

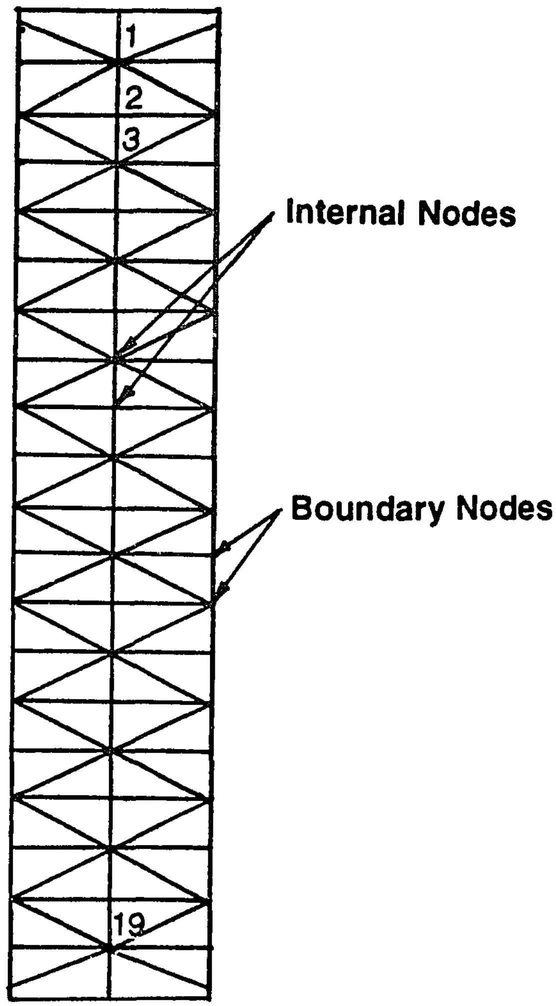


Figure 5-17. Computational Domain for Boundary Integral Solution for One-Dimensional Moisture-Flow Problem.

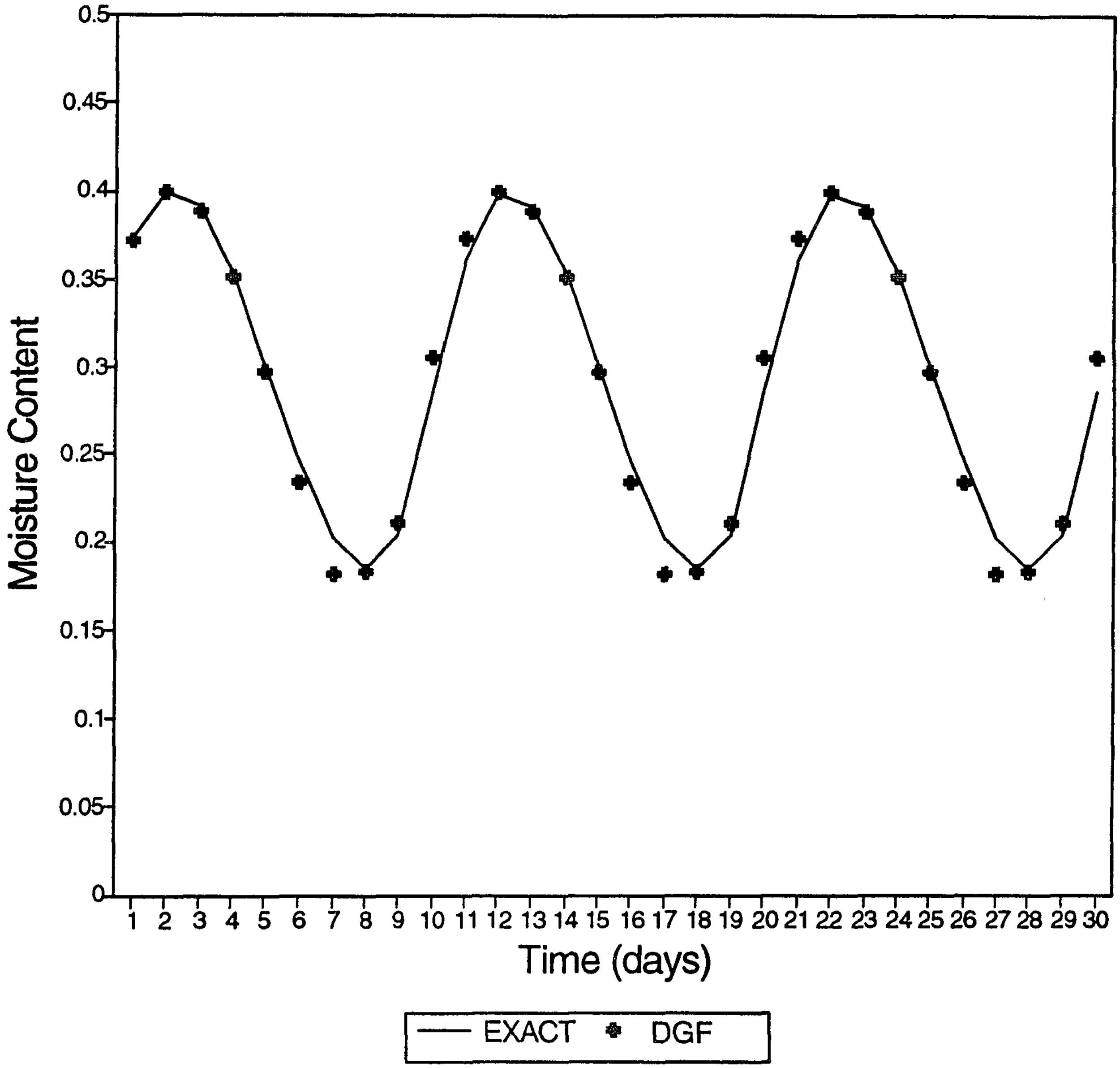


Figure 5-18. Comparison of Analytical and Boundary Integral Solution at Node Number 10 for One-Dimensional Moisture-Flow Problem.

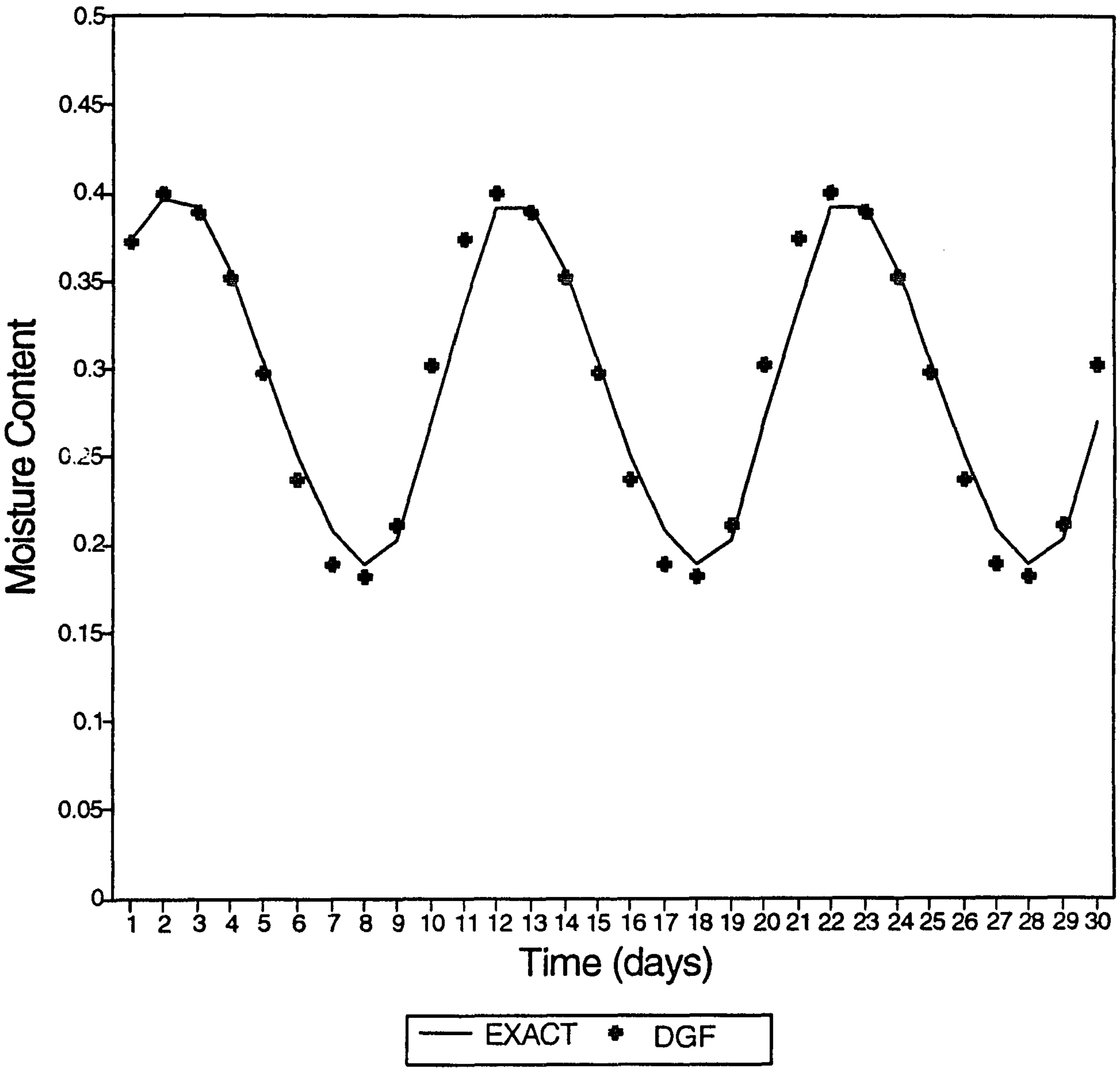


Figure 5-19. Comparison of Analytical and Boundary Integral Solution at Node Number 16 for One-Dimensional Moisture-Flow Problem.

than the finite-difference method of solution. In the direct Green's function boundary integral solution the CPU time required is 172 seconds but in Gauss-Seidel iteration finite-difference solution, the CPU time is only 10 seconds. The more CPU time required in the boundary integral computation is due to more mathematical operations needed to obtain the solution.

The input data file required more computer storage in the boundary integral scheme than a regular finite-difference grid network. The more computer storage requirement in the boundary integral method is due to storing coordinates of the boundary and internal node points. The schematization of the domain can not be avoided for the unsaturated leachate flow problem in the boundary integral method. The domain is discretized into triangular elements and the geometric data of the elements are also stored for the evaluation of the area integrals in the boundary integral formulation. However, in a regular finite-difference scheme of equal nodal areas only the space intervals Δx and Δz are stored as single values and no array is required to store the geometric data of the computational domain. This reduces the computer storage requirement in the finite-difference method.

5.2.2 Saturated Flow Problem

The simplest saturated leachate flow problem can be defined by a saturated leachate mound in the refuse on the top of a bottom clay layer as shown in Figure 5-20. To compare with the analytical solution, the angle of inclination γ of the bottom clay layer (as shown in Figure 2-4) is considered negligible and is assumed to be zero. The thickness of the bottom clay layer is considered 2 feet with a saturated hydraulic conductivity of 10^{-7} cm/sec. A steady leachate accretion of 0.0025 ft./day is assumed. The hydraulic conductivity of the drainage layer is considered 28 ft./day.

The leachate mound head is computed at each time step for a space interval of $\Delta x = 20$ ft. The mound extended for a horizontal length of 105 ft with 20 nodes in which the leachate mound-head is computed. The two finite-difference solution techniques, Gauss-Seidel iteration and Gauss-Jordan elimination are investigated by comparing with the analytical solution.

The saturated leachate flow model was run for $\Delta t = 1$ day for a total time period of 10 days. The numerical solutions are compared for node number 8 and 10 as shown in Figures 5-21 and 5-22 respectively. Both the Gauss-Seidel iteration and Gauss-Jordan elimination methods give identical results. The solution obtained by both methods are in close agreement with the exact solution. The similarity in the solution obtained by both the iteration and elimination methods establishes the credibility of the saturated flow model for it's applicability in real field conditions. Although the comparisons are made using both Gauss-Seidel iteration and Gauss-Jordan elimination methods, yet in the main FILL model Gauss-Jordan elimination

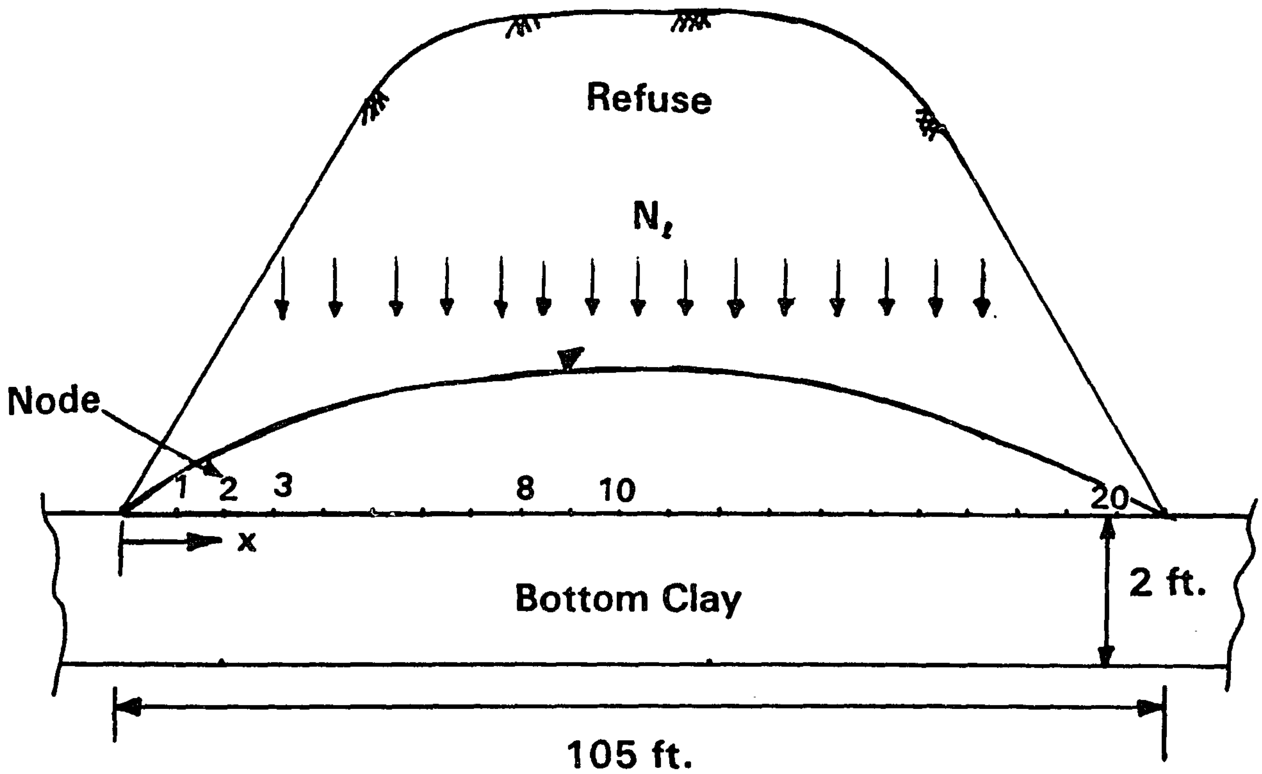


Figure 5-20. Computational Domain for Saturated Flow Problem.

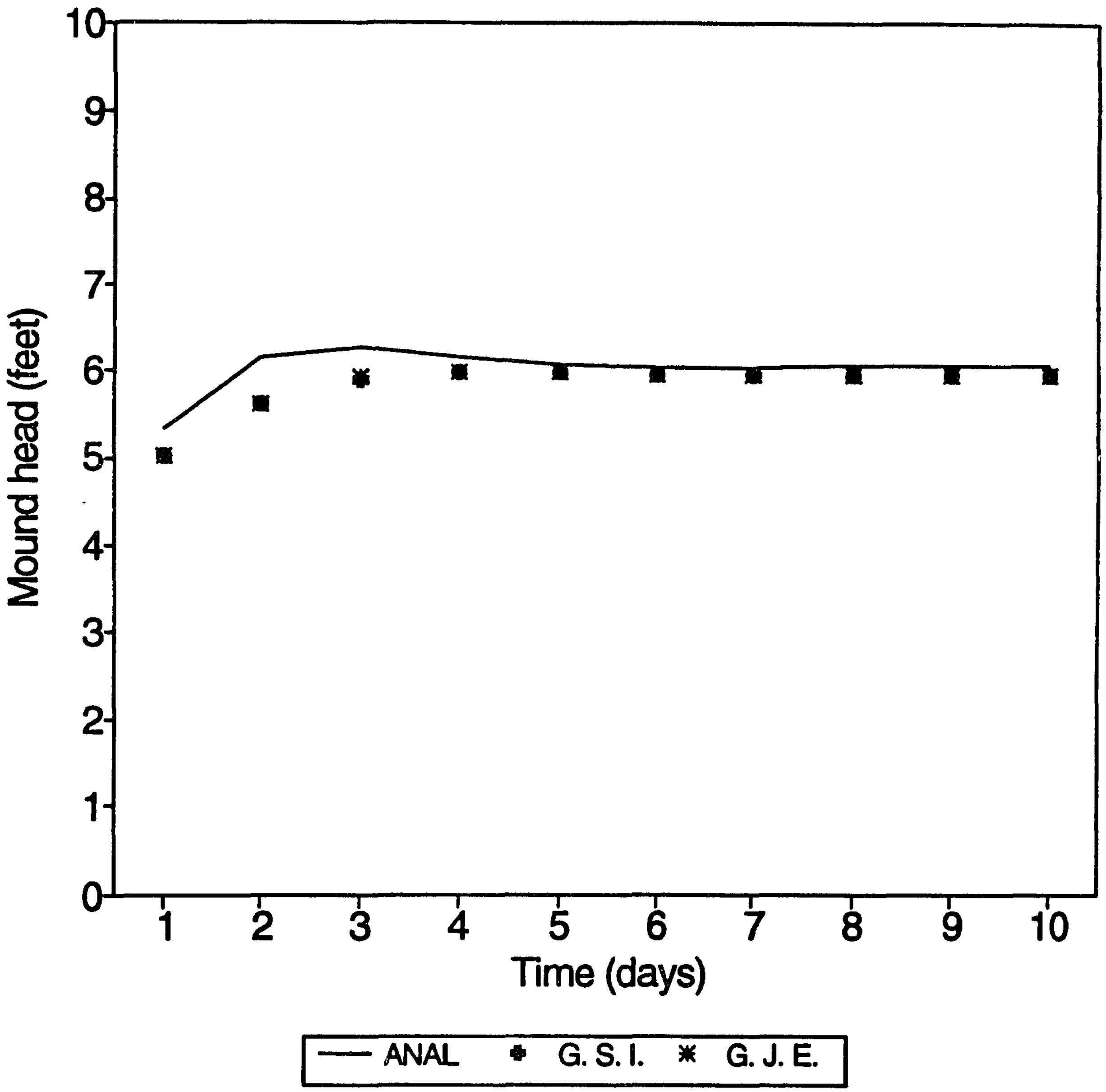


Figure 5-21. Variation of Leachate Mound-head at Node Number 8.

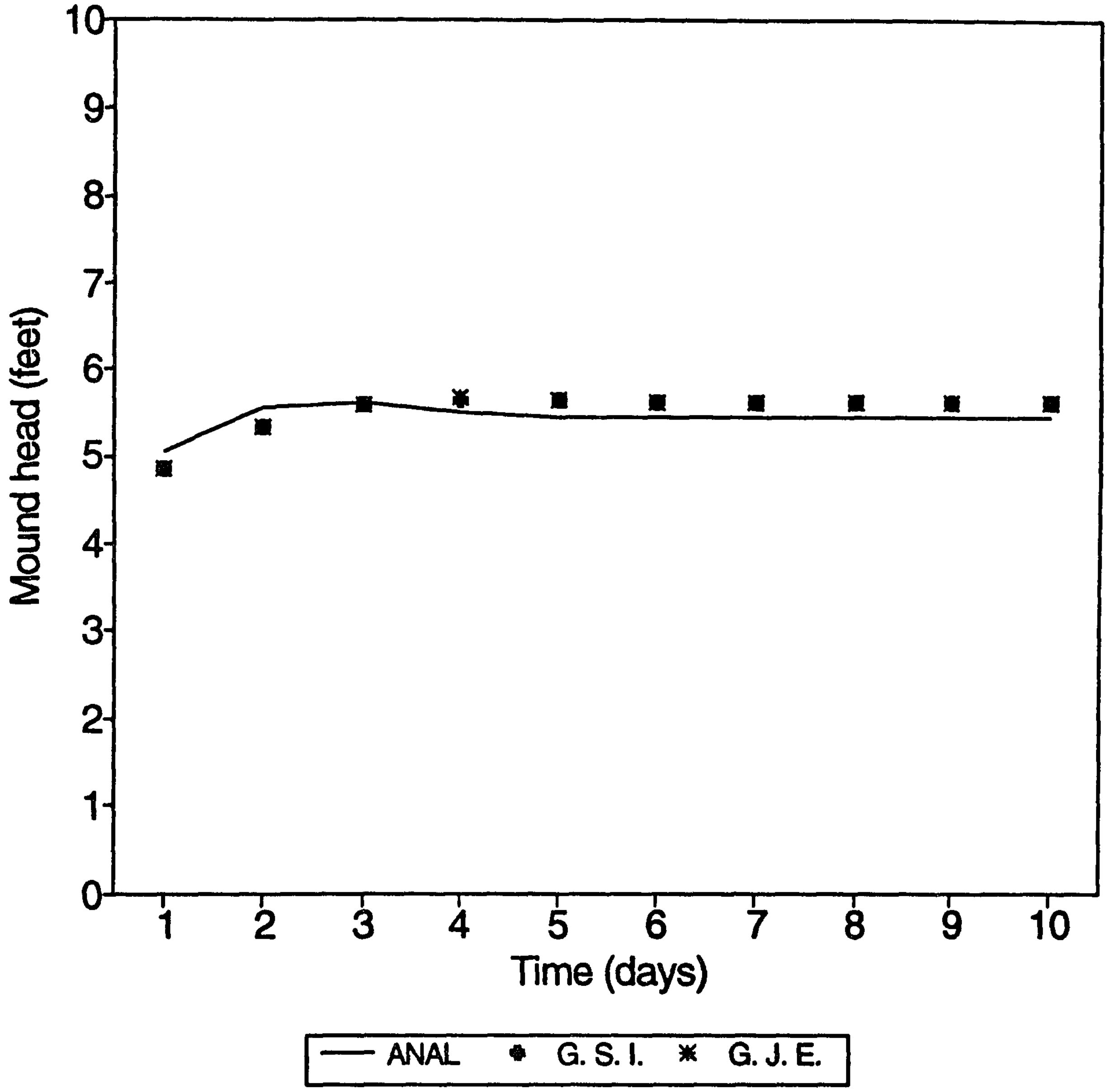


Figure 5-22. Variation of Leachate Mound-head at Node Number 10.

method is used to avoid excessive computing time required for iteration due to large accretion from the unsaturated zone.

5.3 CONCLUSIONS ON COMPARISON OF RESULTS

The conclusions following this chapter can be summarized as follows:

- The boundary integral solution using direct Green's function and perturbation Green's function boundary integral approaches show close agreement with the exact solution for the unsaturated moisture-flow equation. The close agreement of the solution by both approaches establishes the validity of the extension of the boundary integral method for solving flow problems in the unsaturated zone.

- The difference in finite-difference solution using Gauss-Seidel iteration and Gauss-Jordan elimination methods is insignificant. The close agreement of the solution by both iteration and elimination methods establishes the credibility of the finite-difference technique used in the FILL model for moisture-flow equation.

- Both the direct Green's function and the perturbation Green's function boundary integral solution show close agreement to the exact solution compared to the results obtained by finite-difference techniques using both Gauss-Seidel iteration and Gauss-Jordan elimination methods. The perturbation Green's function solution gives more agreeable results compared to the direct Green's function boundary integral and finite-difference solution.

- Unlike the finite-difference method, extension of the boundary integral method gives satisfactory success in providing information of moisture-flux by directly differentiating the moisture content function in the domain. The boundary integral solution shows close agreement of the moisture-flux compared to the exact solution.

- The finite-difference solution using the Gauss-Seidel iteration and Gauss-Jordan elimination methods for saturated leachate flow equation shows close agreement to the exact solution. Both the Gauss-Seidel iteration and elimination methods give identical results. However, to avoid time-consuming iteration process for large leachate accretion from the unsaturated zone, it is found suitable to use Gauss-Jordan elimination method for real field simulations.

CHAPTER 6

FILL MODEL APPLICATION

The FILL model, developed using finite-difference technique, represents flow in both unsaturated and saturated zones of a landfill. The unsaturated flow model generates the time-history of leachate accretion to the saturated leachate mound. The saturated leachate flow equation is solved to predict leachate mound-head using recharge from the unsaturated zone as the source term.

The FILL model is applied to an example landfill section to demonstrate the effectiveness of the model to predict leachate flow rates in a landfill. The data are chosen from a local landfill section. Before applying the FILL model, a sensitivity test of the input parameters are done to observe the changes in the model results for the extreme values of the parameters. All these results are presented in this chapter.

6.1 SENSITIVITY TEST

Sensitivity analysis is done to see how the model reacts to extreme values of the input parameters. Variations in the value of leachate flow parameters have unequal effects on the model results. Some of the parameters have greater effects than the others. The important input parameters for the model are saturated hydraulic conductivity, field capacity, porosity, slope of moisture characteristic curve or Campbell's parameter (b), capillary pressure head, temperature, and solar radiation.

A rigorous sensitivity analysis is done by making arbitrary changes in the aforementioned parameters. The model run was made to observe the effect on the model results for 5 percent, 10 percent, 20 percent, 35 percent, 40 percent, and 50 percent change in the values of the input parameters. The results are shown in Figures 6-1 to 6-13. Figure 6-1 shows the effect of the variation of hydraulic conductivity upon recharge to the saturated leachate mound in a landfill. Decrease in the value of hydraulic conductivity by 50 percent causes a decrease in the value of recharge by 6.31 percent as shown in Figure 6-1. The hydraulic conductivity is observed to be very sensitive to the lateral flow computed by the FILL model. Increase in the value of the hydraulic conductivity by 50 percent causes an increase of 39.35 percent in the lateral flow as shown in Figure 6-2.

The change in recharge is moderate due to the change in the value of field capacity as shown in Figure 6-3. In the FILL model, the field capacity is considered as the initial and irreducible moisture content to compute leachate accretion in the unsaturated zone. The nonlinear variation in recharge for change in field capacity is also due to the fact that the leachate accretion in the unsaturated zone of a

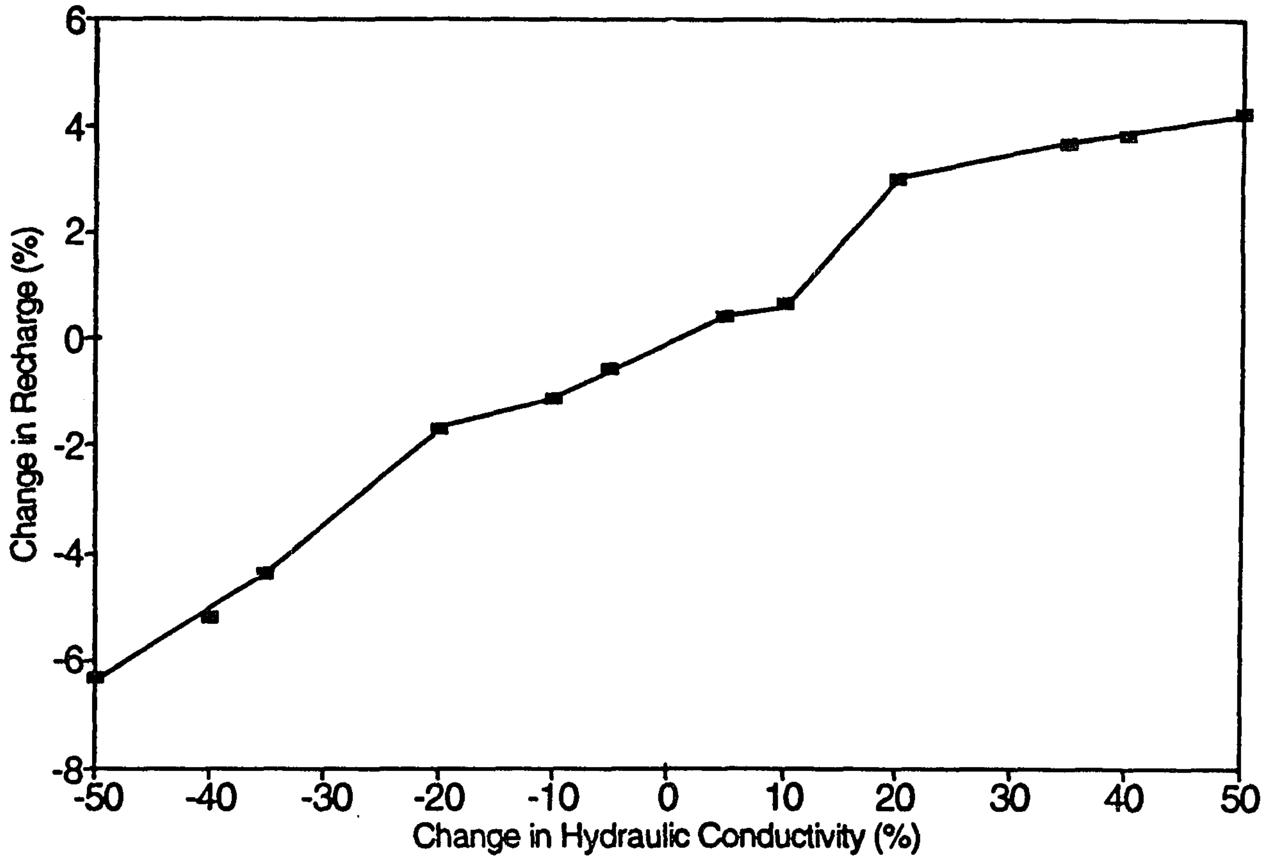


Figure 6-1. Sensitivity of Hydraulic Conductivity to Annual Recharge Computed by the FILL Model.

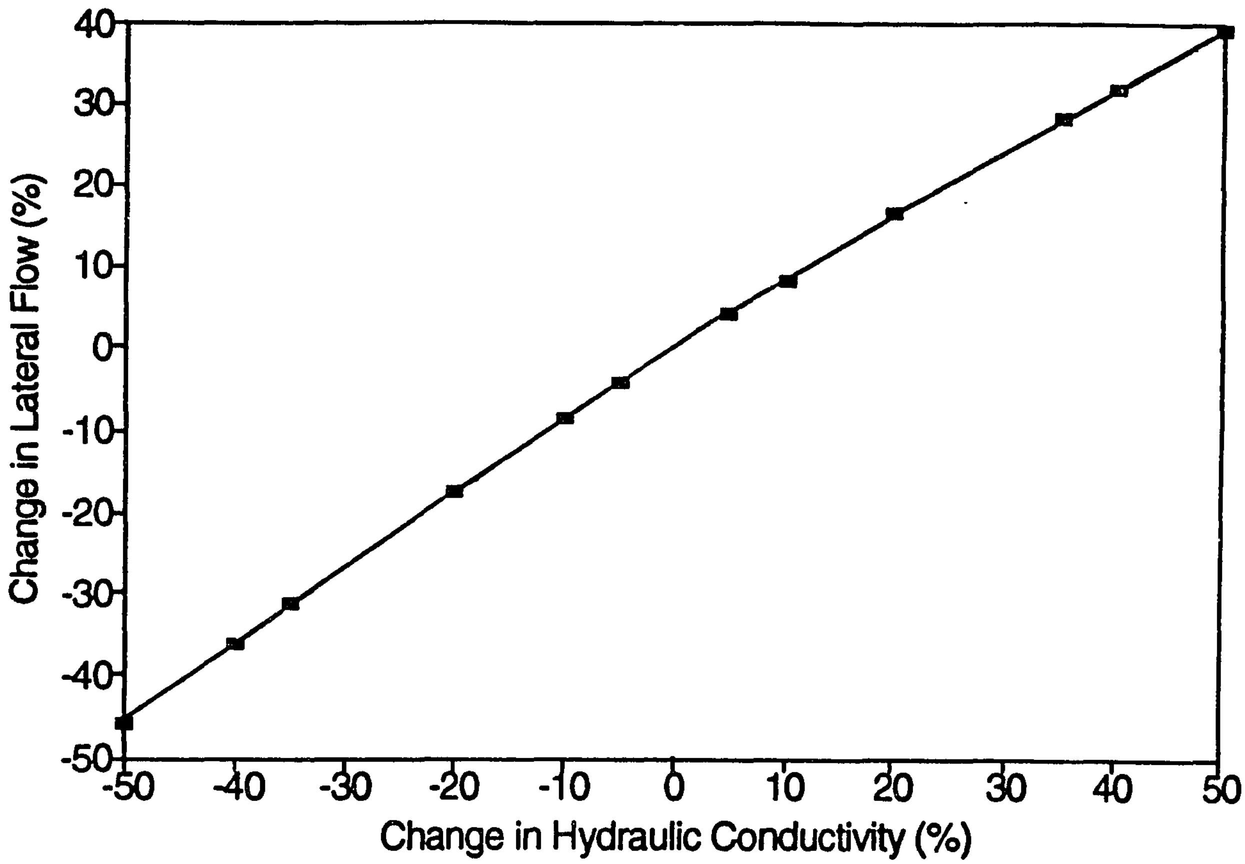


Figure 6-2. Sensitivity of Hydraulic Conductivity to Average Daily Lateral Flow Computed by the FILL Model.

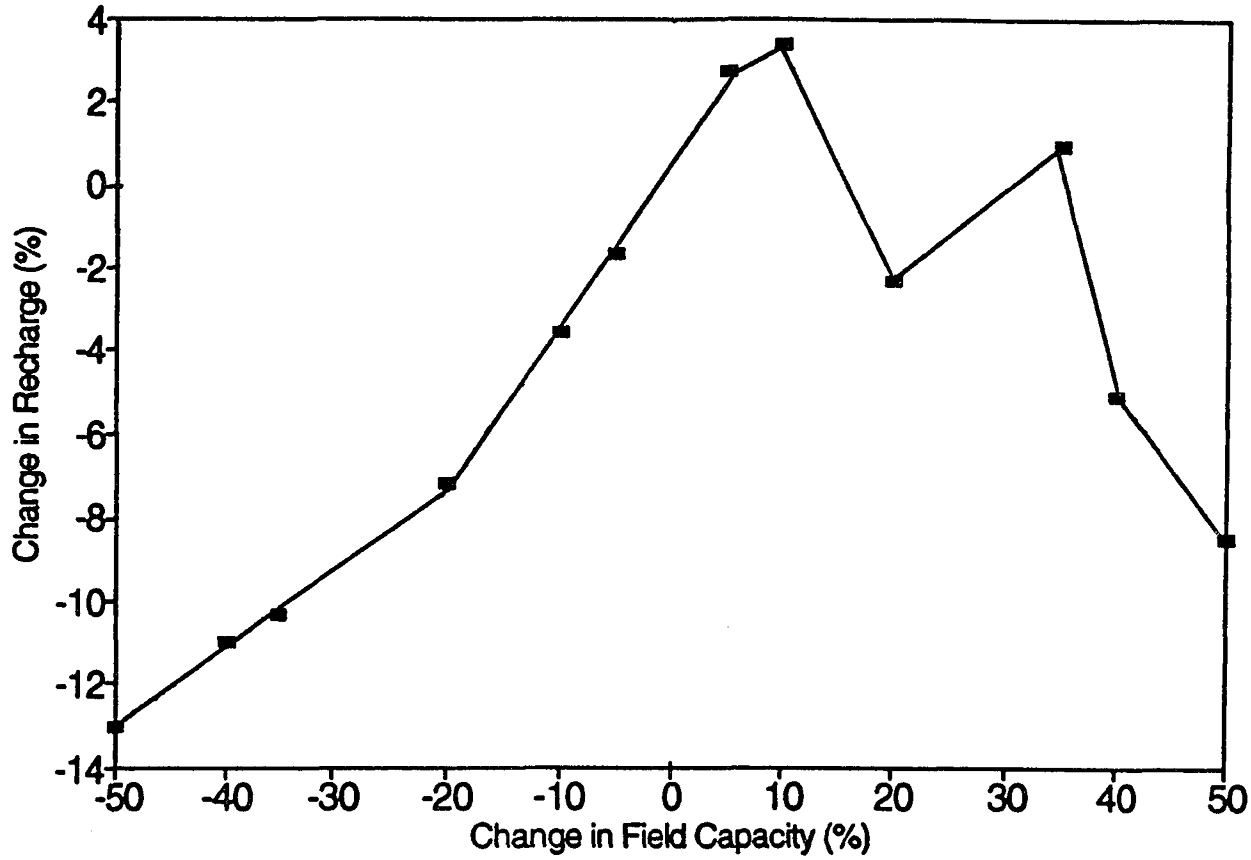


Figure 6-3. Sensitivity of Field Capacity to Annual Recharge Computed by the FILL Model.

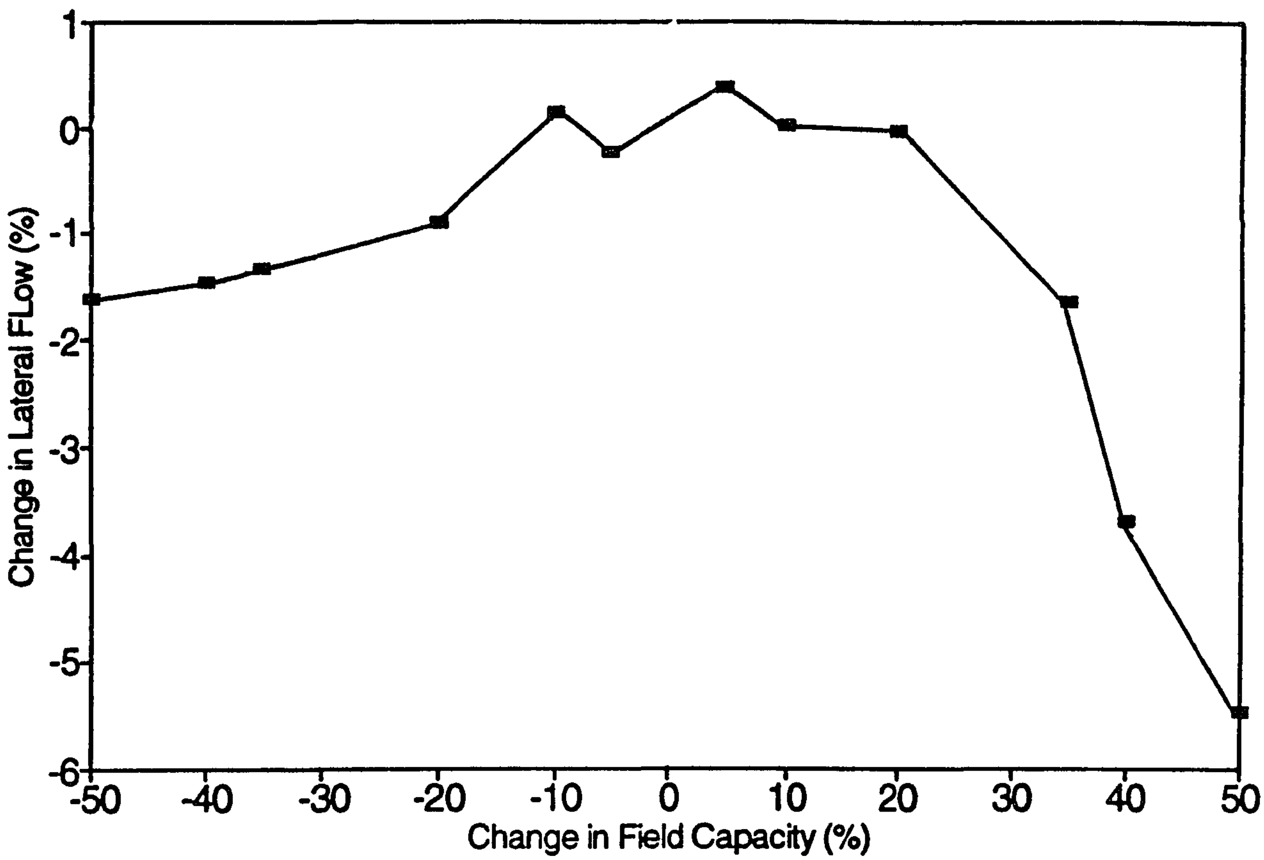


Figure 6-4. Sensitivity of Field Capacity to Average Daily Lateral Flow Computed by the FILL Model.

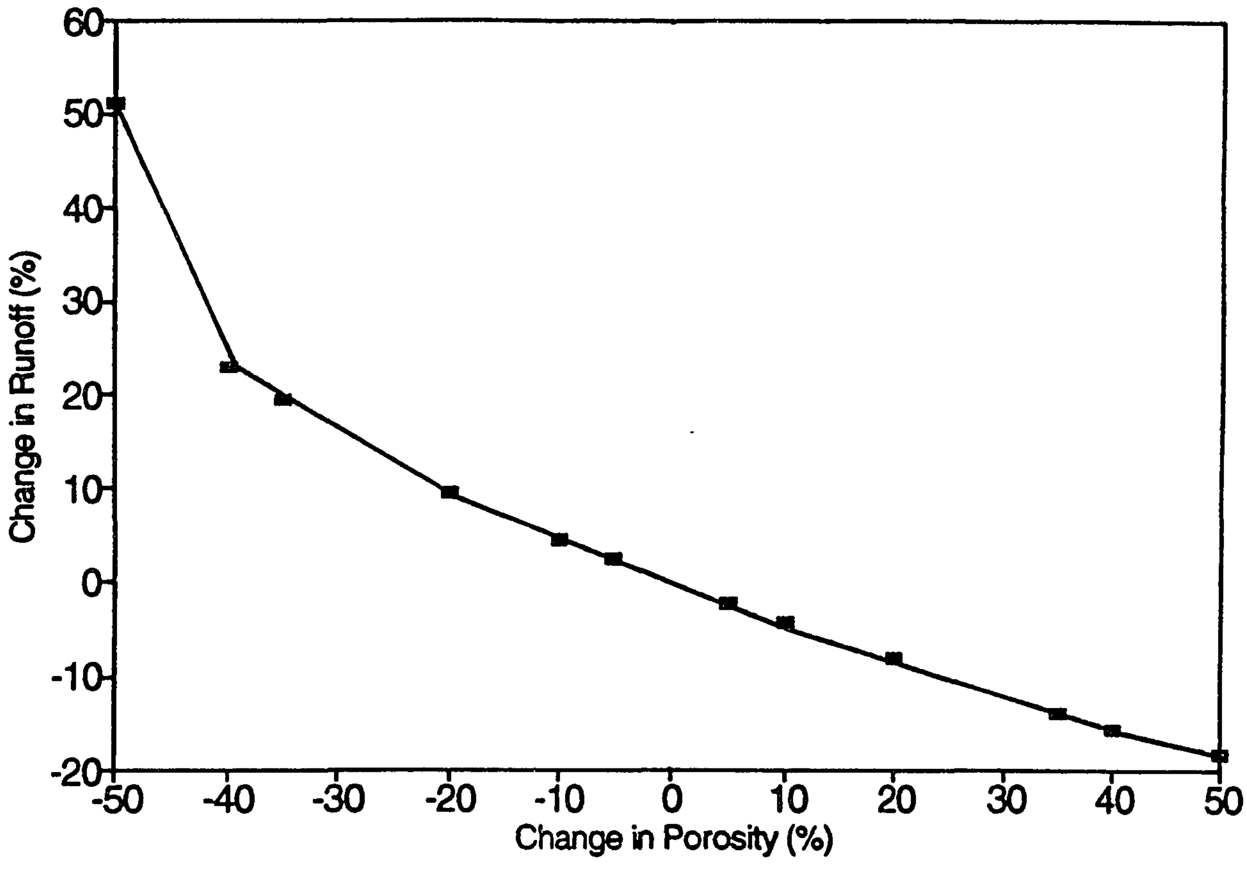


Figure 6-5. Sensitivity of Porosity to Surface Runoff Computed by the FILL Model.

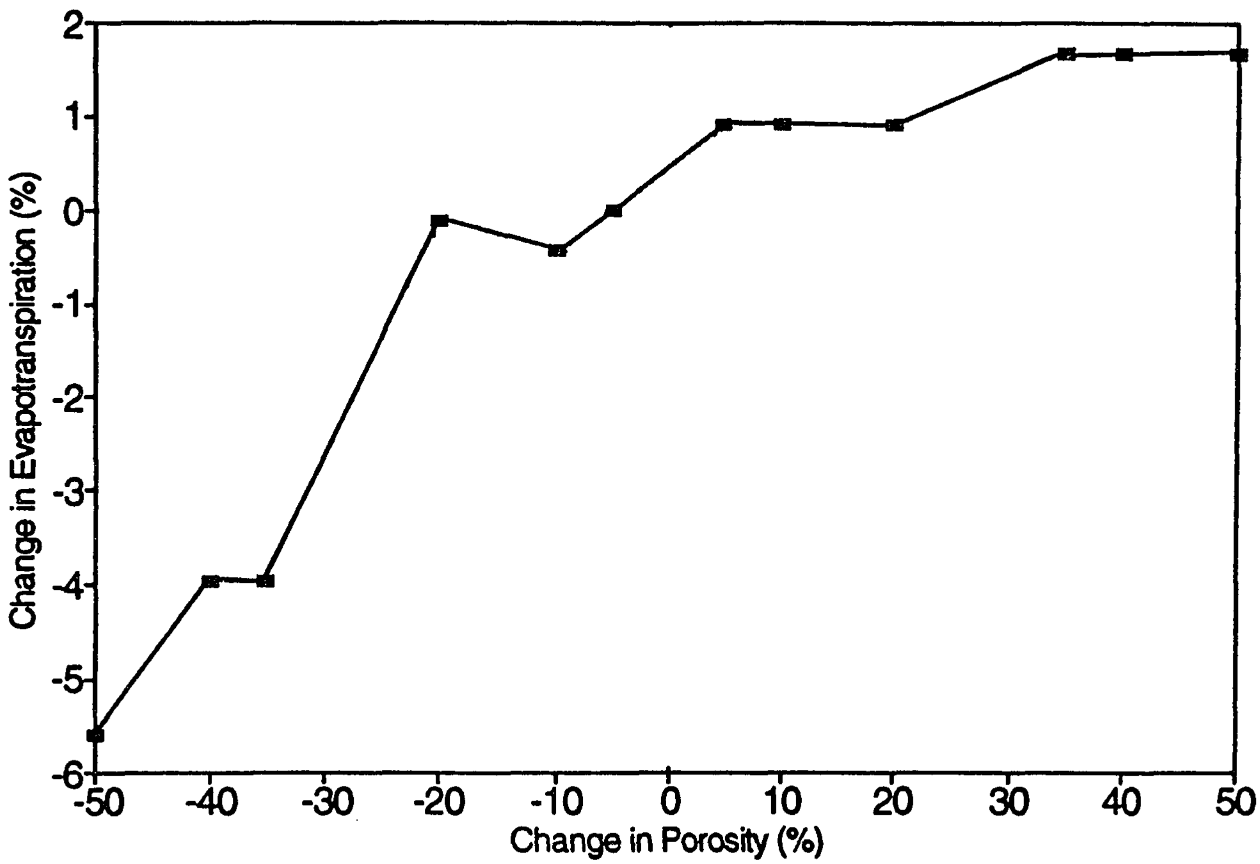


Figure 6-6. Sensitivity of Porosity to Evapotranspiration
Computed by the FILL Model.

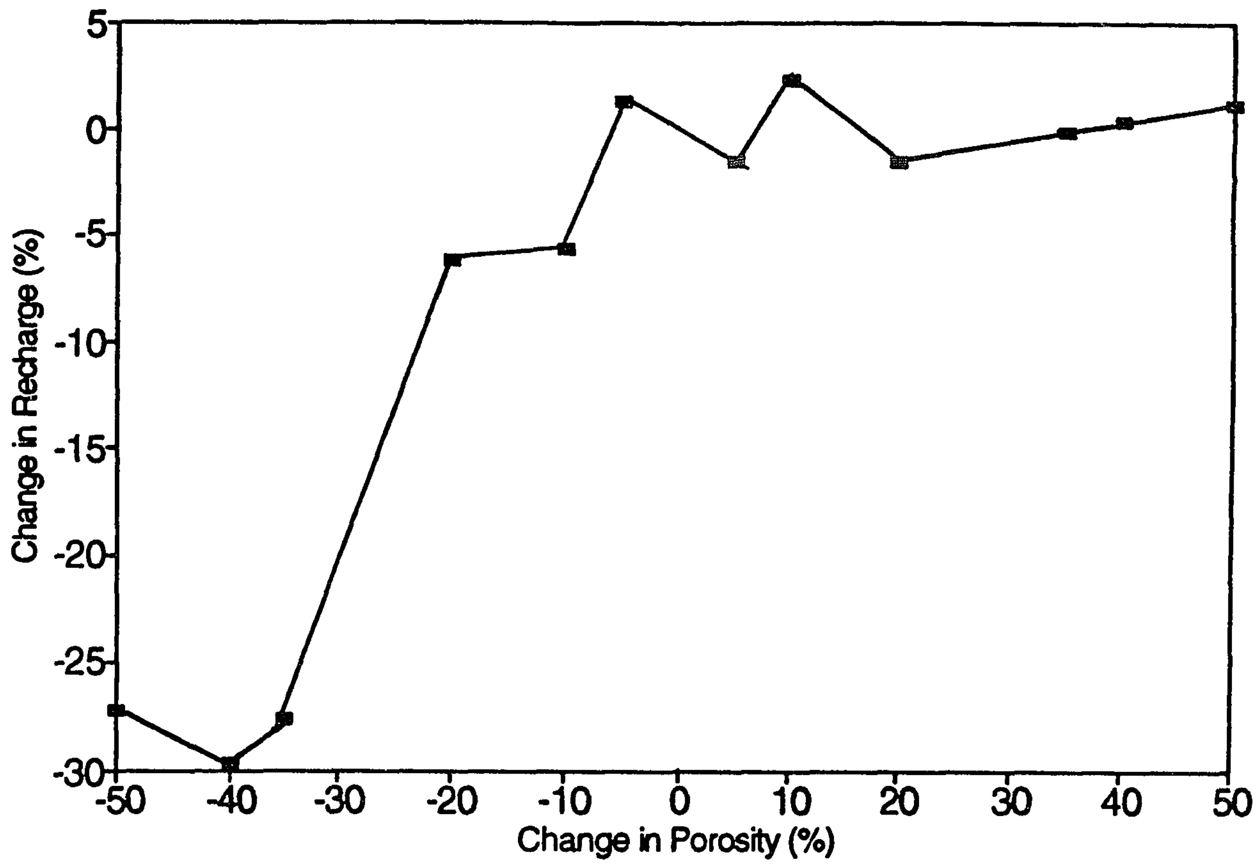


Figure 6-7. Sensitivity of Porosity to Annual Recharge Computed by the FILL Model.

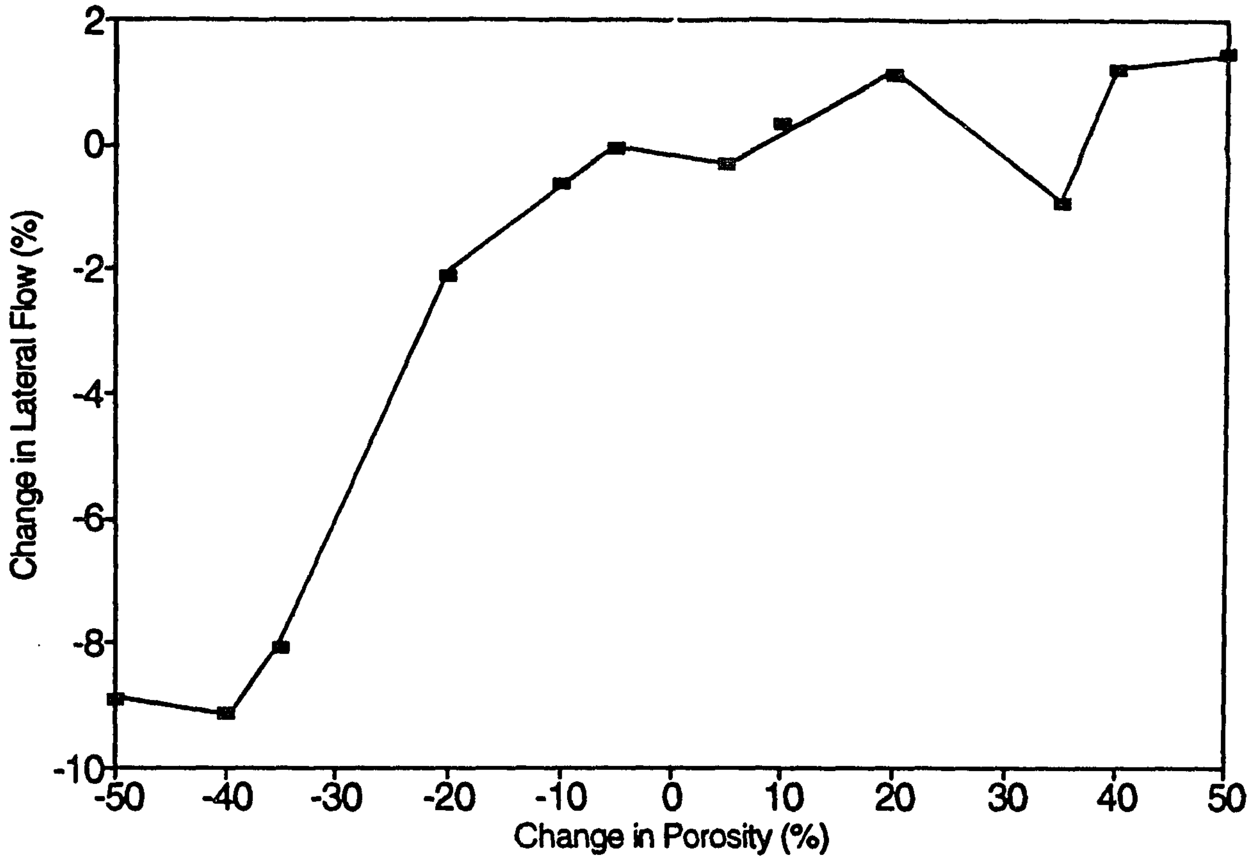


Figure 6-8. Sensitivity of Porosity to Lateral Flow Computed by the FILL Model.

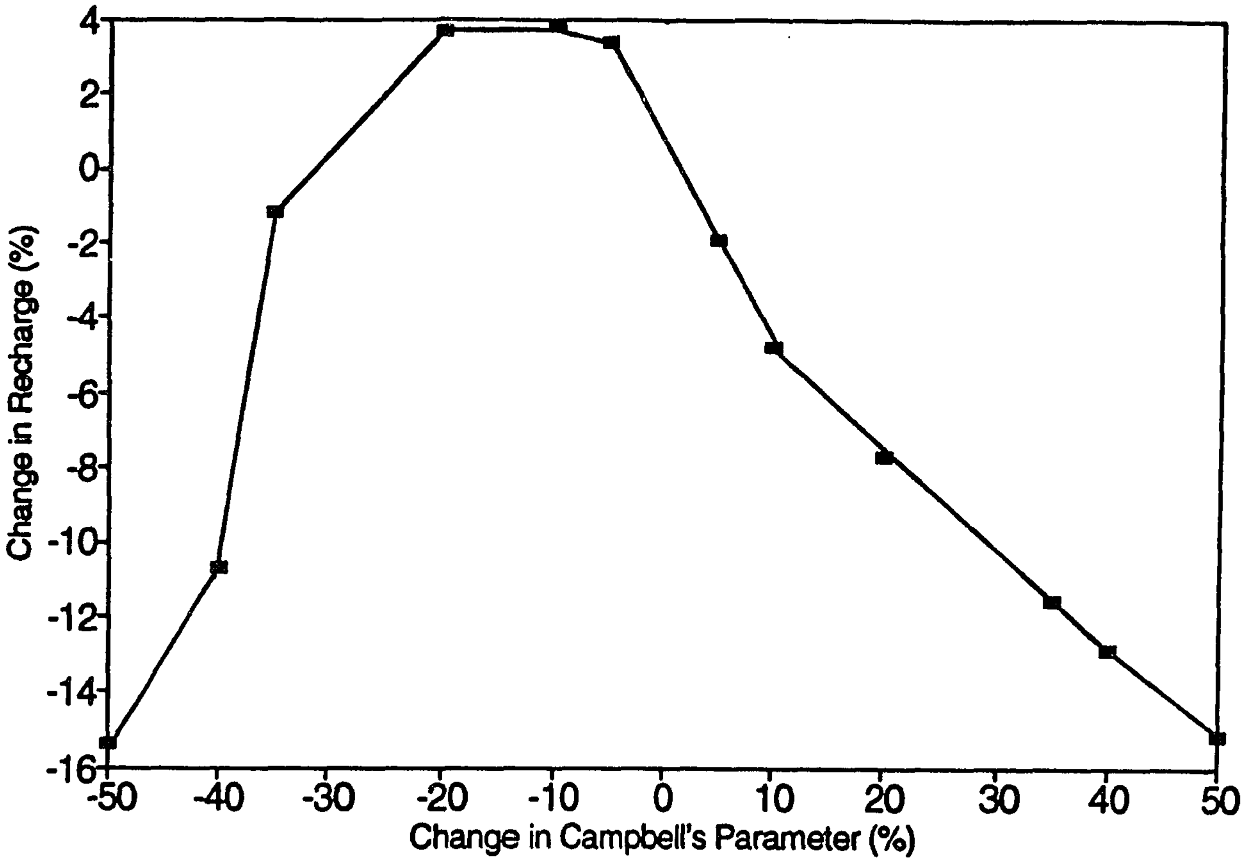


Figure 6-9. Sensitivity of Campbell's Parameter to Annual Recharge Computed by the FILL Model.

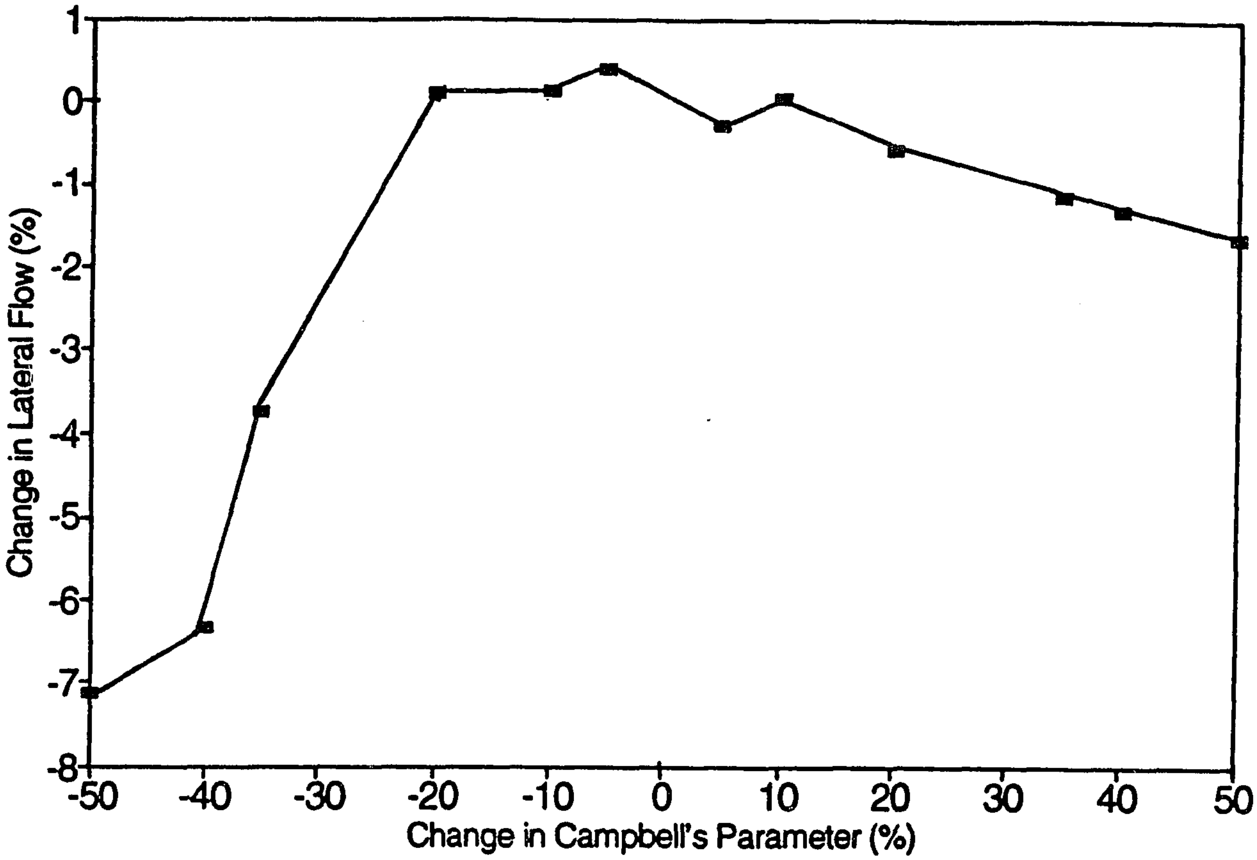


Figure 6-10. Sensitivity of Campbell's Parameter to Average Daily Lateral Flow Computed by the FILL Model.

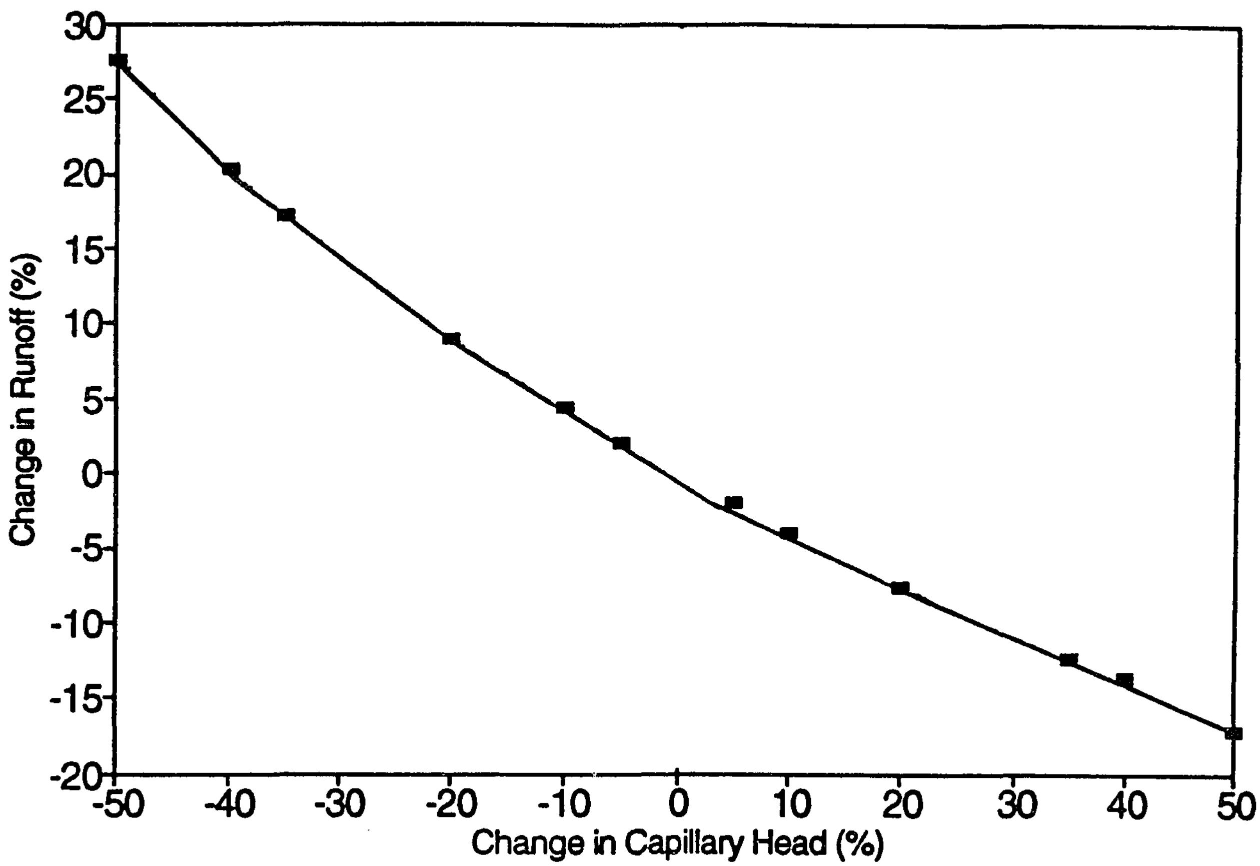


Figure 6-11. Sensitivity of Capillary Pressure Head to Surface Runoff Computed by the FILL Model.

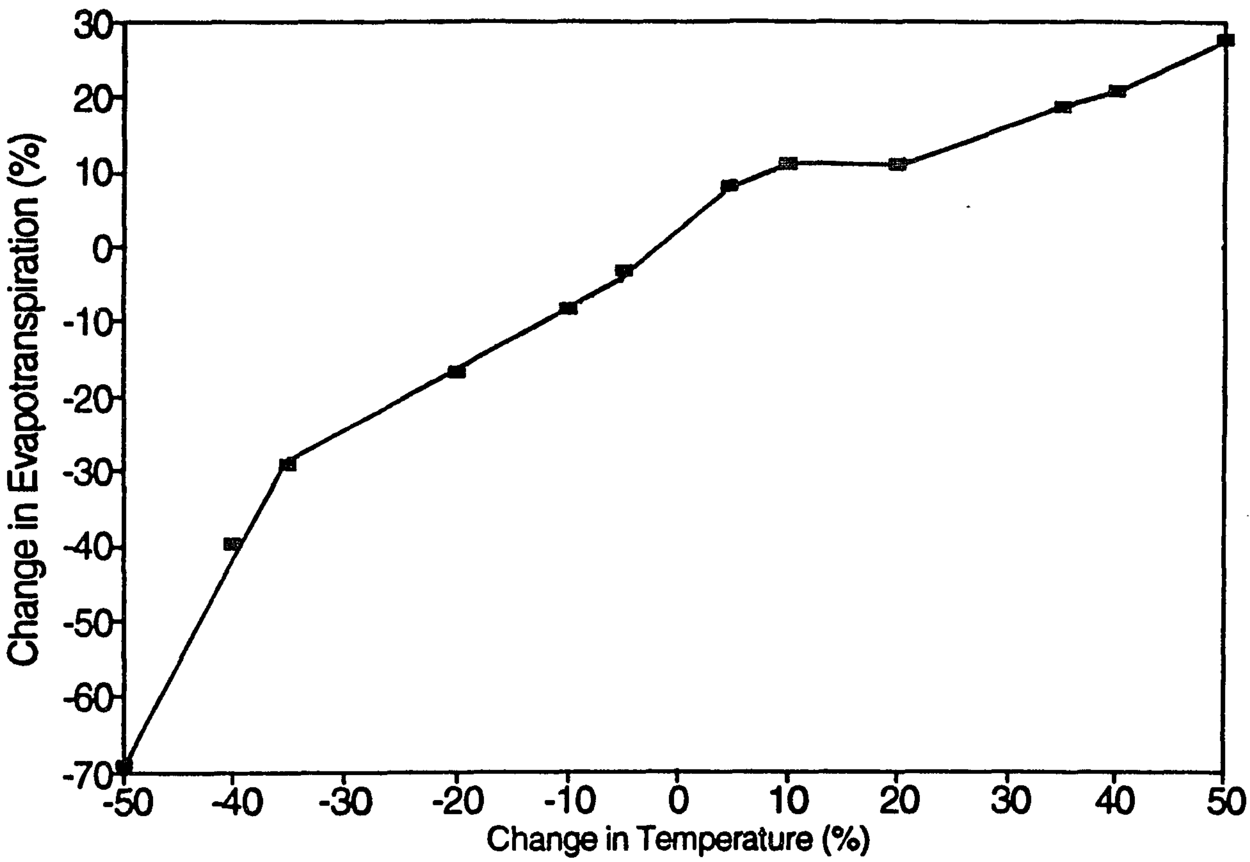


Figure 6-12. Sensitivity of Temperature to Evapotranspiration
Computed by the FILL Model.

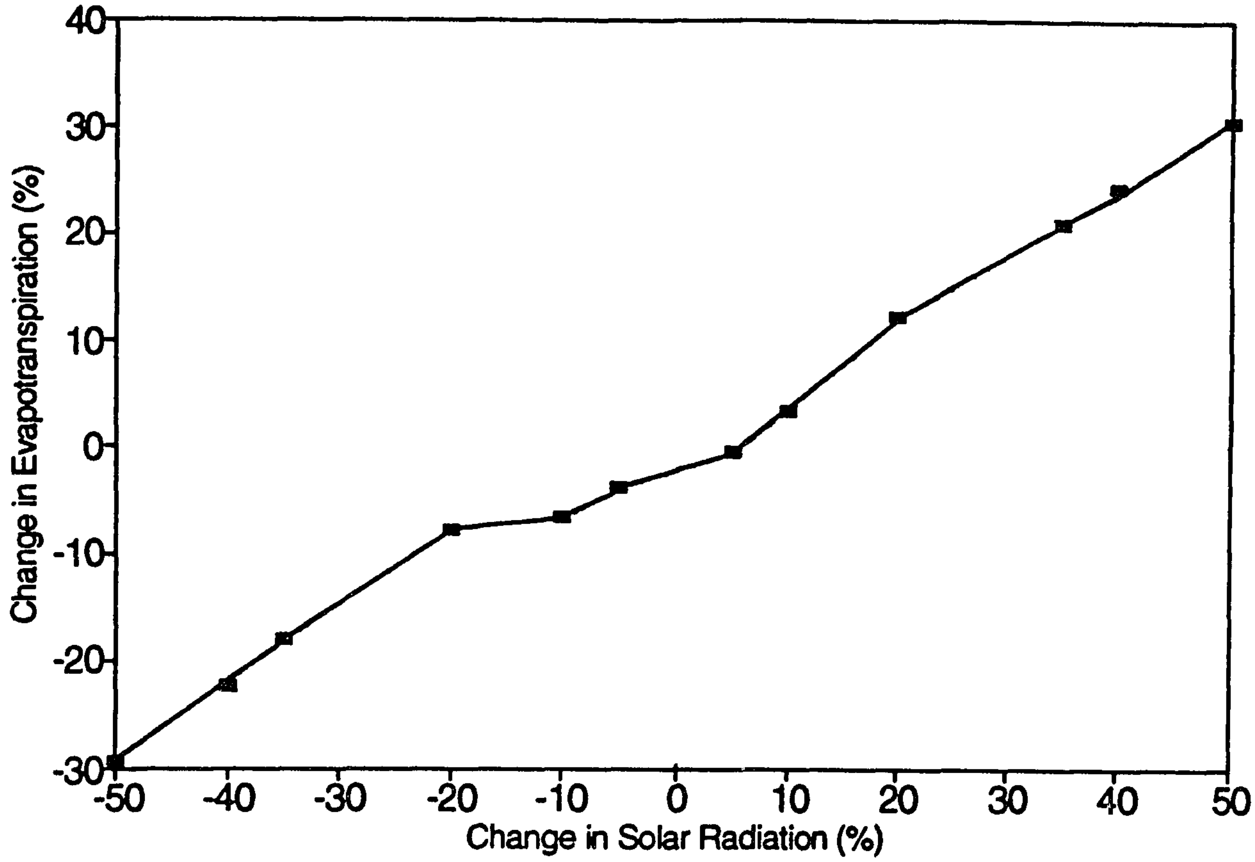


Figure 6-13. Sensitivity of Solar Radiation to Evapotranspiration Computed by the FILL Model.

landfill is a nonlinear function of moisture content. A 20 percent decrease in the value of field capacity causes a decrease of 7.13 percent in the annual recharge to the saturated leachate mound as shown in Figure 6-3. The effect of field capacity upon lateral flow is insignificant as shown in Figure 6-4. For a 50 percent increase in field capacity, the lateral flow decreases by 5.46 percent as shown in Figure 6-4.

Porosity defines the total volume of pore space in the soil matrix that can be filled up for complete saturation. The infiltration is a function of the porosity of the top soil. The rainfall excess is the amount of precipitation less infiltration that produces surface runoff. Porosity has significant effect upon surface runoff. Decrease in the value of porosity by 50 percent causes an increase of 51.17 percent in the annual surface runoff as shown in Figure 6-5. Evapotranspiration is not very sensitive to porosity. For a 50 percent increase in the value of porosity, evapotranspiration increases by 1.68 percent as shown in Figure 6-6. Porosity is also used to define the hydraulic conductivity and diffusivity in the unsaturated zone and these properties are used to compute recharge to the saturated leachate mound. The recharge is observed to be very sensitive for decrease in the value of the porosity. Decrease in the value of the porosity by 50 percent causes a decrease of 27.16 percent in the annual recharge as shown in Figure 6-7. Porosity is also sensitive to the lateral flow computed by the model. For a 50 percent decrease in the value of the porosity causes 8.9 percent decrease in the lateral flow as shown in Figure 6-8.

Campbell's parameter (b) is used to determine the hydraulic conductivity and diffusivity of the unsaturated porous media. Campbell's parameter contribute to the nonlinearity of unsaturated hydraulic conductivity and diffusivity functions. The change in recharge to the saturated leachate mound and lateral flow due to the change in Campbell's parameter is shown in Figures 6-9 and 6-10 respectively. For a 50 percent increase in the value of the Campbell's parameter (b), the annual recharge decreases by 15.09 percent as shown in Figure 6-9. The sensitivity of Campbell's parameter (b) to the lateral flow is shown in Figure 6-10. For a 50 percent decrease in the value of the Campbell's parameter (b), the lateral flow decreases by 7.12 percent as shown in Figure 6-10. The capillary pressure head is found to be sensitive to compute surface runoff using kinematic wave equation. An increase of 27.59 percent in runoff occurs due to a 50 percent decrease in the capillary pressure head as shown in Figure 6-11.

Temperature and solar radiation values are also found to be sensitive to compute evapotranspiration using modified Penman method as shown in Figures 6-12 and 6-13. For a 50 percent decrease in the daily values of temperature causes a decrease of 68.94 percent in the total evapotranspiration computed by the FILL model. Solar radiation is less sensitive to the evapotranspiration computed by the FILL model compared to the sensitivity shown by the variation in temperature. A decrease of 29.32 percent in the total evapotranspiration occur due to a 50 percent decrease in the daily values of solar radiation.

Influence of Manning's roughness coefficient used to compute the surface runoff has insignificant effect on the model output. Wilting point is used to compute evapotranspiration by modified Penman method. It has also insignificant effect on the computation of evapotranspiration for significant changes in its values.

6.2 MODEL APPLICATION

To demonstrate the effectiveness of the model to predict seasonal variation of leachate flow rates and to estimate the average leachate flow rates, the model was applied to an example landfill section as shown in Figure 6-14. The landfill has a surface area of 376 acres. The characteristics of the landfill matrix (porosity, field capacity, wilting point, and hydraulic conductivity) and local climatological data are used to estimate the leachate flow rates in this landfill section.

The climatological data (temperature and solar radiation) were used to compute evapotranspiration and snowmelt. These data were obtained from the default values of the HELP model. Values of leaf-area index and winter cover factor were also obtained from the default values of the HELP model.

Rainfall data for seven years (1949 to 1955) were obtained from New York City department of Environmental Protection and analyzed to obtain the average rainfall year. The average annual rainfall was obtained; and the annual rainfall for the year

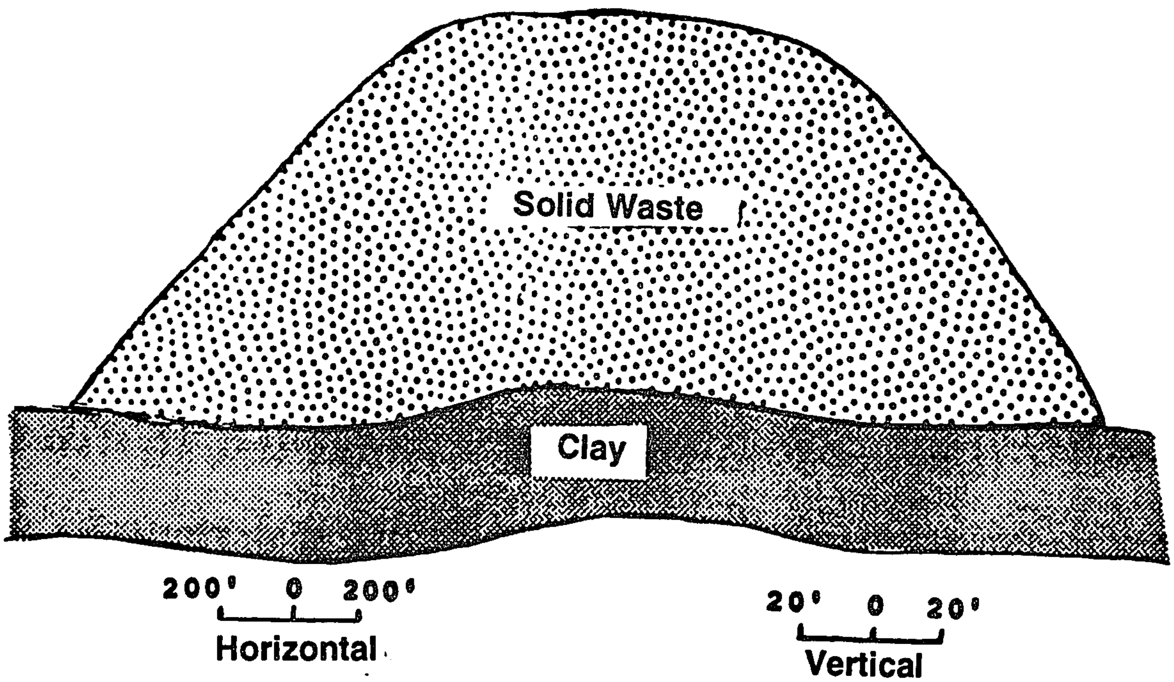


Figure 6-14. A landfill Section for FILL Model Application.

closest to the average annual value was selected as the average rainfall-year. The model was run to determine the variation of leachate flow for an average rainfall condition in the example landfill section.

The hydraulic conductivity value is obtained from the recovery test conducted by Wehran (1983) for the solid wastes at Fresh Kills landfill. The Fresh Kills landfill is located on the western shore of Staten Island, New York. The 3000-acre facility, owned and operated by the City of New York, has been receiving household wastes for disposal since 1947. The landfill has been developed into four distinct mounds, which correspond to the areas designated as Sections 1/9, 2/8, 3/4, and 6/7. The hydraulic conductivity for solid waste in Section 6/7 of Fresh Kills landfill was 0.02 cm/sec using the recovery test conducted by Wehran (1983). This value of hydraulic conductivity is used in the model to compute moisture content and leachate flow rates. A six-inch daily cover is considered on top the solid wastes with an average hydraulic conductivity value of 10^{-4} cm/sec. The model was run by considering the field capacity as the initial condition. The information to run the model for the example section is summarized in Table 6-1. The vertical landfill section was schematized into a network of finite-difference nodal areas as shown in Figure 6-15. There are a total of 64 grid-points.

The monthly variation of precipitation, runoff, evapotranspiration, and net infiltration in the landfill area is shown in Figure 6-16. The monthly values of precipitation for the average rainfall year, shown in Figure 6-16 are computed from the hourly

TABLE 6-1

DATA REQUIRED FOR THE FILL MODEL

Area	376 acres
Hydraulic conductivity:	
Daily cover	10^{-4} cm/sec (0.2835 ft./day)
Waste inside the landfill	0.02 cm/sec
Bottom clay	10^{-7} cm/sec
Capillary pressure head	4.0 inches
Porosity of the waste	0.417
Field capacity of the waste	0.05
Wilting point of the waste	0.02
Campbell's Parameter (b)	1.54

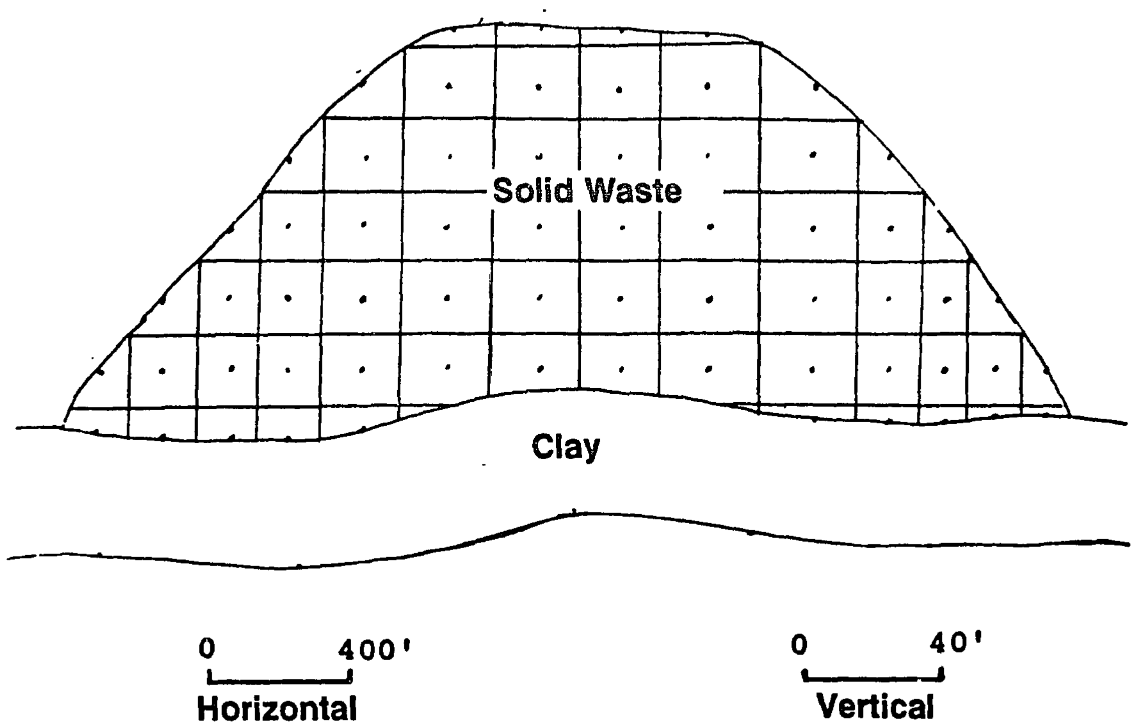


Figure 6-15. Computational Domain for FILL model.

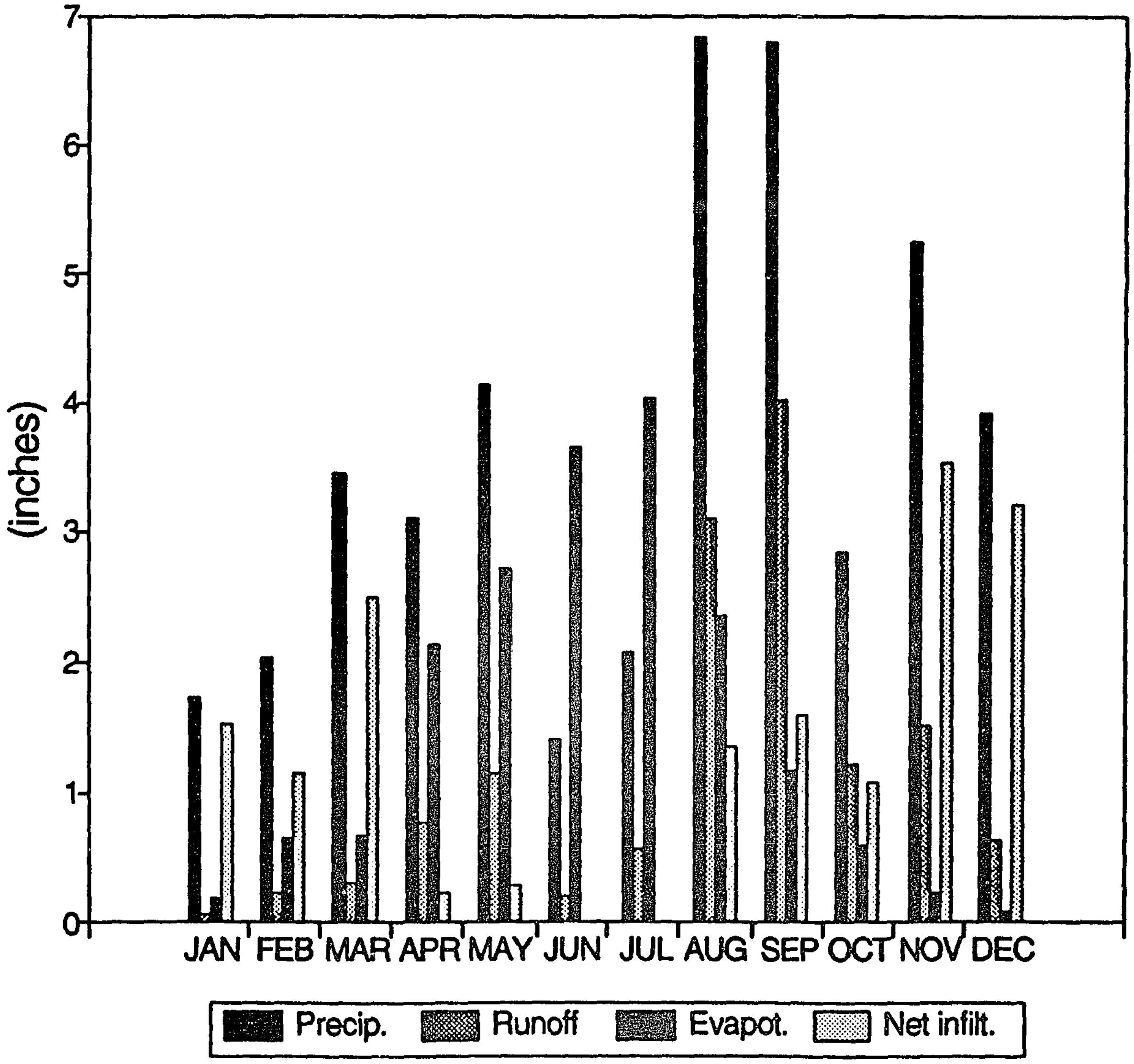


Figure 6-16. Variation of Precipitation, Runoff, Evapotranspiration, and Net Infiltration for an Average Rainfall Year.

rainfall values collected from New York City Department of Sanitation. Based on the hourly values, the runoff submodel runs for 24 hours a day when the temperature is above freezing (32°F) and when there is rainfall and/or snowmelt on the landfill surface. Figure 6-16 shows that a peak runoff value of 4.01 inches occurs during September when the rainfall is 6.78 inches. The evapotranspiration submodel computes the daily values of evaporation and transpiration using the modified Penman method described by Schroeder et al. (1984). In this estimation, evapotranspiration depends on surface runoff, infiltration, and moisture content of the soil. It is observed that the higher evapotranspiration occurs during July compared to the rest of the period in the average rainfall-year. The total annual surface runoff and evapotranspiration are computed at 13.7 inches and 18.45 inches respectively in the year when the rainfall was 43.6 inches. A higher amount of surface runoff is due to considering the side slope along the landfill surface.

The net infiltration is the precipitation less evapotranspiration and surface runoff. The variation of average daily values of net infiltration is shown in Figure 6-17. The total annual net infiltration of 11.45 inches occur, which is 26.26 percent of the total annual rainfall during the average rainfall-year. The recharge is the leachate accretion that reaches the saturated leachate mound. This is computed from the moisture content by solving the unsaturated leachate flow equation. The variation of average daily recharge is shown in Figure 6-18. The variation of recharge is consistent with the variation of net infiltration into the landfill as shown in Figures 6-17 and 6-18. Figure 6-17 shows that there is no net infiltration during June and

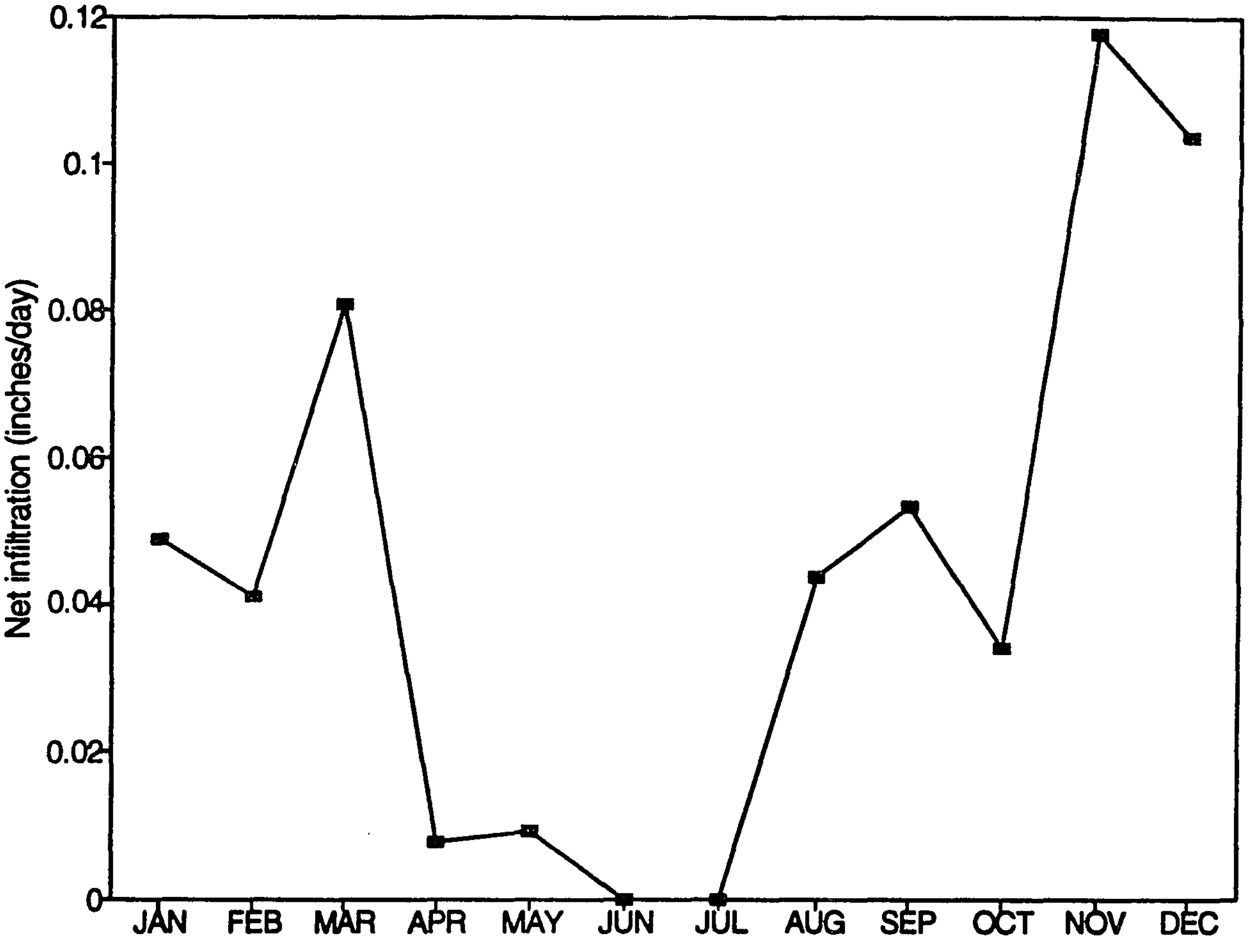


Figure 6-17. Variation of Daily Net Infiltration Computed by the FILL Model.

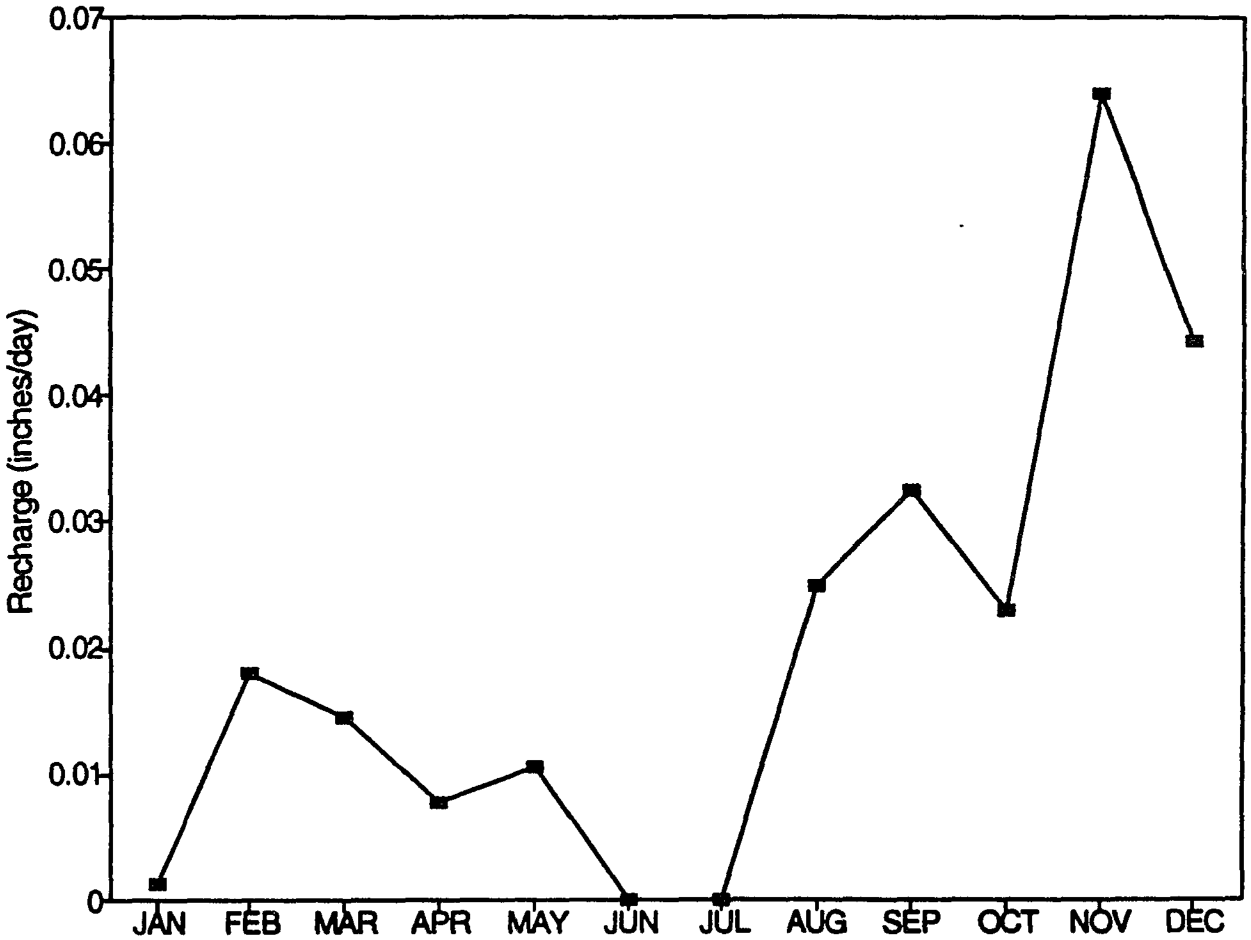


Figure 6-18. Variation of Daily Recharge Computed by the FILL Model.

July. During this time, the evapotranspiration remains higher than the total rainfall and no water is added to the retained moisture content inside the landfill. Due to the higher amount of evapotranspiration, the soil moisture content remains close to the field capacity and no recharge occurs as shown in Figure 6-17.

The outflows from the saturated leachate mound are the lateral flow towards the perimeter or the collection system and the vertical flow through the clay at the bottom of a landfill. The variation of average daily lateral flow and vertical flow is shown in Figures 6-19 and 6-20 respectively. An average lateral flow of 0.4233 mgd occurs during the average rainfall-year in the example landfill section. The vertical component of leachate flow is small compared to the lateral flow. The small value of daily vertical flow is due to the small hydraulic conductivity (10^{-7} cm.sec) of the bottom clay that restricts leakage through the bottom of the landfill. An average vertical flow of 0.0428 mgd is obtained from the model results.

The total daily flow is the sum of the lateral and vertical flow in mgd and the variation of total flow with time for the example landfill section is shown in Figure 6-21. The total average flow varies from 0.3728 mgd to 0.5977 mgd with an average daily flow of 0.466 mgd.

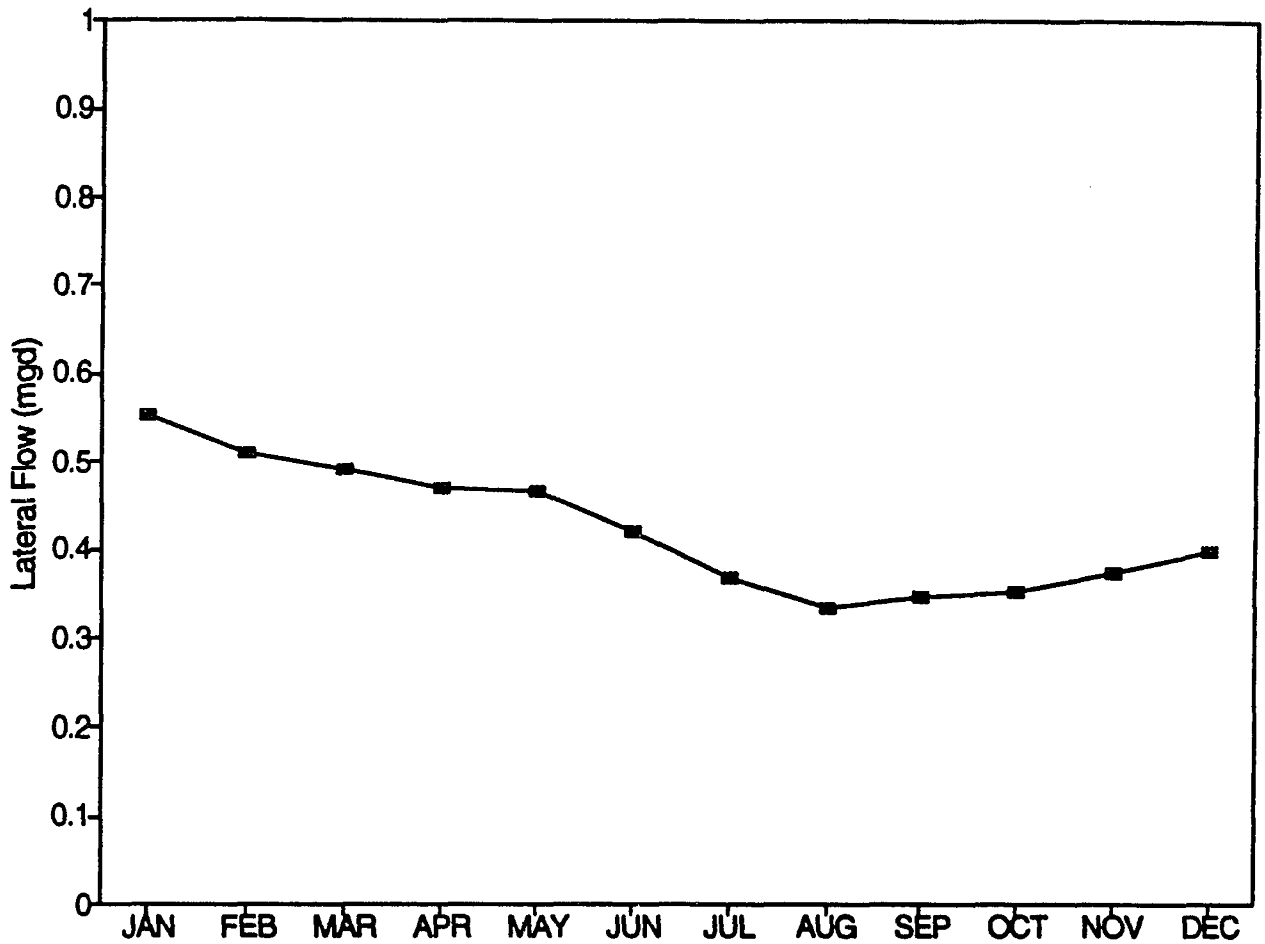


Figure 6-19. Variation of Average Daily Lateral Flow Computed by the FILL Model.

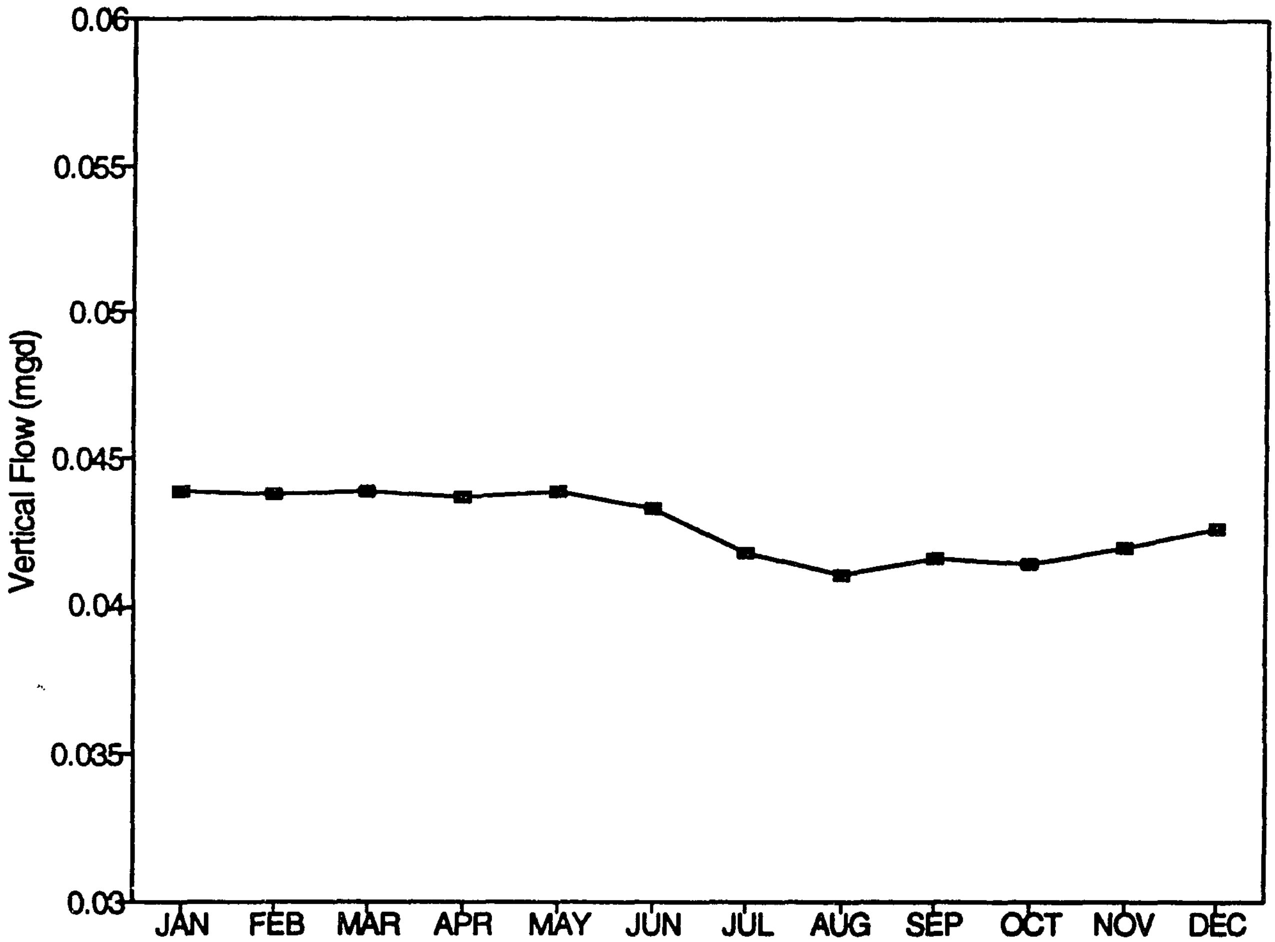


Figure 6-20. Variation of Average Daily Vertical Flow Computed by the FILL Model.

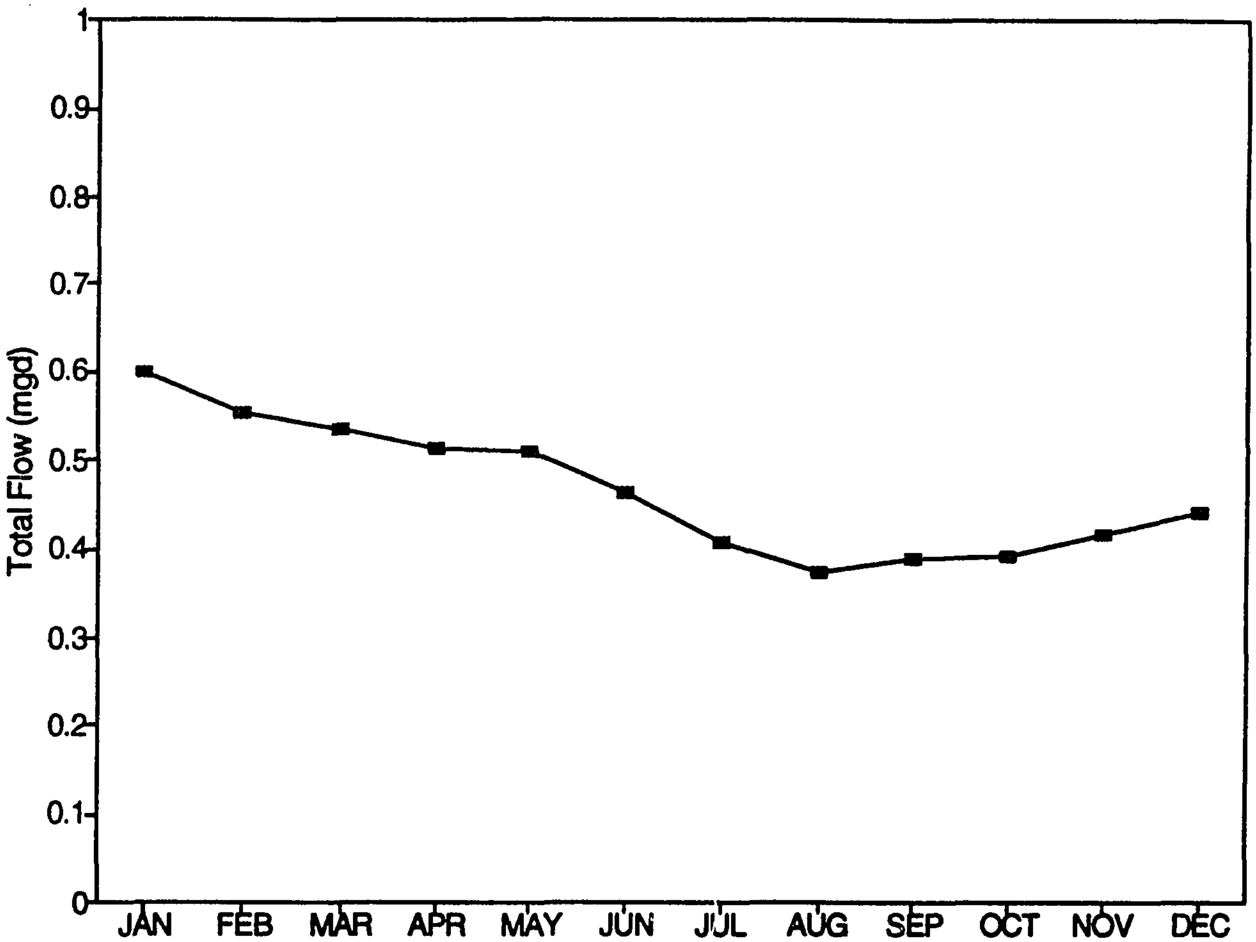


Figure 6-21. Variation of Average Daily Total Flow Computed by the FILL Model.

CHAPTER 7

DISCUSSION, CONCLUSIONS AND RECOMMENDATIONS

7.1 DISCUSSION AND CONCLUSIONS

In this dissertation, the leachate flow process into, through, and out of a solid waste landfill have been investigated. The leachate flow problems are described in both the unsaturated and saturated zones of a landfill. The unsaturated zone constitute a major portion of the landfill matrix. On-site leachate accretion occurs primarily in the unsaturated zone of a landfill. Saturated leachate mound exists at the bottom of a landfill. Leachate mound is caused by the restriction of moisture flow by a natural impermeable clay layer or an artificial clay liner beneath the landfill.

The governing equations in both the unsaturated and saturated zones in a landfill are described by partial differential equations obtained by considering mass balance and Darcy's Law. The governing equations are expressed in terms of moisture content in the unsaturated zone and leachate mound-head in the saturated zone of a landfill.

The unsaturated leachate accretion is described by a two-dimensional unsteady state moisture-flow equation in a vertical domain. The unsteady state flow condition is considered to achieve realistic results for the time-varying field conditions. The properties of the medium, such as unsaturated hydraulic conductivity and diffusivity change with time in the unsaturated zone due to change in the moisture content. Also, the boundary condition (boundary condition depends on precipitation, evapotranspiration, and surface runoff) gives a time-varying net infiltration into the landfill that produces an unsteady leachate accretion. The saturated leachate mound-head variation in the two-dimensional vertical domain is formulated by considering leachate accretion from the unsaturated zone and flow due to hydraulic gradient and landfill bottom slope. The FILL model incorporates the leachate flow process from the upper boundary through the unsaturated and saturated zones by solving the governing equations using the implicit finite-difference method.

Leachate flow modeling requires accurate solution of the governing partial differential equations with appropriate boundary and initial conditions. To achieve that, numerical solution techniques are investigated in this work. The finite-difference solutions for both the unsaturated and saturated zones of a landfill are compared with the exact solutions. To gain more confidence, two numerical solution techniques, Gauss-Seidel iteration and Gauss-Jordan elimination methods are used and tested for accuracy by comparing with the exact solutions.

High-quality numerical modeling also depends on adopting efficient and suitable solution scheme. The most common numerical schemes used to solve the unsaturated leachate flow equation are developed using finite-difference and finite element methods. To study the accuracy and suitability of numerical schemes in providing information in the unsaturated zone of a landfill, two boundary integral approaches, direct Green's function and perturbation Green's function, are used and compared with the finite-difference technique.

Accurate analysis was carried out by first developing analytical solutions for the governing equations. Some simplifying assumptions are made and the method of separation of variables is used to derive the analytical solution for the unsaturated and saturated leachate flow equation. Both Gauss-Seidel iteration and Gauss-Jordan elimination methods give agreeable solution compared to the analytical solution for the unsaturated moisture-flow equation. The difference in solution using Gauss-Seidel iteration and Gauss-Jordan elimination methods is insignificant. The close agreement of the solution by both iteration and elimination methods establishes the credibility of the finite-difference model used for moisture-flow equation. The solution of the finite-difference scheme for saturated leachate flow equation is obtained using Gauss-Seidel iteration and Gauss-Jordan elimination methods. These two methods of finite-difference solutions are also tested by comparing with the exact solution and close agreement is observed.

Extension of the boundary integral method to solve the unsaturated moisture-flow equation gives satisfactory success compared to the finite-difference method in providing information such as the moisture content and the moisture flux. The use of time-dependent Green's function helped solve the unsteady unsaturated moisture-flow equation in both space and time. The two boundary integral approaches, direct Green's function and perturbation Green's function solutions show close agreement to the exact solution in comparison with the finite-difference solution.

In the direct Green's function formulation, the gravity term $\partial K(\theta)/\partial z$ is computed by considering the moisture content from the previous time step. In the perturbation Green's function formulation, the gravity term is perturbed and it is obtained from the preceding level of solution in the perturbation series. The perturbation series solution helps represent the physical system realistically by evaluating the gravity term at the current time level. But the perturbation series solution is done at the expense of more computing time than the direct Green's function solution.

Comparison of the finite-difference and the boundary integral approaches are done by computing the differences in the solution from the analytical solution. The absolute errors in boundary integral solutions remain less compared to the finite-difference method of solution. The maximum absolute deviation from the exact solution is higher in the finite-difference solution by both the Gauss-Seidel iteration and Gauss-Jordan elimination methods than that in the boundary integral solution. An average absolute error of 0.01523 is obtained in Gauss-Seidel iteration method of finite-difference solution. The results obtained by the Gauss-Jordan elimination method is similar to the result obtained by the Gauss-Seidel iteration method in the finite-difference solution. An average absolute error of 0.01522 is observed in the Gauss-Jordan elimination method of finite-difference solution. In the direct Green's function solution, an average absolute error of 0.01336 is obtained. An average absolute error of 0.01165 is obtained in the perturbation Green's function solution. The perturbation Green's function solution is found to give more satisfactory results than the rest of the methods used in this study.

The two-dimensional boundary integral solutions are also suitable for simulating the one-dimensional moisture-flow problems. The close agreement of the variation of the moisture contents predicted by the one-dimensional analytical and the two-dimensional boundary integral solutions shows the capability of the boundary integral techniques to solve the moisture-flow problems for a soil column where flow is occurring predominantly in the vertical direction.

The boundary integral solution is capable of generating moisture fluxes in the unsaturated flow-domain of a landfill. The internal fluxes are compared with the analytical solution and show close agreement. The internal fluxes are used to determine the leachate accretion to the saturated leachate mound. The ability of the boundary integral method to directly predict the internal fluxes is an advantage that is useful to know for solving contaminant transport problem in the unsaturated zone.

In the finite-difference method, only the solution of the dependent variable is obtained at discrete points in the domain. The internal velocities can not be predicted by directly differentiating the function because the derivative is approximated in the finite-difference grid. However, in the boundary integral method, the function is obtained as a spatially continuous solution and the formulation of the dependent variable allows to compute the velocity by directly differentiating the function. The internal fluxes are compared with the exact solutions and close results are observed. The close agreement of the internal fluxes computed by the boundary integral method and the analytical solution is observed. The close agreement shows the ability of the boundary integral method to estimate the leachate accretion and recharge to the saturated leachate mound using the internal moisture-flux in the domain.

The performance of the finite-difference scheme for saturated leachate flow equation is investigated by comparing with the analytical solution. To establish credibility in the performance of the finite-difference scheme for saturated leachate flow equation, two numerical solution techniques, Gauss-Seidel iteration and Gauss-Jordan elimination are used and the results are compared with the analytical solution. It is observed that the numerical and exact solutions are in close agreement. The numerical solution by Gauss-Seidel iteration and Gauss-Jordan elimination methods show identical results. Gauss-Seidel iteration method requires iteration and it is observed that the number of iteration increases in the main model for large leachate accretion from the unsaturated zone. To avoid time consuming iteration process, Gauss-Jordan elimination method is found suitable to be used in the FILL model.

Sensitivity analysis was done to examine the influence of the input parameters on the solution obtained by the model. The sensitivity analysis was done in an example computational landfill domain. Extremely small variations are observed for field capacity and wilting point to compute evapotranspiration. Porosity is observed to have a significant effect on surface runoff and annual recharge but an insignificant effect on evapotranspiration. The recharge to the saturated leachate mound depends on the medium properties such as diffusivity and unsaturated hydraulic conductivity. Both of the properties are functions of porosity. Hydraulic conductivity is also a sensitive parameter. A 50 percent decrease in its value causes a decrease of 45.61 percent in the lateral flow. Recharge also increases with the increase in hydraulic conductivity. A 4.25 percent increase in annual recharge is observed due to 50 percent increase in the hydraulic conductivity value.

The FILL model was applied to an example landfill section to illustrate its effectiveness in providing information of flow components, such as surface runoff, evapotranspiration, infiltration, and leachate flow rates in both the lateral and vertical directions. Surface runoff is computed based on the hourly rainfall data. A significant amount of surface runoff occurs from the landfill surface considering side slope and less permeable daily soil cover (hydraulic conductivity 10^{-4} cm/sec) on the landfill surface. An annual amount of 13.7 inches of surface runoff occurs from the example landfill section for an annual precipitation of 43.6 inches.

The evapotranspiration submodel computes the daily values of evaporation and transpiration using the modified Penman method. The evapotranspiration depends on infiltration and moisture content of the top soil. The modified Penman method also considers the effect of climatological data such as temperature and solar radiation. The soil data such as field capacity, wilting point, and porosity are also used to compute the actual evapotranspiration using the modified Penman method. A total amount of 18.45 inches of evapotranspiration is obtained for the example landfill area.

The recharge is the amount of leachate that is added to the saturated leachate mound from the unsaturated zone in a landfill. The main purpose of the solution for the unsaturated flow equation is to compute leachate accretion generating recharge to the saturated leachate mound. The variation of net infiltration and recharge are compared. The pattern of the variation of net infiltration into the landfill is similar to the pattern of the variation of recharge predicted for a period of one year. An annual amount of 7.29 inches of recharge to the leachate mound occurs in a period of one year for a total rainfall of 43.6 inches.

The FILL model provides the time-varying leachate flow rates which has the practical applications in the appropriate design for treatment facilities. The model provides time-varying drainage rate from the leachate mound called lateral flow and leakage through the bottom clay called vertical flow. An average daily lateral flow of 0.4233 mgd is obtained for the example landfill area. The vertical flow values are observed to vary only insignificantly with time. An average vertical flow of 0.0428 mgd is computed by the FILL model. The insignificant variation of the vertical flow is due to the lower value of the barrier hydraulic conductivity of 10^{-7} cm/sec in the landfill.

The finite-difference formulation for both unsaturated and saturated zones are evaluated by comparing with the analytical solution. Satisfactory performance and credibility of the model results are achieved for both the unsaturated and saturated flow problem for idealized flow conditions. In real field applications, the major uncertainties in making more accurate simulations are in the material properties such as the capillary pressure head and the coefficients related to the soil moisture diffusivity $D(\theta)$ and hydraulic conductivity $K(\theta)$. These uncertainties can only be resolved by experimental studies in landfill refuse materials.

7.2 RECOMMENDATIONS FOR FURTHER STUDY

The present single barrier layer configuration of the FILL model may be extended to include operational barrier layers considering both the unsaturated and saturated flows. The model can also be modified to consider simultaneously the final cover hydraulics, unsaturated-saturated flow through the intermediate barrier layers and the leachate flow at the bottom of a landfill. The same governing equations for both the unsaturated and saturated flows can be solved for the zones above the intermediate barrier layers. Such a model will be more useful for practical applications in providing simultaneously the information of the landfill final cover efficiency, leachate accretion above the intermediate barrier layers, leakage through all of the barrier layers, and leachate flow rates at the bottom of a landfill.

The present boundary integral formulations have been employed to solve unsteady leachate flow in the unsaturated zone of a landfill. In the saturated zone, the same boundary integral approaches can be employed to compute the leachate mound-head. To solve the unsaturated-saturated flow problem simultaneously, the possibility is to use zone concept with a moving interface between the unsaturated and saturated zone of a landfill. The development of such a model using boundary integral method is meaningful to realistically represent the physical system describing simultaneously the unsaturated and the saturated leachate flow phenomena in a landfill.

The finite-difference and boundary element models for the unsaturated flow in a landfill were compared with the analytical solutions. Experimental investigations can also be performed by constructing landfill cells to generate data of surface runoff, evapotranspiration, moisture content, and leachate accretion. Model results can be compared with the experimental data for further verification of the numerical models.

APPENDIX A

DERIVATION OF ANALYTICAL SOLUTION FOR THE UNSATURATED MOISTURE FLOW EQUATION

The simplified form of the moisture-flow equation in the unsaturated zone is:

$$\frac{\partial\theta}{\partial t} = \hat{D}\left(\frac{\partial^2\theta}{\partial x^2} + \frac{\partial^2\theta}{\partial z^2}\right) - p \qquad \text{(A-1)}$$

Here p defines the gravity $\partial K(\theta^k)/\partial z$ as expressed in equation (5-1).

The boundary conditions as illustrated in Figure 5-1 are as follows:

$$\theta(x,0,t) = \bar{A}$$

$$\theta(0,z,t) = \bar{B}$$

$$\frac{\partial\theta}{\partial x}(L_1,z,t) = \bar{C}$$

$$\frac{\partial\theta}{\partial z}(x,L_2,t) = \bar{D}$$

The initial condition is :

$$\theta(x,z,0) = \theta_o$$

The solution for θ(x,z,t) can be set in the following form:

$$\theta(x,z,t) = v(x,z,t) + \overline{A} + \frac{z^2}{2L_2}\overline{D} \qquad \text{(A-2)}$$

Here v(x,z,t) is another variable as a function of both space and time to be determined.

From equation (A-2):

$$\frac{\partial\theta}{\partial x} = \frac{\partial v}{\partial x}$$

$$\frac{\partial^2\theta}{\partial x^2} = \frac{\partial^2 v}{\partial x^2} \qquad \text{(A-3)}$$

$$\frac{\partial\theta}{\partial x}(L_1, z, t) = \frac{\partial v}{\partial x}(L_1, z, t) = \overline{C}$$

$$\frac{\partial\theta}{\partial z} = \frac{\partial v}{\partial z} + \frac{\overline{D}z}{L_2} = \overline{D}$$

$$\frac{\partial^2\theta}{\partial z^2} = \frac{\partial^2 v}{\partial z^2} + \frac{\overline{D}}{L_2} \qquad \text{(A-4)}$$

$$\frac{\partial\theta}{\partial z}(x, L_2, t) = \frac{\partial v}{\partial z}(x, L_2, t) + \overline{D} = \overline{D}$$

$$\frac{\partial v}{\partial z}(x, L_2, t) = 0$$

$$\frac{\partial \theta}{\partial t} = \frac{\partial v}{\partial t} \tag{A-5}$$

Substituting (A-3), (A-4) and (A-5) into (A-1):

$$\frac{\partial v}{\partial t} = \hat{D}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial z^2}\right) + \hat{p} \tag{A-6}$$

where,

$$\hat{p} = \frac{\hat{D}\overline{D}}{L_2} - p$$

The initial and boundary conditions for the dependent variable v(x,z,t) in equation (A-6) are as follows:

$$v(x, z, 0) = \hat{\theta} \tag{A-7a}$$

$$v(x, 0, t) = 0 \tag{A-7b}$$

$$v(0, z, t) = \hat{B} \tag{A-7c}$$

$$\frac{\partial v}{\partial x}(L_1, z, t) = \overline{C} \tag{A-7d}$$

$$\frac{\partial v}{\partial z}(x, L_2, t) = 0 \tag{A-7e}$$

where,

$$\hat{B} = \overline{B} - \overline{A} - \frac{z^2}{2L_2}\overline{D}$$

$$\boldsymbol{\theta} = \theta_o - \overline{A} - \frac{z^2}{2L_2}\overline{D}$$

The solution for v(x,z,t) can be written as:

$$v(x,z,t) = \sum_{n=0}^{\infty} F_n(x,t)\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\} \tag{A-8}$$

Here $F_n(x,t)$ is a coefficient that depends on horizontal dimension x and time t, which is to be determined.

From equation (A-8):

$$\frac{\partial v}{\partial t} = \sum_{n=0}^{\infty} \frac{\partial F_n(x,t)}{\partial t}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}$$

$$\frac{\partial v}{\partial x} = \sum_{n=0}^{\infty} \frac{\partial F_n(x,t)}{\partial x}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\} \tag{A-9}$$

$$\frac{\partial^2 v}{\partial x^2} = \sum_{n=0}^{\infty} \frac{\partial^2 F_n(x,t)}{\partial x^2}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\} \tag{A-10}$$

$$\frac{\partial v}{\partial z} = \sum_{n=0}^{\infty} F_n(x,t)\,\frac{(2n+1)\pi}{2L_2}\,\mathrm{Cos}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}$$

$$\frac{\partial^2 v}{\partial z^2} = -\sum_{n=0}^{\infty} F_n(x,t)\left\{\frac{(2n+1)\pi}{2L_2}\right\}^2 \mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\} \qquad \text{(A-11)}$$

The sine series expression for the nonhomogeneous term in equation (A-6) can be expressed as:

$$\hat{P} = \sum_{n=0}^{\infty} t_n(x,t)\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\} \qquad \text{(A-12)}$$

From which:

$$t_n(x,t) = \frac{2}{L_2}\int_0^{L_2} \hat{p}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\} dz \qquad \text{(A-13)}$$

Substituting equations (A-10) to (A-13) in equation (A-6), the following equation is obtained:

$$\frac{\partial F_n}{\partial t} = \hat{D}\left[\frac{\partial^2 F_n}{\partial x^2} - \left\{\frac{(2n+1)\pi}{2L_2}\right\}^2 F_n\right] + t_n(x,t) \qquad \text{(A-14)}$$

From equation (A-8):

$$v(0,z,t) = \sum_{n=0}^{\infty} F_n(0,t)\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}$$

Therefore,

$$F_n(0,t) = \frac{2}{L_2}\int_0^{L_2}\hat{B}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}dz = \beta_n \tag{A-15}$$

Again from equation (A-8):

$$v(x,z,0) = \sum_{n=0}^{\infty} F_n(x,0)\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}$$

From which:

$$F_n(x,0) = \frac{2}{L_2}\int_0^{L_2} v(x,z,0)\,\mathrm{Sin}\left\{\frac{2n+1)\pi z}{2L_2}\right\}dz$$

By equation (A-7a):

$$F_n(x,0) = \frac{2}{L_2}\int_0^{L_2}\theta\,\mathrm{Sin}\left\{\frac{2n+1)\pi z}{2L_2}\right\}dz = \gamma_n \tag{A-16}$$

From equation (A-9):

$$\frac{\partial v}{\partial x}(L_1,z,t) = \sum_{n=0}^{\infty} \frac{\partial F_n(L_1,t)}{\partial x}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}$$

Therefore, using equation (A-7d):

$$\frac{\partial F_n(L_1,t)}{\partial x} = \frac{2}{L_2}\int_0^{L_2}\bar{C}\,\mathrm{Sin}\left\{\frac{(2n+1)\pi z}{2L_2}\right\}dz = \hat{C} \tag{A-17}$$

Hence the initial and boundary condition for the problem defined by equation (A-14) are as follows:

$$F_n(x,0) = \gamma_n$$

$$F_n(0,t) = \beta_n$$

$$\frac{\partial F_n(L_1,t)}{\partial x} = \hat{C}$$

The solution for $F_n(x,t)$ can be written as:

$$F_n(x,t) = u_n(x,t) + \beta_n + \frac{x^2}{2L_1}\hat{C} \tag{A-18}$$

From equation (A-18):

$$F_n(0,t) = u_n(0,t) + \beta_n = \beta_n$$

Therefore,

$$u_n(0,t) = 0$$

Similarly,

$$\frac{\partial u_n}{\partial x}(L_1,t) = 0$$

Also, from equation (A-18):

$$\frac{\partial^2 F_n}{\partial x^2} = \frac{\partial^2 u_n}{\partial x^2} + \frac{\hat{C}}{L_1} \tag{A-19}$$

and

$$\frac{\partial F_n}{\partial t} = \frac{\partial u_n}{\partial t} \tag{A-20}$$

Substituting equations (A-18) to (A-20) in equation (A-14):

$$\frac{\partial u_n}{\partial t} = \hat{D}\left(\frac{\partial^2 u_n}{\partial x^2} - \left\{\frac{(2n+1)\pi}{2L_2}\right\}^2 u_n\right) + \hat{t}_n(x,t) \tag{A-21}$$

where,

$$\hat{t}_n(x,t) = \alpha_c - \alpha_\beta - \hat{u}\frac{x^2}{2L_1} + t_n(x,t) \tag{A-22}$$

$$\alpha_c = \frac{\hat{D}\hat{C}}{L_1}$$

$$\alpha_\beta = \hat{D}\lambda_1^2\beta_n$$

$$\hat{u} = \hat{D}\lambda_1^2\hat{C}$$

$$\lambda_1 = \frac{(2n+1)\pi}{2L_2}$$

The solution for $u_n(x,t)$ in equation (A-21) can be written as:

$$u_n(x,t) = \sum_{m=0}^{\infty} C_{mn}(t) \operatorname{Sin}\left\{\frac{2m+1)\pi x}{2L_1}\right\} \tag{A-23}$$

Here $C_{mn}(t)$ is a coefficient to be determined that depends only on time.

From equation (A-23):

$$u_n(x,0) = \sum_{m=0}^{\infty} C_{mn}(0) \operatorname{Sin}\left\{\frac{(2m+1)\pi x}{2L-1}\right\}$$

Therefore,

$$C_{mn}(0) = \frac{2}{L_1}\int_0^{L_1} u_n(x,0) \operatorname{Sin}\left\{\frac{(2m+1)\pi x}{2L_1}\right\}dx \tag{A-24}$$

From equation (A-18):

$$F_n(x,0) = \gamma_n = u_n(x,0) + \frac{x^2\hat{C}}{2L_1}$$

From which:

$$u_n(x,0) = \gamma_n - \beta_n - \frac{x^2}{2L_1}\hat{C} = \tilde{\gamma} \tag{A-25}$$

From equations (A-24) and (A-25):

$$C_{mn}(0) = \frac{2}{L_1}\int_0^{L_1} \tilde{\gamma}\, \mathrm{Sin}\left\{\frac{(2m+1)\pi x}{2L_1}\right\} dx = \hat{\gamma} \qquad \text{(A-26)}$$

Differentiating equation (A-23) twice with respect to x:

$$\frac{\partial^2 u_n}{\partial x^2} = -\sum_{m=0}^{\infty} C_{mn}(t)\left\{\frac{(2m+1)\pi}{2L_1}\right\}^2 \mathrm{Sin}\left\{\frac{(2m+1)\pi x}{2L_1}\right\} \qquad \text{(A-27)}$$

Substituting equations (A-23) and (A-27) in equation (A-21):

$$\frac{dC_{mn}(t)}{dt} + \hat{\alpha} C_{mn}(t) = t_m(x,t) \qquad \text{(A-28)}$$

where,

$$\hat{\alpha} = \hat{D}(\lambda_1^2 + \lambda_2^2)$$

$$\lambda_1 = \frac{(2n+1)\pi}{2L_2}$$

$$\lambda_2 = \frac{(2m+1)\pi}{2L_1}$$

The nonhomogeneous term $t_m(x,t)$ can be written as:

$$t_m(x,t) = \frac{2}{L_1}\int_0^{L_1} \hat{t}_n(x,t)\, \mathrm{Sin}\lambda_2 x \, dx \qquad \text{(A-29)}$$

The solution for the first order ordinary differential equation (A-28) can be written as:

$$C_{mn}(t) = \frac{1}{e^{\alpha t}}\int_0^t e^{\alpha t} t_m(x,t)\,dt + \frac{g}{e^{\alpha t}} \qquad \text{(A-30)}$$

where,

g = a constant.

From equation (A-30):

$$C_{mn}(0) = g \qquad \text{(A-31)}$$

From equations (A-26) and (A-31):

$$g = \hat{\gamma}$$

Evaluating the integral in equation (A-17):

$$\hat{C} = \frac{4\overline{C}}{(2n+1)\pi}$$

Evaluating the integral in equation (A-15):

$$\beta_n = \frac{4}{(2n+1)\pi}\left\{\overline{B} - \overline{A} - \frac{\overline{D}}{L_2}\left[\frac{L_2 \mathrm{Sin}(L_2\lambda_1)}{\lambda_1} - \frac{1}{\lambda_1^2}\right]\right\}$$

Evaluating the integral in equation (A-16):

$$\gamma_n = \frac{4}{(2n+1)\pi}\left\{\theta_o - \bar{A} - \frac{\bar{D}}{L_2}\left[\frac{L_2 \text{Sin}(L_2\lambda_1)}{\lambda_1} - \frac{1}{\lambda_1^2}\right]\right\}$$

Evaluating the integral in equation (A-26):

$$\hat{\gamma} = \frac{4}{(2m+1)\pi}\left\{\gamma_n - \beta_n - \frac{\hat{C}}{L_1}\left[\frac{L_1 \text{Sin}(L_1\lambda_2)}{\lambda_2} - \frac{1}{\lambda_2^2}\right]\right\}$$

Evaluating the integral in equation (A-13):

$$t_n(x,t) = \frac{4\alpha_D}{(2n+1)\pi} - \frac{4p}{(2n+1)\pi} = \bar{t}_n$$

where,

$$\alpha_D = \frac{\hat{D}\bar{D}}{L_2}$$

From equation (A-22):

$$\hat{t}_n(x,t) = \alpha_c - \alpha_\beta - \hat{u}\frac{x^2}{2L_1} + \bar{t}_n$$

Evaluating the integral in equation (A-29), $t_m(x,t)$ can be expressed as:

$$t_m(x,t) = \frac{4}{(2m+1)\pi}[t_{mn1} - t_{mn2}] = t_{mn} \qquad \text{(A-32)}$$

where,

$$t_{mn1} = \overline{t}_n + \alpha_c - \alpha_\beta$$

$$t_{mn2} = \frac{\hat{u}}{L_1}\left\{\frac{L_1 \text{Sin}(L_1\lambda_2)}{\lambda_2} - \frac{1}{\lambda_2^2}\right\}$$

From equations (A-30) and (A-32):

$$C_{mn}(t) = \frac{t_{mn}}{\hat{\alpha}} + \hat{\gamma}e^{-\hat{\alpha}t} \qquad \text{(A-33)}$$

Substituting equations (A-8), (A-18), (A-23), and (A-33) in equation (A-2), the moisture content θ(x,z,t) can be expressed as:

$$\theta = \sum_{n=0}^{\infty}\left[\sum_{m=0}^{\infty}\left\{\frac{t_{mn}}{\hat{\alpha}} + \hat{\gamma}e^{-\hat{\alpha}t}\right\}\text{Sin}\lambda_2 x + \beta_n + \frac{x^2}{2L_1}\hat{C}\right]\text{Sin}\lambda_1 z + \overline{A} + \frac{z^2}{2L_2}\overline{D} \qquad \text{(A-34)}$$

Differentiating equation (A-34), the fluxes in x- and z-directions are obtained as:

$$\frac{\partial\theta}{\partial x} = \sum_{n=0}^{\infty}\left\{\sum_{m=0}^{\infty}\left[\frac{t_{mn}}{\hat{\alpha}} + \hat{\gamma}e^{-\hat{\alpha}t}\right]\lambda_2 \text{Cos}(\lambda_2 x) + \frac{\hat{C}x}{L_1}\right\}\text{Sin}(\lambda_1 z)$$

$$\frac{\partial\theta}{\partial z} = \sum_{n=0}^{\infty}\left\{\sum_{m=0}^{\infty}\left[\frac{t_{mn}}{\hat{\alpha}} + \hat{\gamma}e^{-\hat{\alpha}t}\right]\mathrm{Sin}(\lambda_2 x) + \beta_n + \frac{x^2}{2L_1}\hat{C}\right]\lambda_1 \mathrm{Cos}(\lambda_1 z) + \frac{z\bar{D}}{L_2}$$

APPENDIX B

DERIVATION OF ANALYTICAL SOLUTION FOR THE SATURATED LEACHATE FLOW EQUATION

The simplified form of the leachate flow equation in the unsaturated zone is expressed as:

$$\frac{\partial h}{\partial t} = \alpha \frac{\partial^2 h}{\partial x^2} + \beta h - \overline{p} \qquad \text{(B-1)}$$

where,

$$\alpha = \frac{T}{n_e}$$

$$\beta = -\frac{K}{n_e d}$$

$$\overline{p} = \frac{-N_\ell + K_c}{n_e}$$

The initial and boundary conditions for the saturated flow problem described by equation (B-1) are as follows:

$$h(x,0) = h_i \tag{B-2}$$

$$h(L,t) = h_L \tag{B-3}$$

$$\frac{\partial h}{\partial x}(0,t) = i_o \tag{B-4}$$

where,

L = length of the saturated leachate mound from the origin at x = 0.

The solution for h(x,t) described by equation (B-1) can be written as:

$$h(x,t) = u(x,t) + i_o\left(1 - \frac{x}{L}\right) + \frac{x^2}{L^2}h_L \tag{B-5}$$

Here u(x,t) is the correction as a function of both space and time which is to be determined.

Using the initial and boundary conditions described by equations (B-2) to (B-4) in the expression (B-5), the initial and the homogeneous boundary conditions for the dependent variable u(x,t) is obtained as follows:

$$u(x,0) = u_i(x) \tag{B-6}$$

$$\frac{\partial u}{\partial x}(0,t) = 0 \tag{B-7}$$

$$u(L,t) = 0 \tag{B-8}$$

where,

$$u_i(x) = h_i - i_o x + \frac{i_o x^2}{L} - \frac{x^2}{L^2} h_L$$

Using the expression (B-5) in equation (B-1), the governing equation for the dependent variable u(x,t) can be obtained as:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} + \beta u + q \qquad \text{(B-9)}$$

where,

$$q = -\frac{2\alpha i_o}{L} + \frac{2\alpha h_L}{L^2} - \bar{p} + \beta i_o x\left(1 - \frac{x}{L}\right) + \frac{\beta x^2}{L^2} h_L \qquad \text{(B-10)}$$

The solution for u(x,t) in equation (B-9) can be written as:

$$u(x,t) = \sum_{n=0}^{\infty} C_n(t) \operatorname{Cos}\left\{\frac{(2n+1)\pi x}{2L}\right\} \qquad \text{(B-11)}$$

The cosine series expression for the nonhomogeneous term q can be written as:

$$q = \sum_{n=0}^{\infty} \phi_n(x,t) \operatorname{Cos}\left\{\frac{(2n+1)\pi x}{2L}\right\} dx \qquad \text{(B-12)}$$

Therefore,

$$\phi_n(x,t) = \frac{2}{L}\int_0^L q \, \mathrm{Cos}\left\{\frac{(2n+1)\pi x}{2L}\right\} dx = \Phi_n \tag{B-13}$$

Using equation (B-11) and (B-12) in equation (B-9), the following ordinary differential equation for $C_n(t)$ is obtained as:

$$\frac{dC_n(t)}{dt} + \vartheta C_n(t) = \phi_n(x,t) \tag{B-14}$$

where,

$$\vartheta = \lambda - \beta$$

$$\lambda = \alpha\left\{\frac{(2n+1)\pi}{2L}\right\}^2$$

The solution for the first order ordinary differential equation (B-14) is expressed as:

$$C_n(t) = \frac{1}{e^{\vartheta t}}\int_0^t e^{\vartheta t}\,\Phi_n dt + \frac{g_s}{e^{\vartheta t}} \tag{B-15}$$

where,

g_s = a constant.

From equation (B-15):

$C_n(0) = g_s$

From equation (B-11), $C_n(t)$ can be expressed as:

$$C_n(t) = \frac{2}{L}\int_0^L u(x,t)\,\mathrm{Cos}\left\{\frac{(2n+1)\pi x}{2L}\right\}dx$$

Therefore,

$$C_n(0) = \frac{2}{L}\int_0^L u(x,0)\,\mathrm{Cos}\left\{\frac{(2n+1)\pi x}{2L}\right\}dx \qquad \text{(B-16)}$$

Substituting equation (B-6) in equation (B-16) and evaluating the integral:

$$C_n(0) = g_s = \frac{2}{L}\left[h_i u_0 - i_0 u_1 + \frac{i_0}{L}u_2 - \frac{h_L}{L^2}u_2\right] \qquad \text{(B-17)}$$

where,

$$u_0 = \frac{2L}{(2n+1)\pi}(-1)^n$$

$$u_1 = \frac{2L^2}{(2n+1)\pi}\left[(-1)^n - \frac{2}{(2n+1)\pi}\right]$$

$$u_2 = \frac{2L}{(2n+1)\pi}\left[(-1)^n L^2 - \frac{8(-1)^n}{\pi^2}\left(\frac{L}{2n+1}\right)^2\right]$$

Substituting equation (B-10) in equation (B-13) and evaluating the integral, the following expression is obtained as:

$$\Phi_n = \frac{4\alpha}{L^2}u_0\left[-i_0 + \frac{h_L}{L}\right] + \frac{2}{L}\left[-\bar{p}u_0 + \beta i_0 u_1 - \beta i_0\frac{u_2}{L} + \beta h_L\frac{u_2}{L^2}\right]$$

Finally, the expression for the leachate mound head h(x,t) can be written from equations (B-5) and (B-11) as:

$$h(x,t) = \sum_{n=0}^{\infty} C_n(t)\,\mathrm{Cos}\left\{\frac{(2n+1)\pi x}{2L}\right\} + i_0 x\left(1 - \frac{x}{L}\right) + \frac{x^2}{L^2}h_L$$

where,

$$C_n(t) = \frac{\Phi_n}{\vartheta} + g_s e^{-\vartheta t}$$

APPENDIX C

FORTRAN PROGRAM FOR FILL MODEL

```
C
C ESTIMATION OF UNSTEADY STATE LEACHATE FLOW THROUGH LANDFILLS FOR
C UNSATURATED-SATURATED CONDTION IN A TWO-DIMENSIONAL DOMAIN
C                          MAIN PROGRAM
C
    IMPLICIT REAL *8(A-H,O-Z)
    COMMON /STOR/ BCONL,BCONR
    COMMON /MELT/ TOTMET,SNOWF
    COMMON /RUN/ HF,PINFIL,T,RUNOF,TOTRUN,THETAP,TAVINF,TAVRUN,TRR,HR
    COMMON /GAUMAIN/ ALBEDO,PSYCRO,AS,WP,FC,BC,SNOWP,ACEVAT,
    +DIFFUS,HCON
    COMMON /MFLO/ AQV,FLOWL,FLOWR,FLOWVV,FLOWLV,FLOWLI,BEFFICY,S
   +MEFICY,EFFICY
    COMMON /MONFO/ GRAD,BIG
    COMMON /SILA/ FLOMLL,FLOMLR,FLOWMV,BIGLAT,SMALAT,AVMOST,
   +RAINM,PERCM,RUNM,EVAM,RECHM,BIGVER,SMAVER
    COMMON /BIND/ FLOML,TOTML,TOMFLO,BIGEST,SMALST,TOTFL,TOTFV,
   +AVDALF,AVMOLF,AVMOVF,AVDAVF,AVDL,AVML,DAIFTO,TOTLL,TOTLR,
   +AVDLL,AVDLR,AVDA+LL,AVDALR,AVMLL,AVMLR,ANRAIN,
   +ANPERC,ANRUN,ANEVAT,AVDAIN,AVDAIC,AVD
   +AUN,AVDAET,AVEFCY,ANVPE,ANVERT
    PARAMETER(NOD=100,NOC=50,NEX=50,NOR=20)
    REAL *8 FLOWLV(1),THETAF(NOD),THETAP(NOD),TEMP(2,365),PRECIP(2,365
   +),DIFFUS(NOD,4),HCON(NOD,4),SLOPE(NEX),SOLRAD(2,365),WIND(365),THI
   +CK(NOC),SLOPBR(NOC)
    REAL *8 FLOWL(1),FLOWR(1),FLOML(1,2,12),TOTML(2,12),TOMFLO(2,12),T
   +OTFL(1),TOTFV(1),AVDALF(1),AVMOLF(1),AVMOVF(1),AVDAVF(1),FLOMLL(1,
   +2,12),RAIN(2,365),RAINM(2,12),PERCM(2,12),RUNM(2,12),EVAM(2,12)
    REAL *8 HIN(NOC),XLAI(365),WCF(365),PINFIL(NEX),RUNOF(NEX),HR(NEX)
   +,BC(NEX),GRAD(1,NOC),CBIG(1),CAPINF(2,12),FLORUN(2,12),HDD(NOC)
    REAL *8 HDIN(NOC),FLOMLR(1,2,12),FLOWMV(1,2,12),BIGLAT(1),SMALAT(1
   +),HDP(NOC),TOTLL(1),TOTLR(1),AVDALL(1),AVDALR(1),AVMOST(1,2,12),FL
   +OWLI(1),BEFFICY(2),SMEFICY(2)
    REAL *8 THETAM(NOD),HORAIN(24),HF(NOC,24),TOTRUN(NOC,24),RECHM(1,2
   +,12),BIGVER(1),SMAVER(1),BIG(1),BBIG(1,12),COFINT(NOD,4),VDISTB(NO
   +D,1),HCOL(1,NEX)
    INTEGER INTNO(NOD),LINKNT(NOD),LINKNB(NOD),LINKNL(NOD),LINKNR(NOD)
   +,NEXUP(NEX),ICONF(NOD),NODE(NOD),IGROW(365),NRCONF(NOD),NCCONF(NOD
   +),ICRNO(NOR,NOC),IBRCON(NOD),IBET(NOD),INEAR(NOD)
    INTEGERLCONL(NOC),ICOL(NOC),LINKNU(NEX),LINKND(NEX),LCONR(NOC),MO
   +ND(NOD)
    CHARACTER*3 MONTH(12)
    EXTERNAL IN3PUT,SNOW,CONFIG,RUNOFF,SELECT,FDMITER,SATELIM,IN4PUT,F
```

```
     +LOW,IN2PUT
      OPEN(2,FILE='LAND1.DAT',STATUS='OLD')
      OPEN(3,FILE='RESULT.OUT',STATUS='UNKNOWN')
      OPEN(5,FILE='LAND2.DAT',STATUS='OLD')
      OPEN(6,FILE='FILLINP',STATUS='OLD')
      OPEN(7,FILE='LAND3.DAT',STATUS='OLD')
      CALL IN3PUT(NINTER,NEXTUP,NEXBOT,NOROW,NOCOL,NOBAR,DT,
     +TDAYS,NYEARS,NMONTH,DTR,MONTH,AS,ALBEDO,PSYCRO,SNOWP,WP,
     +ROUGH,FORCOV,TEMPD,EXPOC,SLOPMC,COFM,HC,FC,THETAS,
     +HCONS,CONDB,THETIN,THETRR,DX,DZ,SAIE,BCOND,EFPORO,HCONCOV,DRC)
      NONODE=NINTER+NEXTUP
      NODTOT=NONODE+NEXBOT
      CALLIN4PUT(ICONF,NRCONF,NCCONF,IBRCON,INEAR,LINKNT,LINKNB,LINKNR,
     +LINKNL,SLOPBR,HDIN,NOCOL,ICOL,NEXTUP,LINKND,LINKNU,SLOPE,TDAYS,DT,
     +NYEARS,TEMP,SOLRAD,PRECIP,RAIN,XLAI,WCF,IGROW,WIND,THICK,NODTOT,LC
     +ONL,LCONR,COFINT)
      TRR=0.0
      DO 670 I=1,NOBAR
      DO 670 J=1,NYEARS
      DO 670 K=1,NMONTH
      FLOMLL(I,J,K)=0.0
      FLOMLR(I,J,K)=0.0
      FLOWMV(I,J,K)=0.0
      AVMOST(I,J,K)=0.0
      RECHM(I,J,K)=0.0
670   CONTINUE
      DO 675 I=1,NYEARS
      DO 675 J=1,NMONTH
      TOTML(I,J)=0.0
      RAINM(I,J)=0.0
      PERCM(I,J)=0.0
      RUNM(I,J)=0.0
      EVAM(I,J)=0.0
      CAPINF(I,J)=0.0
      FLORUN(I,J)=0.0
675   CONTINUE
      DO 680 I=1,NOBAR
      TOTFL(I)=0.0
      TOTLL(I)=0.0
      TOTLR(I)=0.0
      TOTFV(I)=0.0
680   CONTINUE
      BIGEST=0.0
      SMALST=0.0
      DTTIME=TDAYS-DT
      CALL CONFIG(ICONF,NODTOT,IBRCON,NOROW,NOCOL,LINKNT,LINKNB,
     +NOBAR,NONODE,NRCONF,NCCONF,NODE,NEXUP,INTNO,IBET,ICRNO)
      EDIFF=0.0
      DO 413 J=1,NODTOT
      DO 413 I=1,NOBAR
```

```
      VDISTB(J,I)=0.0
 413  CONTINUE
      DO 410 I=1,NOROW
      DO 410 J=1,NOCOL
      NN=ICRNO(I,J)
      IF(NN.EQ.0)GOTO 410
      DO 411 K=1,NOBAR
      KK=2*K+1
      IF((IBET(NN).EQ.KK).OR.(IBET(NN).EQ.K))IB=K
 411  CONTINUE
      NNB=LINKNB(NN)
      IF(ICONF(NN).EQ.4)THEN
      VDISTB(NN,IB)=0.0
      ELSE
      IF(ICONF(NNB).EQ.4)THEN
      VDISTB(NN,IB)=COFINT(NN,3)*DZ
      ELSE
      VDISTB(NN,IB)=COFINT(NN,3)*DZ+VDISTB(NNB,IB)
      ENDIF
      ENDIF
 410  CONTINUE
      DO 412 I=1,NEXTUP
      NN=NEXUP(I)
      HCOL(IB,I)=VDISTB(NN,IB)
 412  CONTINUE
      II=0
 16   II=II+1
      T=0.0
 15   T=T+DT
      write(*,*)'t=',t
      J1=T/DT
      DO 415 I=1,NOROW
      DO 415 J=1,NOCOL
      NN=ICRNO(I,J)
      IF(NN.EQ.0)GOTO 415
      DO 420 K=1,NOBAR
      KK=2*K+1
      IF((IBET(NN).EQ.KK).OR.(IBET(NN).EQ.K))IB=K
 420  CONTINUE
      IF(J1.GT.1)THEN
      IF(HDP(J).GE.VDISTB(NN,IB))THEN
      MOND(NN)=1
      ELSE
      MOND(NN)=0
      ENDIF
      ELSE
      IF(HDIN(J).GE.VDISTB(NN,IB))THEN
      MOND(NN)=1
      ELSE
      MOND(NN)=0
```

```
      ENDIF
      ENDIF
  415 CONTINUE
      CALL IN2PUT(HORAIN)
      IF(J1.EQ.254)THEN
      DARAN=0.0
      DO 29 J=1,24
      DARAN=DARAN+HORAIN(J)
  29  CONTINUE
      ELSE
      ENDIF
      DO 26 I=1,NEXTUP
      HIN(I)=0.0
  26  CONTINUE
  95  IF(TEMP(II,J1).LE.32.0)THEN
      SNOWF=SNOWP+RAIN(II,J1)
      TOTMET=0.0
      SNOWP=SNOWF
      ELSE
      IF(SNOWP.GT.0.0)THEN
      CALL SNOW(RAIN(II,J1),TEMP(II,J1),SOLRAD(II,J1),WIND(J1),FORCOV,EX
     +POC,ALBEDO,SNOWP,COFM,TEMPD)
      ELSE
      TOTMET=0.0
      SNOWF=0.0
      SNOWP=SNOWF
      ENDIF
      ENDIF
      IF((TEMP(II,J1).LE.32.0).OR.((PRECIP(II,J1)+TOTMET).EQ.0.0))THEN
      DO 21 I=1,NEXTUP
      HR(I)=0.0
      DO 687 K=1,24
      HF(I,K)=0.0
      TOTRUN(I,K)=0.0
  687 CONTINUE
      RUNOF(I)=0.0
      PINFIL(I)=0.0
      TAVINF=0.0
      TAVRUN=0.0
      FLONET=0.0
  21  CONTINUE
      ELSE
      DO 31 I=1,NONODE
      IF((II.EQ.1).AND.(J1.EQ.1))THEN
      THETAP(I)=THETIN
      ELSE
      ENDIF
  31  CONTINUE
      CALL RUNOFF(TOTMET,PRECIP,HIN,SLOPE,ROUGH,DTR,THETAS,HCONS,HORAIN,
     +HC,FC,II,J1,NEXTUP,NEXUP,LINKNB,ICONF,LCONL,LCONR,NOCOL,FLONET,DX,
```

```
     +TEMP,NODE,NONODE,BCOND,COFINT,HCONCOV)
      ENDIF
      CALL FDMITER(THETIN,SLOPMC,HCONS,SAIE,THETAS,LINKNT,LINKNR,LINKNB,
     +LINKNL,ICONF,THETAF,DT,DX,DZ,J1,NONODE,NEXTUP,NEXUP,II,NODE,PRECIP
     +(II,J1),HR,TEMP(II,J1),RAIN(II,J1),SOLRAD(II,J1),IGROW(J1),XLAI(J1
     +),WCF(J1),TAVRUN,THETAP,COFINT,MOND)
      AVIN=RAIN(II,J1)-TAVRUN-ACEVAT
      CALL SATELIM(NODTOT,NOBAR,NOROW,NOCOL,ICRNO,ICONF,IBET,
     +LINKNB,NEXTUP,NEXUP,THETAS,BC,HDIN,ICOL,LCONL,LCONR,HCONS,
     +CONDB,EFPORO,BCONL,BCONR,DT,DX,DZ,THICK,TAVRUN,ACEVAT,
     +PRECIP(II,J1),NODE,NONODE,SLOPBR,THETAF,II,J1,GRAD,THETAM,
     +MOND,LINKNT,HDD,HDP,TAVSOR,HCOL,SLOPMC,
     +SAIE,THETAP,VDISTB,HCON,DIFFUS,FC,LINKNR,LINKNL,DRC)
      DO 139 I=1,NOBAR
      IF(J1.EQ.1)CBIG(I)=0.0
 139  CONTINUE
      CALL FLOW(NOCOL,NOBAR,HDD,SLOPBR,HCONS,J1,II,RAIN(II,J1),THICK,CON
     +DB)
      AVTHET=0.0
      DO 902 I=1,NONODE
      NN=NODE(I)
      AVTHET=AVTHET+THETAF(I)
 902  CONTINUE
      DO 903 I=1,NEXBOT
      AVTHET=AVTHET+THETAS
 903  CONTINUE
      AVTHET=AVTHET/NODTOT
      IF((J1.EQ.1).OR.(J1.EQ.182).OR.(J1.EQ.365))THEN
      WRITE(3,1000)J1
 1000 FORMAT(5X,'TIME T = ',I3,1X,'DAY(S)')
      WRITE(3,750)
 750  FORMAT(5X,'TABLE :  DEPTH OF LEACHATE MOUND',//,5X,'______________
     +_____________________________',/)
      WRITE(3,751)
 751  FORMAT(5X,'BARRIER',7X,'COLUMN NO.',5X,'DEPTH OF MOUND')
      WRITE(3,752)
 752  FORMAT(38X,'(FT)',/,5X,'_______',7X,'__________',5X,'_____________
     +_',/)
      DO 93 J=1,NOCOL
      WRITE(3,94)J,HDD(J)
 94   FORMAT(13X,I3,8X,F10.2)
 93   CONTINUE
      ELSE
      ENDIF
      CALL SELECT(NOBAR,FLOWL,FLOWR,AQV,J1,II,AVTHET,RAIN,AVIN,TAVRUN,AC
     +EVAT,TAVSOR,BIG,BBIG,TAVINF,FLONET,CAPINF,FLORUN)
      IF(T.LE.DTTIME)GOTO 15
      JJ=J1
      IF(II.LT.NYEARS)GOTO 16
      CALL FIND(NYEARS,NMONTH,NOBAR,FLOMLL,FLOMLR,FLOWMV,BIGLAT,
```

```
   +SMALAT,TDAYS,RAINM,RUNM,EVAM,RECHM,FLORUN,CAPINF,ANFLOW,BALFLO)
    WRITE(3,791)
791 FORMAT(1X,'TABLE :  MONTHLY VALUES OF PRECIPITATION, RUNOFF, INPUT
   +,     ',/,9X,'AND EVAPOTRANSPIRATION IN INCHES',/,1X,'__________
   +_____________________________________________',/)
    DO 756 I=1,NYEARS
    WRITE(3,761)(MONTH(K),K=1,6)
761 FORMAT(17X,2(4X,A3),4(5X,A3),/)
    WRITE(3,762)(RAINM(I,K),K=1,6)
762 FORMAT(1X,'PRECIPITATION',2X,6(2X,F6.2),//)
    WRITE(3,811)(RUNM(I,K),K=1,6)
811 FORMAT(1X,'RUNOFF',9X,6(2X,F6.2),//)
    WRITE(3,812)(PERCM(I,K),K=1,6)
812 FORMAT(1X,'NET INFILT. ',3X,6(2X,F6.2),//)
    WRITE(3,813)(EVAM(I,K),K=1,6)
813 FORMAT(1X,'EVAPOTRANSP.',3X,6(2X,F6.2),//)
    WRITE(3,763)(MONTH(K),K=7,12)
763 FORMAT(17X,2(4X,A3),4(5X,A3),/)
    WRITE(3,792)(RAINM(I,K),K=7,12)
792 FORMAT(1X,'PRECIPITATION',2X,6(2X,F6.2),//)
    WRITE(3,816)(RUNM(I,K),K=7,12)
816 FORMAT(1X,'RUNOFF',9X,6(2X,F6.2),//)
    WRITE(3,817)(PERCM(I,K),K=7,12)
817 FORMAT(1X,'NET INFILT. ',3X,6(2X,F6.2),//)
    WRITE(3,818)(EVAM(I,K),K=7,12)
818 FORMAT(1X,'EVAPOTRANSP.',3X,6(2X,F6.2),///)
756 CONTINUE
    WRITE(3,757)
757 FORMAT(1X,'TABLE :  MONTHLY VALUES OF AVERAGE MOISTURE CONTENTS
   +IN THE UNSATURATED ZONE',/,1X,'_____________________________
   +______________________________',/)
    DO 759 I=1,NOBAR
    WRITE(3,758)I
758 FORMAT(10X,'AVERAGE MOISTURE CONTENTS UPON BARRIER NO. ',I2,/,10X,
   +'_____________________________________',/)
    DO 759 J=1,NYEARS
    WRITE(3,801)J
801 FORMAT(///,1X,'MONTHLY VALUES FOR THE YEAR ',I2,':',//)
    WRITE(3,773)(MONTH(K),K=1,6)
773 FORMAT(1X,2(7X,A3),4(8X,A3),/)
    WRITE(3,774)(AVMOST(I,J,K),K=1,6)
774 FORMAT(1X,6(3X,F8.3),//)
    WRITE(3,776)(MONTH(K),K=7,12)
776 FORMAT(1X,2(7X,A3),4(8X,A3),/)
    WRITE(3,777)(AVMOST(I,J,K),K=7,12)
777 FORMAT(1X,6(3X,F8.3))
759 CONTINUE
    WRITE(3,781)
781 FORMAT(/////1X,'TABLE :  MONTHLY VALUES OF RECHARGE ONTO THE LEACH
   +ATE MOUND',/,9X,'IN INCHES',/,1X,'_____________________________
```

```
   +________________________________',//)
    DO 784 I=1,NOBAR
    WRITE(3,783)I
783 FORMAT(9X,'RECHARGE VALUES ONTO THE MOUND UPON BARRIER  NO. ',I2,/
   +,9X,'__________________________________________________')
    DO 784 J=1,NYEARS
    WRITE(3,721)J
721 FORMAT(///,1X,'MONTHLY VALUES FOR THE YEAR ',I2,':',//)
    WRITE(3,722)(MONTH(K),K=1,6)
722 FORMAT(1X,2(7X,A3),4(8X,A3),/)
    WRITE(3,723)(RECHM(I,J,K),K=1,6)
723 FORMAT(1X,6(3X,F8.2),/)
    WRITE(3,724)(MONTH(K),K=7,12)
724 FORMAT(1X,2(7X,A3),4(8X,A3),/)
    WRITE(3,726)(RECHM(I,J,K),K=7,12)
726 FORMAT(1X,6(3X,F8.2))
784 CONTINUE
    WRITE(3,790)
790 FORMAT(/////,1X,'TABLE :  MONTHLY VALUES OF LATERAL AND VERTICAL F
   +LOW',/)
    WRITE(3,795)
795 FORMAT(1X,'_______________________________________________________
   +__________',/)
    DO 755 I=1,NOBAR
    WRITE(3,760)I
760 FORMAT(21X,'FLOW DUE TO BARRIER NO. ',I2)
    WRITE(3,800)
800 FORMAT(21X,'___________________________',/)
    DO 755 J=1,NYEARS
    WRITE(3,765)J
765 FORMAT(///,1X,'MONTHLY VALUES FOR THE YEAR ',I2,' :',//)
    WRITE(3,771)(MONTH(K),K=1,6)
771 FORMAT(18X,2(4X,A3),4(5X,A3),/)
    WRITE(3,770)(FLOMLL(I,J,K),K=1,6)
770 FORMAT(1X,'LATERAL FLOW',/,1X,'(LEFT DIRECTION)',1X,6(2X,F6.2),/)
    WRITE(3,772)(FLOMLR(I,J,K),K=1,6),(FLOML(I,J,K),K=1,6),(FLOWMV(I,J
   +,K),K=1,6)
772 FORMAT(1X,'LATERAL FLOW',/,1X,'(RIGHT DIRECTION)',6(2X,F6.2),//,1X
   +,'TOTAL MONTHLY',/,1X,'LATERAL FLOW',5X,6(2X,F6.2),//,1X,'VERTICAL
   + FLOW',4X,6(2X,F6.4),//)
    WRITE(3,775)(MONTH(K),K=7,12)
775 FORMAT(18X,2(4X,A3),4(5X,A3),/)
    WRITE(3,780)(FLOMLL(I,J,K),K=7,12)
780 FORMAT(1X,'LATERAL FLOW',/,1X,'(LEFT DIRECTION)',1X,6(2X,F6.2),/)
    WRITE(3,785)(FLOMLR(I,J,K),K=7,12),(FLOML(I,J,K),K=7,12),(FLOWMV(I
   +,J,K),K=7,12)
785 FORMAT(1X,'LATERAL FLOW',/,1X,'(RIGHT DIRECTION)',6(2X,F6.2),//,1X
   +,'TOTAL MONTHLY',/,1X,'LATERAL FLOW',1X,6(2X,F6.2),//,1X,'VERTICAL
   + FLOW',4X,6(2X,F6.4))
755 CONTINUE
```

```
      WRITE(3,805)
  805 FORMAT(/,1X,'_______________________________________________________
     +____________',/,1X,'LATERAL FLOW IN CU.FT. PER UNIT WIDTH OF LANDF
     +ILL',/,1X,'VERTICAL FLOW IN CU.FT. PER UNIT AREA OF LANDFILL',////
     +/)
      WRITE(3,810)
  810 FORMAT(1X,'TABLE :  TOTAL MONTHLY LATERAL FLOW FROM
     +THE LANDFILL',/)
      WRITE(3,815)
  815 FORMAT(1X,'_______________________________________________________
     +______')
      DO 685 I=1,NYEARS
      WRITE(3,635)I
  635 FORMAT(/,28X,'YEAR ',I3)
      WRITE(3,820)
  820 FORMAT(28X,'________',//)
      WRITE(3,655)(MONTH(J),J=1,6)
  655 FORMAT(3X,6(7X,A3),/)
      WRITE(3,690)(TOTML(I,J),J=1,6)
  690 FORMAT(1X,6(2X,F8.4),//)
      WRITE(3,700)(MONTH(J),J=7,12)
  700 FORMAT(3X,6(7X,A3),/)
      WRITE(3,695)(TOTML(I,J),J=7,12)
  695 FORMAT(1X,6(2X,F8.4),//)
  685 CONTINUE
      WRITE(3,825)
  825 FORMAT(1X,'_______________________________________________________
     +______',/,1X,'LATERAL FLOW IN CU.FT. PER UNIT WIDTH OF LANDFILL',/
     +////)
      WRITE(3,830)
  830 FORMAT(1X,'TABLE :  AVERAGE DAILY AND MONTHLY FLOWS UPON
     +EACH BARRIER',//,1X,'_________________________________________
     +____________',/)
      DO 720 I=1,NOBAR
      WRITE(3,715)I
  715 FORMAT(1X,'FLOW DUE TO BARRIER ',I2,/,1X,'_____________________',
     +/)
      WRITE(3,725)AVDALL(I),AVDALR(I),AVDALF(I),AVMOLF(I),AVDAVF(I),AVMO
     +VF(I)
  725 FORMAT(1X,'AVERAGE DAILY LATERAL FLOW (LEFT DIRECTION)',1X,F10.4,/
     +/,1X,'AVERAGE DAILY LATERAL FLOW (RIGHT DIRECTION)',5X,F10.4,//,1X
     +,'AVERAGE DAILY LATERAL FLOW',10X,F10.4,//,5X,'AVERAGE MONTHLY LAT
     +ERAL FLOW',10X,F10.4,//,1X,' AVERAGE DAILY VERTICAL FLOW',10X,F10.
     +4,//,1X,'AVERAGE MONTHLY VERTICAL FLOW',10X,F10.4,//,1X,'_________
     +__________________________________________________',/////)
  720 CONTINUE
      WRITE(3,730)
  730 FORMAT(1X,'TABLE :  AVERAGE, PEAK, AND MINIMUM VALUES OF FLOW',/,1
     +4X,'UPON ALL OF THE BARRIERS',//,1X,'___________________________
     +_____________________',/)
```

```
      WRITE(3,650)BIGEST,SMALST,AVDLL,AVDLR,AVDL
  650 FORMAT(1X,'PEAK DAILY LATERAL FLOW   ',6X,F10.4,//,1X,'MINIMUM DAIL
     +Y LATERAL FLOW  ',4X,F10.4,//,1X,'AVERAGE DAILY LATERAL FLOW (LEFT
     +DIRECTION)',3X,F10.4,//,1X,'AVERAGE DAILY LATERAL FLOW (RIGHT DIRE
     +CTION)',3X,F10.4,//,1X,'AVERAGE DAILY LATERAL FLOW',10X,F10.4,//)
      WRITE(3,651)AVMLL,AVMLR,AVML
  651 FORMAT(1X,'AVERAGE MONTHLY LATERAL FLOW (LEFT DIRECTION)',3X,F10.4
     +,//,1X,'AVERAGE MONTHLY LATERAL FLOW (RIGHT DIRECTION)',3X,F10.4,/
     +/,1X,'AVERAGE MONTHLY LATERAL FLOW   ',1X,F10.4,//,1X,'___________
     +____________________________________',/////)
  113 STOP
      END




C
C THIS SUBROUTINE READS DATA FROM THE DATA FILE
C
      SUBROUTINE IN3PUT(NINTER,NEXTUP,NEXBOT,NOROW,NOCOL,NOBAR,DT,
     +TDAYS,NYEARS,NMONTH,DTR,MONTH,AS,ALBEDO,PSYCRO,SNOWP,WP,
     +ROUGH,FORCOV,TEMPD,EXPOC,SLOPMC,COFM,HC,FC,THETAS,HCONS,
     +CONDB,THETIN,THETRR,DX,DZ,SAIE,BCOND,EFPORO,HCONCOV,DRC)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /STOR/ BCONL,BCONR
      CHARACTER*3 MONTH(12)
      READ(2,*)NINTER,NEXTUP,NEXBOT,NOROW,NOCOL,NOBAR
      READ(2,*)DT,TDAYS,NYEARS,NMONTH,DTR,DX,DZ
      READ(2,212)(MONTH(I),I=1,NMONTH)
  212 FORMAT(12A3)
      READ(2,*)AS,ALBEDO,PSYCRO,SNOWP,WP,EFPORO
      READ(2,*)ROUGH,FORCOV,TEMPD,EXPOC,SLOPMC,COFM,HC
  185 FORMAT(6F10.3)
      READ(2,*)BCONL,BCONR,FC,THETAS,HCONS,HCONCOV,CONDB,THETIN
      READ(2,*)THETRR,SAIE,BCOND,DRC
      RETURN
      END




C
C THIS SUBROUTINE READS DATA FROM THE DATA FILE
C
      SUBROUTINE IN4PUT(ICONF,NRCONF,NCCONF,IBRCON,INEAR,LINKNT,LINKNB,L
     +INKNR,LINKNL,SLOPBR,HDIN,NOCOL,ICOL,NEXTUP,LINKND,LINKNU,SLOPE,TDA
     +YS,DT,NYEARS,TEMP,SOLRAD,PRECIP,RAIN,XLAI,WCF,IGROW,WIND,THICK,NOD
     +TOT,LCONL,LCONR,COFINT)
      IMPLICIT REAL *8(A-H,O-Z)
```

```
      PARAMETER(NOD=100,NOC=50)
      REAL *8 SLOPE(NOC),PRECIP(2,365),TEMP(2,365),SOLRAD(2,365),WIND(36
     +5),XLAI(365),WCF(365),THICK(NOC),SLOPBR(NOC),HDIN(NOC),RAIN(2,365)
     +,COFINT(NOD,4)
      INTEGER LINKNT(NOD),LINKNB(NOD),LINKNL(NOD),LINKNR(NOD),IGROW(365)
     +,ICONF(NOD),NRCONF(NOD),NCCONF(NOD),IBRCON(NOD),INEAR(NOD),LCONL(N
     +OC),LCONR(NOC),ICOL(NOC),LINKND(NOC),LINKNU(NOC)
      DO 161 I=1,NODTOT
      READ(6,*)ICONF(I),NRCONF(I),NCCONF(I),IBRCON(I),INEAR(I)
  161 CONTINUE
      DO 190 I=1,NODTOT
      READ(6,*)LINKNT(I),LINKNB(I),LINKNR(I),LINKNL(I)
  190 CONTINUE
      DO 370 I=1,NOCOL
      READ (6,*)LCONL(I),LCONR(I),SLOPBR(I),HDIN(I),THICK(I)
  370 CONTINUE
      DO 375 I=1,NOCOL
      ICOL(I)=1
  375 CONTINUE
      ICOL(NOCOL+1)=2
      ICOL(NOCOL+2)=2
      DO 380 I=1,NEXTUP
      READ(6,*)LINKND(I),LINKNU(I)
  380 CONTINUE
      DO 181 I=1,NEXTUP
      READ(6,*)SLOPE(I)
  181 CONTINUE
      DO 182 I=1,NODTOT
      READ(6,*)(COFINT(I,J),J=1,4)
  182 CONTINUE
      NSTEP=TDAYS/DT
      DO 210 I=1,NYEARS
      DO 210 J=1,NSTEP
      READ(2,*)TEMP(I,J),SOLRAD(I,J)
  210 CONTINUE
      WRITE(*,*)'returned from 14th read'
      DO 213 I=1,NYEARS
      DO 213 J=1,NSTEP
      READ(7,*)PRECIP(I,J)
      RAIN(I,J)=PRECIP(I,J)
  213 CONTINUE
      DO 200 I=1,NSTEP
      READ(2,*)XLAI(I),WCF(I),IGROW(I),WIND(I)
  200 CONTINUE
      RETURN
      END
```

```
C
C THIS SUBROUTINE READS HOURLY RAINFALL DATA
C
      SUBROUTINE IN2PUT(HORAIN)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL *8 HORAIN(24)
      READ(5,*)(HORAIN(J),J=1,24)
      RETURN
      END
```

```
C
C THIS SUBROUTINE GIVES CONFIGURATION TO IDENTIFY
C INTERNAL AND EXTERNAL NODES
C
      SUBROUTINE CONFIG(ICONF,NODTOT,IBRCON,NOROW,NOCOL,LINKNT,LINKNB,
     +NOBAR,NONODE,NRCONF,NCCONF,NODE,NEXUP,INTNO,IBET,ICRNO)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER (NOD=100,NEX=50,NI=80,NO=100,NOR=20,NOC=50)
      INTEGER ICONF(NOD),NEXUP(NEX),INTNO(NI),NODE(NO),NRCONF(NOD),NCCON
     +F(NOD),ICRNO(NOR,NOC),IBRCON(NOD),IBET(NOD),LINKNT(NOD),LINKNB(NOD
     +)
      I1=1
      I2=1
      DO 240 I=1,NODTOT
      IF((ICONF(I).EQ.1).OR.((ICONF(I).EQ.2).OR.(ICONF(I).EQ.3)))THEN
      NEXUP(I1)=I
      I1=I1+1
      ELSE
      IF(ICONF(I).EQ.5)THEN
      INTNO(I2)=I
      I2=I2+1
      ELSE
      GOTO 240
      ENDIF
      ENDIF
240   CONTINUE
      I3=1
      DO 250 I=1,NODTOT
      IF((ICONF(I).EQ.1).OR.(ICONF(I).EQ.2).OR.(ICONF(I).EQ.3).OR.(ICONF
     +(I).EQ.5))THEN
      NODE(I3)=I
      I3=I3+1
      ELSE
      GOTO 250
      ENDIF
```

```
 250  CONTINUE
      DO 275 I=1,NOROW
      DO 275 J=1,NOCOL
      ICRNO(I,J)=0
 275  CONTINUE
      DO 260 I=1,NODTOT
      DO 265 J=1,NOROW
      IF(NRCONF(I).EQ.J)IR=J
 265  CONTINUE
      DO 270 K=1,NOCOL
      IF(NCCONF(I).EQ.K)IC=K
 270  CONTINUE
      ICRNO(IR,IC)=I
 260  CONTINUE
      DO 280 I=1,NOROW
      DO 280 J=1,NOCOL
      NN=ICRNO(I,J)
      IF(NN.EQ.0)GOTO 280
      IF(IBRCON(NN).EQ.0)THEN
      NNT=NN
 300  NNT=LINKNT(NNT)
      IF(NNT.EQ.0)THEN
      NNB=NN
 295  NNB=LINKNB(NNB)
      IF(NNB.EQ.0)GOTO 280
      DO 285 K=1,NOBAR
      IF((IBRCON(NNB).EQ.K).OR.(ICONF(NNB).EQ.4))THEN
      IBET(NN)=2*K+1
      ELSE
      IBET(NN)=0
      ENDIF
 285  CONTINUE
      IF(IBET(NN).EQ.0)GOTO 295
      ELSE
      DO 290 K=1,NOBAR
      IF(IBRCON(NNT).EQ.K)THEN
      IBET(NN)=2*K-1
      ELSE
      IBET(NN)=0
      ENDIF
 290  CONTINUE
      IF(IBET(NN).EQ.0)GOTO 300
      ENDIF
      ELSE
      IBET(NN)=IBRCON(NN)
      ENDIF
 280  CONTINUE
      RETURN
      END
```

```
C
C THIS SUBROUTINE COMPUTES TOTAL DAILY SNOWMELT
C
      SUBROUTINE SNOW(RAINN,TEMPP,SOLRDD,WINDD,FORCOV,EXPOC,ALBEDO,
     +SNOWP,COFM,TEMPD)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /MELT/ TOTMET,SNOWF
      IF(RAINN.GT.0.0)THEN
      TEMA=TEMPP-32.0
      RAINMT=0.029+0.0084*EXPOC*WINDD+0.007*RAINN
      TOTMET=0.09+RAINMT*TEMA
      IF(TOTMET.GT.SNOWP)TOTMET=SNOWP
      IF(TOTMET.LT.0.0)TOTMET=0.0
      ELSE
      SOLMET=0.004*SOLRDD*COFM*(1.0-FORCOV)*(1.0-ALBEDO)
      TEMT=0.22*(TEMPP-32.0)
      TEMDM=0.78*(TEMPD-32.0)
      FMT=0.029*FORCOV*(TEMPP-32.0)
      TEMWMT=0.0084*EXPOC*WINDD*(TEMT+TEMDM)
      TOTMET=SOLMET+FMT+TEMWMT
      IF(TOTMET.GT.SNOWP)TOTMET=SNOWP
      IF(TOTMET.LT.0.0)TOTMET=0.0
      ENDIF
      SNOWF=SNOWP-TOTMET+RAINN
      IF(SNOWF.LT.0.0)SNOWF=0.0
      SNOWP=SNOWF
      RETURN
      END
```

```
C
C THIS SUBROUTINE COMPUTES DAILY VALUES OF ACTUAL
C ACTUAL EVAPOTRANSPIRATION
C
      SUBROUTINE EVAT(EDIFF,TEMPER,ALBEDO,SOLRDD,PSYCRO,IGRRO,XXLAI,
     +WCFF,AS,THETA,WP,FC,RAINN,TAVRUN,TOTMET,ACEVAT,J1)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /AET/ IT,PT,EVAS,IN,SNOP,ESS
      IN=0
      TK=5.0*(TEMPER-32.0)/9.0 + 273.0
      EXPO=21.255-5304.0/TK
      SLOPVP=5304.0*EXP(EXPO)/(TK**2)
      SOLNET=(1.0-ALBEDO)*SOLRDD/58.3
      POTEV=1.28*SLOPVP*SOLNET/(25.4*(SLOPVP+PSYCRO))
      TINTMX=0.05*XXLAI/3.0
      IF(TINTMX.EQ.0.0)TINTMX=0.05
```

```
TINTI=0.0
TINEXP=0.0
J2=J1-1
IF((SNOP+RAINN).GE.POTEV)ESS=POTEV
IF((SNOP+RAINN).LT.POTEV)ESS=SNOP+RAINN
IF(TEMPER.LT.32.0)SNOF=SNOP+RAINN-ESS
IF((TEMPER.GT.32.0).AND.(ESS.LE.(TOTMET+RAINN)))SNOF=SNOP-ESS
IF((TEMPER.GT.32.0).AND.(ESS.GT.(TOTMET+RAINN)))SNOF=SNOP-ESS
SNOP=SNOF
IF(IGRRO.EQ.1)THEN
POTEVS=POTEV*EXP(-0.4*XXLAI)
ELSE
POTEVS=POTEV*EXP(-0.4*WCFF)
ENDIF
UPLIM=(9.0*(AS-3.0)**0.42)/25.4
EVAS=POTEV
AVINFF=RAINN+TOTMET-TAVRUN-ESS
IF(AVINFF.LT.0.0)AVINFF=0.0
EDIFF=EDIFF+EVAS-AVINFF
IF(EDIFF.LT.UPLIM)THEN
IN=1
IT=J1
EVAS1=POTEVS
ELSE
IF(J1.EQ.1)PT=0.0
YT=J1-PT
YT1=YT-1.0
IF(YT1.LE.0.0)YT1=0
IF(YT.LE.0.0)YT=0
EVAS2=AS*(SQRT(YT)-SQRT(YT1))/25.4
ENDIF
IF((EDIFF.LT.UPLIM).AND.((ESS+EVAS1).LE.POTEV))EVAS=EVAS1
IF((EDIFF.GE.UPLIM).AND.((ESS+EVAS2).LE.POTEV))EVAS=EVAS2
IF((EDIFF.LT.UPLIM).AND.((ESS+EVAS1).GT.POTEV))EVAS=POTEV-ESS
IF((EDIFF.GE.UPLIM).AND.((ESS+EVAS2).GT.POTEV))EVAS=POTEV-ESS
POTEVP=(POTEV*XXLAI)/3.0
SUM=POTEVP+ESS+EVAS
IF(SUM.LE.POTEV)THEN
EPD=POTEVP
ELSE
EPD=POTEV-ESS-EVAS
ENDIF
EVAP=EPD*(1.20-4.0*EPD+(THETA-WP)/(FC-WP))
IF(EVAP.GT.EPD)EVAP=EPD
ADDEV=EVAS+EVAP
DIFFEV=POTEV-ESS
IF(ADDEV.GT.DIFFEV)THEN
ACEVAT=DIFFEV
ELSE
ACEVAT=ADDEV
```

```
      ENDIF
      RETURN
      END



C
C THIS SUBROUTINE COMPUTES SURFACE RUNOFF ALONG THE UPPER BOUNDARY
C
      SUBROUTINERUNOFF(TOTMET,PRECIP,HIN,SLOPE,ROUGH,DTR,THETAS,HCONS,H
     +ORAIN,HC,FC,II,J1,NEXTUP,NEXUP,LINKNB,ICONF,LCONL,LCONR,NOCOL,FLON
     +ET,DX,TEMP,NODE,NONODE,BCOND,COFINT,HCONCOV)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /RUN/ HF,PINFIL,T,RUNOF,TOTRUN,THETAP,TAVINF,TAVRUN,TRR,HR
      PARAMETER (NOD=100,NOC=50,NS=220,NCC=20)
      REAL *8 PRECIP(2,365),HF(NOC,24),HIN(NOC),HP(NOC),SLOPE(NOC),PINFI
     +L(NOC),THETAP(NOD),RUNOF(NOC),TOTRUN(NOC,24),HORAIN(24),HR(NOC),QF
     +F(NOC),QFP(NOC),QFB(NOC),TEMP(2,365),DDX(NOC),FLOW(NOC),QPP(NOC),H
     +FF(NOC),COFINT(NOD,4)
      INTEGERNEXUP(NOC),LINKNB(NOD),ICONF(NOD),N1(NOD),LCONL(NOC),LCONR
     +(NOC)
      TOTLEN=0.0
      DO 150 I=1,NEXTUP
      XX=ATAN(SLOPE(I))
      DDX(I)=DX/(COS(XX))
      TOTLEN=TOTLEN+DDX(I)
 150  CONTINUE
      IF(TOTMET.GT.0.0)PRECIP(II,J1)=PRECIP(II,J1)+TOTMET
      NO1COL=NOCOL+1
      NO2COL=NOCOL+2
      IF(J1.EQ.1)AVRUN=0.0
      TAVRUN=0.0
      TAVINF=0.0
      NOR=0
      TRS=0.0
      HRON=0.0
      OUTFLOW=0.0
      TOTFLOW=0.0
      TOTDEP=0.0
      FLONET=0.0
      HHRON=0.0
      TR=0.0
 140  TR=TR+DTR
      J11=TR/DTR
      IF(TOTMET.GT.0.0)HORAIN(J11)=HORAIN(J11)+TOTMET/24.0
      DTRR=3600.0
      TTR=0.0
      HOFLOW=0.0
```

```
      HDEP=0.0
142   TTR=TTR+DTRR
      J1R=TTR/DTRR
      DO 135 I=1,NEXTUP+2
      HIN(I)=0.0
      IF((II.NE.1).AND.(J1.EQ.1))THEN
      IF(TEMP(II-1,365).LE.32.0)THEN
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=HIN(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
      ELSE
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=QFB(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
      ENDIF
      ELSE
      ENDIF
      IF((II.NE.1).AND.(J1.NE.1))THEN
      IF(TEMP(II,J1-1).LE.32.0)THEN
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=HIN(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
      ELSE
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=QFB(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
      ENDIF
      ELSE
      ENDIF
      IF((II.EQ.1).AND.(J1.NE.1))THEN
      IF(TEMP(II,J1-1).LE.32.0)THEN
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=HIN(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
      ELSE
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=QFB(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
```

```
      ENDIF
      ELSE
      ENDIF
      IF((II.EQ.1).AND.(J1.EQ.1))THEN
      IF((J11.EQ.1).AND.(J1R.EQ.1))THEN
      QFP(I)=HIN(I)
      ELSE
      QFP(I)=QFF(I)
      ENDIF
      ELSE
      ENDIF
135   CONTINUE
      DO 173 I=1,NEXTUP+2
      IF((II.EQ.1).AND.(J1.EQ.1))QPP(I)=QFP(I)
173   CONTINUE
      DO 155 I=1,NEXTUP
      ALPHA=(ROUGH/(1.49*DSQRT(SLOPE(I))))**0.6
      HP(I)=(QFP(I)**0.6)*ALPHA
155   CONTINUE
      IF(HORAIN(J11).EQ.0.0)TRS=0.0
      IF(HORAIN(J11).GT.0.0)TRS=TRS+DTRR
      TRS=TRS+DTRR
      IF(J11.EQ.1)THEN
      TT=TTR
      ELSE
      IF((J11.NE.1).AND.(HORAIN(J11-1).EQ.0.0).AND.(HORAIN(J11).EQ.0.0))
     +THEN
      TT=TTR
      ELSE
      IF((J11.NE.1).AND.(HORAIN(J11-1).GT.0.0).AND.(HORAIN(J11).GT.0.0))
     +THEN
      TT=TRS
      ELSE
      IF((J11.NE.1).AND.(HORAIN(J11-1).EQ.0.0).AND.(HORAIN(J11).GT.0.0))
     +THEN
      TT=TRS
      ELSE
      IF((J11.NE.1).AND.(HORAIN(J11-1).GT.0.0).AND.(HORAIN(J11).EQ.0.0))
     +THEN
      TT=TTR
      ELSE
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
145   DO 118 I=1,NEXTUP
136   NN=NEXUP(I)
      NNL=LCONL(I)
      NNR=LCONR(I)
```

```
      THETDF=THETAS-THETAP(NN)
      IF(HDF.LT.0.0)HDF=0.0
      XX=ATAN(SLOPE(I))
      COSX=COS(XX)
      HDF=(HP(I)+HC/12.0)*COSX
      IF(HDF.LT.0.0)HDF=0.0
      SORP=COSX*DSQRT(2.0*HCONCOV*THETDF*HDF/(24.0*3600.0))
      CAPINL=0.5*SORP/(SQRT(TT))
      IF((HORAIN(J11)/(12.0*3600.0)).LE.CAPINL)THEN
      PINFIL(I)=HORAIN(J11)/(12.0*3600.0)
      PFLOW=0.0
      ELSE
      PINFIL(I)=CAPINL
      PFLOW=HORAIN(J11)/(12.0*3600.0)-PINFIL(I)
      ENDIF
      HRANN=HORAIN(J11)/(12.0)
      FLOW(I)=PFLOW
118   CONTINUE
      DO 112 I=1,NEXTUP
      NNL=LCONL(I)
      NNR=LCONR(I)
      IF(I.EQ.1)THEN
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(I))))**0.6
      AVQ=(QFP(I)+QPP(I))/2.0
      QFNL2=DTRR*(HORAIN(J11)/(12.0*3600.0))*(AVQ**0.4)/(ALPHA*0.6)
      BCONDR=1.49*DSQRT(SLOPE(I))*1.67*((HORAIN(J11)/(12.0*3600.0))**0.6
     +7)*BCOND/ROUGH
      QFNL3=DTRR*BCONDR*(AVQ**0.4)/(ALPHA*0.6)
      QFF(NNL)=QFP(NNL)+QFNL2-QFNL3
      DDX(NNL)=DDX(I)
      FLOW(NNL)=FLOW(I)
      ELSE
      IF(I.EQ.NEXTUP)THEN
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(I))))**0.6
      AVQ=(QFP(NNR)+QPP(NNR))/2.0
      QFNL2=DTRR*(HORAIN(J11)/(12.0*3600.0))*(AVQ**0.4)/(ALPHA*0.6)
      BCONDR=1.49*DSQRT(SLOPE(I))*1.67*((HORAIN(J11)/(12.0*3600.0))**0.6
     +7)*BCOND/ROUGH
      QFNL3=DTRR*BCONDR*(AVQ**0.4)/(ALPHA*0.6)
      QFF(NNR)=QFP(NNR)+QFNL2-QFNL3
      DDX(NNR)=DDX(I)
      FLOW(NNR)=FLOW(I)
      ELSE
      ENDIF
      ENDIF
112   CONTINUE
      DO 116 I=1,NEXTUP
      NN=NEXUP(I)
      NNL=LCONL(I)
      NNR=LCONR(I)
```

```
      IF(ICONF(NN).EQ.3)GOTO 116
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(I))))**0.6
      IF((ICONF(NNL).EQ.3).AND.(I.NE.1))THEN
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(NNL))))**0.6
      AVQ=(QFP(NNL)+QPP(NNL))/2.0
      QFNL2=DTRR*(HORAIN(J11)/(12.0*3600.0))*(AVQ**0.4)/(ALPHA*0.6)
      BCONDR=1.49*DSQRT(SLOPE(I))*1.67*((HORAIN(J11)/(12.0*3600.0))**0.6
     +7)*BCOND/ROUGH
      QFNL3=DTRR*BCONDR*(AVQ**0.4)/(ALPHA*0.6)
      QFF(NNL)=QFP(NNL)+QFNL2-QFNL3
      ELSE
      ENDIF
      VOL1=DTRR*QFF(NNL)/(COFINT(NN,4)*DDX(NNL))
      AVG=(QFP(I)+QPP(I))/2.0
      IF(AVG.LE.0.0)THEN
      VOL2=0.0
      ELSE
      VOL2=ALPHA*0.6*QFP(I)*(AVG**(-0.4))
      ENDIF
      VOL3=DTRR*FLOW(I)
      VVOL=VOL1+VOL2+VOL3
      IF(AVG.LE.0.0)THEN
      DELAX=DTRR/(COFINT(NN,4)*DDX(NNL))
      ELSE
      DELAX=DTRR/(COFINT(NN,4)*DDX(NNL))+ALPHA*0.6*(AVG**(-0.4))
      ENDIF
      QFF(I)=VVOL/DELAX
      IF(QFF(I).LT.0.0)QFF(I)=0.0
      IF(J11.EQ.24)QFB(I)=QFF(I)
 116  CONTINUE
      IK=NEXTUP+1
 170  IK=IK-1
      NN=NEXUP(IK)
      NNR=LCONR(IK)
      IF(ICONF(NN).NE.3)GOTO 171
      IF((ICONF(NNR).NE.3).AND.(IK.NE.NEXTUP))THEN
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(NNR))))**0.6
      AVQ=(QFP(NNR)+QPP(NNR))/2.0
      QFNL2=DTRR*(HORAIN(J11)/(12.0*3600.0))*(AVQ**0.4)/(ALPHA*0.6)
      BCONDR=1.49*DSQRT(SLOPE(I))*1.67*((HORAIN(J11)/(12.0*3600.0))**0.6
     +7)*BCOND/ROUGH
      QFNL3=DTRR*BCONDR*(AVQ**0.4)/(ALPHA*0.6)
      QFF(NNR)=QFP(NNR)+QFNL2-QFNL3
      ELSE
      ENDIF
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(IK))))**0.6
      VOL1=DTRR*QFF(NNR)/(COFINT(NN,2)*DDX(NNR))
      AVG=(QFP(IK)+QPP(IK))/2.0
      IF(AVG.LE.0.0)THEN
      VOL2=0.0
```

```
      ELSE
      VOL2=ALPHA*0.6*QFP(IK)*(AVG**(-0.4))
      ENDIF
      VOL3=DTRR*FLOW(IK)
      VVOL=VOL1+VOL2+VOL3
      IF(AVG.LE.0.0)THEN
      DELAX=DTRR/(COFINT(NN,2)*DDX(NNR))
      ELSE
      DELAX=DTRR/(COFINT(NN,2)*DDX(NNR))+ALPHA*0.6*(AVG**(-0.4))
      ENDIF
      QFF(IK)=VVOL/DELAX
      IF(QFF(IK).LT.0.0)QFF(IK)=0.0
      IF(J11.EQ.24)QFB(IK)=QFF(IK)
171   IF(IK.LT.1)GOTO 170
      QHF=0.0
      FFL=0.0
      DO 167 I=1,NEXTUP
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(I))))**0.6
      HFF(I)=(QFF(I)**0.6)*ALPHA
      NN=NEXUP(I)
      THETDF=THETAS-THETAP(NN)
      XX=ATAN(SLOPE(I))
      COSX=COS(XX)
      HDF=(HFF(I)+HC/12.0)*COSX
      SORP=COSX*DSQRT(2.0*0.2835*THETDF*HDF/(24.0*3600.0))
      CAPINL=0.5*SORP/(DSQRT(TT))
      IF((HORAIN(J11)/(12.0*3600.0)).LE.CAPINL)THEN
      FFLOW=0.0
      ELSE
      FFLOW=HORAIN(J11)/(12.0*3600.0)-CAPINL
      ENDIF
      FFL=FFL+FFLOW
      QHF=QHF+QFF(I)
167   CONTINUE
      QHF=QHF/NEXTUP
      FFL=FFL/NEXTUP
      OUT=QHF*DTRR
      IF(HORAIN(J11).GT.0.0)OUTFLOW=OUTFLOW+OUT
      HOFLOW=HOFLOW+OUT
      HDEP=HDEP+FFL*12.0*DTRR
      FLONET=FLONET+FFL*DTRR*12.0
      HRON=HRON+DTRR*12.0*HORAIN(J11)/(12.0*3600.0)
      DO 172 I=1,NEXTUP
      QPP(I)=QFP(I)
172   CONTINUE
      IF(TTR.LT.3600.0)GOTO 142
      AVFLOW=HOFLOW/3600.0
137   AVINFI=0.0
      AVRUN=0.0
121   DO 156 I=1,NEXTUP
```

```
      NN=NEXUP(I)
      AVINFI=AVINFI+PINFIL(I)
      ALPHA=(ROUGH/(1.49*SQRT(SLOPE(I))))**0.6
      HF(I,J11)=(AVFLOW**0.6)*ALPHA*12.0
      XM=5.0/3.0
      TOTRUN(I,J11)=1.49*((HF(I,J11)/12.0)**XM)*SQRT(SLOPE(I))/ROUGH
  156 CONTINUE
      AVINFI=AVINFI/NEXTUP
      AVRUN=AVRUN/NEXTUP
      NOR=NOR+1
      TAVRUN=TAVRUN+AVRUN
      TAVINF=TAVINF+AVINFI
      TOTDEP=TOTDEP+HDEP
      TOTFLOW=TOTFLOW+HOFLOW
      HHRON=HHRON+HORAIN(J11)
      IF(TR.LT.24.0)GOTO 140
      DEPTH=12.0*TOTFLOW/TOTLEN
      IF(DEPTH.GT.1.0)WRITE(3,*)'DEPTH=',DEPTH,'J1=',J1,'P=',PRECIP(II,J
     +1)
      TRUN=0.0
      DO 111 I=1,NEXTUP
      HR(I)=0.0
      N1(I)=0
      DO 117 J=1,24
      HR(I)=HR(I)+HF(I,J)
      IF(HR(I).NE.0.0)N1(I)=N1(I)+1
  117 CONTINUE
      IF(N1(I).EQ.0)N1(I)=1
      HR(I)=HR(I)/N1(I)
      TRUN=TRUN+HR(I)
  111 CONTINUE
      IF(NOR.EQ.0)NOR=1
      TRUN=TRUN/NEXTUP
      TAVRUN=TOTDEP
      RETURN
      END




C
C  THIS SUBROUTINE USES GAUSS-SEIDEL ITERATION TO
C  SOLVE THE UNSATURATED MOISTURE-FLOW EQUATION
C
      SUBROUTINEFDMITER(THETIN,SLOPMC,HCONS,SAIE,THETAS,LINKNT,LINKNR,L
     +INKNB,LINKNL,ICONF,THETAF,DT,DX,DZ,J1,NONODE,NEXTUP,NEXUP,II,NODE,
     +PCP,HR,TEMPER,RAINN,SOLRDD,IGRRO,XXLAI,WCFF,TAVRUN,THETAP,COFINT,M
     +OND)
      IMPLICIT REAL*8(A-H,O-Z)
```

```
      COMMON /AET/ IT,PT,EVAS,IN,SNOP,ESS
      COMMON /GAUMAIN/ ALBEDO,PSYCRO,AS,WP,FC,BC,SNOWP,ACEVAT,DIFFUS,
     +HCON
      PARAMETER (NOD=100,NEX=50,NO=100)
      REAL *8 DIFFUS(NOD,4),THETAP(NOD),THETPP(NOD),THETAF(NOD),BC(NEX),
     +HCON(NOD,4),HR(NEX),THETAB(NOD),COFINT(NOD,4)
      INTEGER LINKNT(NOD),LINKNB(NOD),LINKNR(NOD),LINKNL(NOD),ICONF(NOD)
     +,NEXUP(NEX),NODE(NO),MOND(NOD)
      EXTERNAL EVAT
      CONV=0.0
      DO 30 I=1,NONODE
      NN=NODE(I)
      NNT=LINKNT(NN)
      IF((II.EQ.1).AND.(J1.EQ.1))THEN
      IF(NNT.NE.0)THEN
      IF((MOND(NN).EQ.1).AND.(MOND(NNT).EQ.0))THETAP(I)=THETIN
      IF((MOND(NN).EQ.1).AND.(MOND(NNT).EQ.1))THETAP(I)=THETAS
      IF((MOND(NN).EQ.0).AND.(MOND(NNT).EQ.0))THETAP(I)=THETIN
      ELSE
      IF(NNT.EQ.0)THEN
      IF(MOND(NN).EQ.1)THETAP(I)=THETAS
      IF(MOND(NN).EQ.0)THETAP(I)=THETIN
      ELSE
      ENDIF
      ENDIF
      THETPP(I)=THETAP(I)
      ELSE
      IF(NNT.NE.0)THEN
      IF((MOND(NN).EQ.1).AND.(MOND(NNT).EQ.1))THETPP(I)=THETAS
      IF((MOND(NN).EQ.1).AND.(MOND(NNT).EQ.0))THETPP(I)=THETAP(I)
      IF((MOND(NN).EQ.0).AND.(MOND(NNT).EQ.0))THETPP(I)=THETAP(I)
      IF((MOND(NN).EQ.1).AND.(MOND(NNT).EQ.1))THETAP(I)=THETAS
      IF((MOND(NN).EQ.1).AND.(MOND(NNT).EQ.0))THETAP(I)=THETAF(I)
      IF((MOND(NN).EQ.0).AND.(MOND(NNT).EQ.0))THETAP(I)=THETAF(I)
      ELSE
      IF(NNT.EQ.0)THEN
      IF(MOND(NN).EQ.1)THETPP(I)=THETAS
      IF(MOND(NN).EQ.1)THETAP(I)=THETAS
      IF(MOND(NN).EQ.0)THETPP(I)=THETAP(I)
      IF(MOND(NN).EQ.0)THETAP(I)=THETAF(I)
      ELSE
      ENDIF
      ENDIF
      ENDIF
   30 CONTINUE
      EX1=2.0*SLOPMC+3.0
      EX2=SLOPMC+2.0
      EX3=SLOPMC*HCONS*SAIE/THETAS
      DO 50 I=1,NONODE
      NN=NODE(I)
```

```
      NNT=LINKNT(NN)
      IF(NNT.EQ.0)THEN
      TP1=(THETAP(I)+THETPP(I))/2.0
      ELSE
      DO 51 J=1,NONODE
      IF(NODE(J).EQ.NNT)NT=J
 51   CONTINUE
      TP1=(THETAP(I)+THETPP(I)+THETAP(NT)+THETPP(NT))/4.0
      ENDIF
      HCON(I,1)=HCONS*(TP1/THETAS)**EX1
      DIFFUS(I,1)=-EX3*(TP1/THETAS)**EX2
      NNR=LINKNR(NN)
      IF(NNR.EQ.0)THEN
      TP2=(THETAP(I)+THETPP(I))/2.0
      ELSE
      DO 52 J=1,NONODE
      IF(NODE(J).EQ.NNR)NR=J
 52   CONTINUE
      TP2=(THETAP(I)+THETPP(I)+THETAP(NR)+THETPP(NR))/4.0
      ENDIF
      HCON(I,2)=HCONS*(TP2/THETAS)**EX1
      DIFFUS(I,2)=-EX3*(TP2/THETAS)**EX2
      NNB=LINKNB(NN)
      IF(ICONF(NNB).EQ.4)THEN
      TP3=(THETAP(I)+THETPP(I))/2.0
      ELSE
      DO 53 J=1,NONODE
      IF(NODE(J).EQ.NNB)NB=J
 53   CONTINUE
      TP3=(THETAP(I)+THETPP(I)+THETAP(NB)+THETPP(NB))/4.0
      ENDIF
      HCON(I,3)=HCONS*(TP3/THETAS)**EX1
      DIFFUS(I,3)=-EX3*(TP3/THETAS)**EX2
      NNL=LINKNL(NN)
      IF(NNL.EQ.0)THEN
      TP4=(THETAP(I)+THETPP(I))/2.0
      ELSE
      DO 54 J=1,NONODE
      IF(NODE(J).EQ.NNL)NL=J
 54   CONTINUE
      TP4=(THETAP(I)+THETPP(I)+THETAP(NL)+THETPP(NL))/4.0
      ENDIF
      HCON(I,4)=HCONS*(TP4/THETAS)**EX1
      DIFFUS(I,4)=-EX3*(TP4/THETAS)**EX2
 50   CONTINUE
      IF(J1.EQ.1)SNOP=SNOWP
      IF(J1.EQ.1)ESS=0.0
      IF(J1.EQ.1)EDIFF=0.0
      THETA=0.0
      DO 61 I=1,NEXTUP
```

```
      NN=NEXUP(I)
      DO 62 J=1,NONODE
      IF(NODE(J).EQ.NN)NU=J
 62   CONTINUE
      THETA=THETA+THETAP(NU)
 61   CONTINUE
      THETA=THETA/NEXTUP
      CALL EVAT(EDIFF,TEMPER,ALBEDO,SOLRDD,PSYCRO,IGRRO,XXLAI,WCFF,AS,TH
     +ETA,WP,FC,RAINN,TAVRUN,TOTMET,ACEVAT,J1)
      DO 56 I=1,NEXTUP
      NN=NEXUP(I)
      DO 57 J=1,NONODE
      IF(NODE(J).EQ.NN)NU1=J
 57   CONTINUE
      BC(I)=(RAINN+TOTMET-TAVRUN-ACEVAT)/12.0
      IF(TEMPER.LE.32.0)BC(I)=-ACEVAT/12.0
      IF(BC(I).GT.0.5)THEN
      ELSE
      ENDIF
 56   CONTINUE
      DO 21 I=1,NONODE
      THETAF(I)=THETAP(I)
 21   CONTINUE
      ITER=0
 20   TDIFF=0.0
      ITER=ITER+1
      DO 35 I=1,NONODE
      THETAB(I)=THETAF(I)
 35   CONTINUE
      DO 45 I=1,NONODE
      VOL=(THETAF(I)-THETAP(I))/DT
      NN=NODE(I)
      NNR=LINKNR(NN)
      NNL=LINKNL(NN)
      NNT=LINKNT(NN)
      NNB=LINKNB(NN)
      IF(NNR.NE.0)THEN
      DO 46 J=1,NONODE
      IF(NODE(J).EQ.NNR)NR=J
 46   CONTINUE
      ELSE
      ENDIF
      IF(NNL.NE.0)THEN
      DO 47 J=1,NONODE
      IF(NODE(J).EQ.NNL)NL=J
 47   CONTINUE
      ELSE
      ENDIF
      IF(NNT.NE.0)THEN
      DO 48 J=1,NONODE
```

```
      IF(NODE(J).EQ.NNT)NT=J
48    CONTINUE
      ELSE
      ENDIF
      IF(NNB.NE.0)THEN
      DO 49 J=1,NONODE
      IF(NODE(J).EQ.NNB)NB=J
49    CONTINUE
      ELSE
      ENDIF
      IF((ICONF(NN).EQ.1).OR.(ICONF(NN).EQ.2).OR.(ICONF(NN).EQ.3))THEN
      DO 163 J=1,NEXTUP
      IF(NEXUP(J).EQ.NN)NU=J
163   CONTINUE
      ELSE
      ENDIF
      IF(ICONF(NN).EQ.1)THEN
      IF(CONV.EQ.1.0)THETAF(NB)=THETAF(I)
      IF((ICONF(NNB).EQ.4).OR.(MOND(NNB).EQ.1))THETAF(NB)=THETAF(I)
      FLUXZB=DIFFUS(I,3)*(THETAF(NB)-THETAF(I))
      FLUXX=0.0
      FLUXZ=FLUXZB/(COFINT(NN,3)*DZ)
      GRAVIT=HCON(I,3)/(COFINT(NN,3)*DZ)
      BFLUX=BC(NU)/(COFINT(NN,3)*DZ)
      DELAXZ=DIFFUS(I,3)/((COFINT(NN,3)*DZ)**2)
      DELAX=1.0/DT+DELAXZ
      RES=-VOL+FLUXZ-GRAVIT+BFLUX
      ELSE
      IF(ICONF(NN).EQ.3)THEN
      NNR=LINKNR(NN)
      IF((LINKNB(NN).EQ.0).OR.(ICONF(NNB).EQ.4).OR.(CONV.EQ.1.0).OR.(MON
     +D(NNB).EQ.1))THEN
      THNB=THETAF(I)
      THNP=THETAP(I)
      ELSE
      THNB=THETAF(NB)
      THNP=THETAP(NB)
      ENDIF
      IF(CONV.EQ.1.0)THETAF(NR)=THETAF(I)
      IF((ICONF(NNR).EQ.4).OR.(MOND(NNR).EQ.1))THETAF(NR)=THETAF(I)
      DELAXZ=DIFFUS(I,3)/((COFINT(NN,3)*DZ)**2)
      BFLUX=BC(NU)/(COFINT(NN,3)*DZ)
      FLUXZB=DIFFUS(I,3)*(THNB-THETAF(I))
      FLUXXR=DIFFUS(I,2)*(THETAF(NR)-THETAF(I))
      FLUXX=FLUXXR/((COFINT(NN,2)*DX)**2)
      FLUXZ=FLUXZB/((COFINT(NN,3)*DZ)**2)
      DELAXX=DIFFUS(I,2)/((COFINT(NN,2)*DX)**2)
      GRAVIT=HCON(I,3)/(COFINT(NN,3)*DZ)
      DELAX=1.0/DT+DELAXX+DELAXZ
      RES=-VOL+FLUXX+FLUXZ+BFLUX-GRAVIT
```

```
ELSE
IF(ICONF(NN).EQ.2)THEN
NNL=LINKNL(NN)
IF((LINKNB(NN).EQ.0).OR.(ICONF(NNB).EQ.4).OR.(CONV.EQ.1.0).OR.(MON
+D(NNB).EQ.1))THEN
THNB=THETAF(I)
THNP=THETAP(I)
ELSE
THNB=THETAF(NB)
THNP=THETAP(NB)
ENDIF
IF(CONV.EQ.1.0)THETAF(NL)=THETAF(I)
IF((ICONF(NNL).EQ.4).OR.(MOND(NNL).EQ.1))THETAF(NL)=THETAF(I)
BFLUX=BC(NU)/(COFINT(NN,3)*DZ)
FLUXZB=DIFFUS(I,3)*(THNB-THETAF(I))
DELAXZ=DIFFUS(I,3)/((COFINT(NN,3)*DZ)**2)
FLUXXL=DIFFUS(I,4)*(THETAF(NL)-THETAF(I))
FLUXX=FLUXXL/((COFINT(NN,4)*DX)**2)
FLUXZ=FLUXZB/((COFINT(NN,3)*DZ)**2)
DELAXX=DIFFUS(I,4)/((COFINT(NN,4)*DX)**2)
DELAX=1.0/DT+DELAXZ+DELAXX
GRAVIT=HCON(I,3)/(COFINT(NN,3)*DZ)
RES=-VOL+FLUXX+FLUXZ+BFLUX-GRAVIT
ELSE
IF(ICONF(NN).EQ.5)THEN
IF((ICONF(NNB).EQ.4).OR.(MOND(NNB).EQ.1))THETAF(NB)=THETAF(I)
IF((ICONF(NNR).EQ.4).OR.(MOND(NNR).EQ.1))THETAF(NR)=THETAF(I)
IF((ICONF(NNL).EQ.4).OR.(MOND(NNL).EQ.1))THETAF(NL)=THETAF(I)
FLUXXR=DIFFUS(I,2)*(THETAF(NR)-THETAF(I))/(COFINT(NN,2)*DX)
FLUXXL=DIFFUS(I,4)*(THETAF(I)-THETAF(NL))/(COFINT(NN,4)*DX)
FLUXX=2.0*(FLUXXR-FLUXXL)/((COFINT(NN,2)+COFINT(NN,4))*DX)
FLUXZT=DIFFUS(I,1)*(THETAF(NT)-THETAF(I))/(COFINT(NN,1)*DZ)
FLUXZB=DIFFUS(I,3)*(THETAF(I)-THETAF(NB))/(COFINT(NN,3)*DZ)
FLUXZ=2.0*(FLUXZT-FLUXZB)/((COFINT(NN,1)+COFINT(NN,3))*DZ)
GRAVIT=2.0*(HCON(I,1)-HCON(I,3))/((COFINT(NN,1)+COFINT(NN,3))*DZ)
DELX1=DIFFUS(I,2)/(COFINT(NN,2)*DX)
DELX2=DIFFUS(I,4)/(COFINT(NN,4)*DX)
DELAXX=2.0*(DELX1+DELX2)/(DX*(COFINT(NN,2)+COFINT(NN,4)))
DELZ1=DIFFUS(I,1)/(COFINT(NN,1)*DZ)
DELZ2=DIFFUS(I,3)/(COFINT(NN,3)*DZ)
DELAXZ=2.0*(DELZ1+DELZ2)/(DZ*(COFINT(NN,1)+COFINT(NN,3)))
DELAX=1.0/DT+DELAXX+DELAXZ
RES=-VOL+FLUXX+FLUXZ-GRAVIT
ELSE
ENDIF
ENDIF
ENDIF
ENDIF
RELAX=1.0/DELAX
THETAF(I)=THETAF(I)+RES*RELAX
```

```
 45  CONTINUE
     DO 60 I=1,NONODE
     IF(THETAF(I).GT.THETAS)THETAF(I)=THETAS
     IF(THETAF(I).LT.FC)THETAF(I)=FC
 60  CONTINUE
     DO 90 I=1,NONODE
     TDIFF=TDIFF+ABS(THETAF(I)-THETAB(I))
 90  CONTINUE
     IF(ITER.EQ.1)THEN
     BDIFF=TDIFF
     ELSE
     ENDIF
     CONV=0.0
     IF(ITER.GT.150)WRITE(*,*)'TOO MANY ITERATIONS IN G.S.I. METHOD'
     IF((TDIFF.GT.BDIFF).OR.(ITER.GT.20))THEN
     CONV=1.0
     ELSE
     BDIFF=TDIFF
     ENDIF
     IF(TDIFF.GT.0.04)GOTO 20
     IF(IN.EQ.1)PT=IT
     RETURN
     END
```

```
C
C THIS SUBROUTINE COMPUTES THE COEFFICIENT MATRICES
C TO SOLVE THE SATURATED FLOW EQUATION
C
      SUBROUTINE SATELIM(NODTOT,NOBAR,NOROW,NOCOL,ICRNO,ICONF,IBET,
     +LINKNB,NEXTUP,NEXUP,THETAS,BC,HDIN,ICOL,LCONL,LCONR,HCONS,
     +CONDB,EFPORO,BCONL,BCONR,DT,DX,DZ,THICK,TAVRUN,ACEVAT,
     +PRCPP,NODE,NONODE,SLOPBR,THETAF,II,J1,GRAD,THETAM,MOND,
     +LINKNT,HDD,HDP,TAVSOR,HCOL,SLOPMC,SAIE,THETAP,VDISTB,HCON,
     +DIFFUS,FC,LINKNR,LINKNL,DRC)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /COFMN/ COEFM
      PARAMETER (NC=50,NT=100,NEX=50,NR=20)
      INTEGER LCONL(NC),LCONR(NC),ICOL(NC),IBET(NT),ICRNO(NR,NC),ICONF(N
     +T),NEXUP(NEX),MOND(NT),NUM(NC),LINKNB(NT),NODE(NT),LINKNT(NT),NODU
     +N(NC),NODCOL(NR,NC),IDN(NC),LINKNL(NT),LINKNR(NT)
      REAL *8 THICK(NC),HDIN(NC),SOURCE(NC),HDP(NC),HPP(NC),HDD(NC),HPS(
     +NC,2),COEFM(NT,NT),SLOPBR(NC),HCOL(1,NEX),THETAM(NT),THETAF(NT),BC
     +(NEX),GRAD(1,NC),SOURCHK(NC),THETAP(NT),VDISTB(NT,1),HCON(NT,4),DI
     +FFUS(NT,4),DRAIN(NC)
      EXTERNAL ELIM,SOURCH
      SLOPL=0.0
```

```
      SLOPR=0.0
      DO 12 I=1,NOCOL/2
      SLOPL=SLOPL+SLOPBR(I)
 12   CONTINUE
      SLOPL=SLOPL*2/NOCOL
      DO 20 I=NOCOL/2,NOCOL
      SLOPR=SLOPR+SLOPBR(I)
 20   CONTINUE
      SLOPR=SLOPR*2/NOCOL
      DO 35 I=1,NOCOL
      IF((II.EQ.1).AND.(J1.EQ.1))THEN
      HDP(I)=HDIN(I)
      HPP(I)=HDP(I)
      ELSE
      HPP(I)=HDP(I)
      HDP(I)=HDD(I)
      ENDIF
 35   CONTINUE
      TAVSOR=0.0
      DO 426 I=1,NOCOL
      SOURCE(I)=0.0
      SOURCHK(I)=0.0
      NUM(I)=0
426   CONTINUE
      HHDP=0.0
      HHDPP=0.0
      DO 427 I=1,NOCOL
      HHDP=HHDP+HDP(I)
      HHDPP=HHDPP+HPP(I)
427   CONTINUE
      CALL SOURCH(NOROW,NOCOL,MOND,THETAF,THETAP,VDISTB,SOURCHK,ICRNO,
     +NONODE,NODE,FC,DRAIN,NODUN,NODCOL,BC,LINKNB,DIFFUS,DX,LINKNR,
     +LINKNL,DZ,THETAS,SAIE,HCONS,DRC)
      DO 575 I=1,NOROW
      DO 575 J=1,NOCOL
      NN=ICRNO(I,J)
      IF(NN.EQ.0)GOTO 575
      DO 430 K=1,NONODE
      IF(NODE(K).EQ.NN)N1=K
430   CONTINUE
      IF(MOND(NN).EQ.1)GOTO 575
      NNT=LINKNT(NN)
      NNB=LINKNB(NN)
      IF(NNB.EQ.0)THEN
      THNB=THETAF(N1)
      ELSE
      IF(ICONF(NNB).NE.4)THEN
      DO 431 K=1,NONODE
      IF(NODE(K).EQ.NNB)NB1=K
431   CONTINUE
```

```
      THNB=THETAF(NB1)
      ELSE
      THNB=THETAF(N1)
      ENDIF
      ENDIF
      SOURCE(J)=SOURCE(J)+HCON(N1,3)-DIFFUS(N1,3)*(THNB-THETAF(N1))/DZ
      NUM(J)=NUM(J)+1
575   CONTINUE
      DO 580 I=1,NOROW
      DO 580 J=1,NOCOL
      NN=ICRNO(I,J)
      IF(NN.EQ.0)GOTO 580
      IF(ICONF(NN).EQ.4)GOTO 580
      IF(ICONF(NN).EQ.5)GOTO 580
      IF(NUM(J).EQ.0)THEN
      DO 434 K=1,NEXTUP
      IF(NEXUP(K).EQ.NN)NU=K
434   CONTINUE
      IF(BC(NU).GT.HCON(NU,1))THEN
      SOURCE(J)=SOURCE(J)+HCON(NU,1)
      NUM(J)=NUM(J)+1
      ELSE
      SOURCE(J)=SOURCE(J)+BC(NU)
      NUM(J)=NUM(J)+1
      ENDIF
      ELSE
      GOTO 580
      ENDIF
580   CONTINUE
      TAVIN=(PRCPP-TAVRUN-ACEVAT)/12.0
433   DO 432 I=1,NOCOL
      SOURCE(I)=SOURCE(I)/NUM(I)
      IF(SOURCE(I).LT.0.0)SOURCE(I)=0.0
      IF(SOURCE(I).GT.(0.1*PRCPP/12.0))SOURCE(I)=SOURCHK(I)
      IF(SOURCE(I).LT.(0.05*PRCPP/12.0))SOURCE(I)=DRAIN(I)
      TAVSOR=TAVSOR+SOURCE(I)
432   CONTINUE
      TAVSOR=12.0*TAVSOR/NOCOL
      DO 10 I=1,NOCOL
      NNL=LCONL(I)
      NNR=LCONR(I)
      IF(ICOL(NNL).EQ.2)THEN
      HPS(I,2)=(HDP(I)+HPP(I))/2.0
      HPS(I,1)=(HDP(I)+HPP(I)+HDP(NNR)+HPP(NNR))/4.0
      ELSE
      IF(ICOL(NNR).EQ.2)THEN
      HPS(I,1)=(HDP(I)+HPP(I))/2.0
      HPS(I,2)=(HDP(I)+HPP(I)+HDP(NNL)+HPP(NNL))/4.0
      ELSE
      HPS(I,1)=(HDP(I)+HPP(I)+HDP(NNR)+HPP(NNR))/4.0
```

```
      HPS(I,2)=(HDP(I)+HPP(I)+HDP(NNL)+HPP(NNL))/4.0
      ENDIF
      ENDIF
10    CONTINUE
      DO 45 I=1,NOCOL
      DO 45 J=1,NOCOL
      COEFM(I,J)=0.0
45    CONTINUE
      DO 15 I=1,NOCOL
      IF(I.LT.(NOCOL/2))SLOPB=SLOPL
      IF(I.GE.(NOCOL/2))SLOPB=SLOPR
      NNL=LCONL(I)
      NNR=LCONR(I)
      IF(ICOL(NNL).EQ.2)THEN
      A=(-HCONS*HPS(I,1))/DX**2-CONDB/THICK(I)-EFPORO/DT
      B=0.0
      C=HCONS*HPS(I,1)/DX**2
      P=-SOURCE(I)+CONDB-EFPORO*HDP(I)/DT+HCONS*HPS(I,2)*BCONL/DX-HCONS*
     +SLOPB*BCONL
      COEFM(I,I)=A
      COEFM(I,NNR)=C
      HDD(I)=P
      ELSE
      IF(ICOL(NNR).EQ.2)THEN
      A=(-HCONS*HPS(I,2))/DX**2-CONDB/THICK(I)-EFPORO/DT
      B=HCONS*HPS(I,2)/DX**2
      C=0.0
      P=-SOURCE(I)+CONDB-EFPORO*HDP(I)/DT-HCONS*HPS(I,1)*BCONR/DX-HCONS*
     +SLOPB*BCONR
      COEFM(I,I)=A
      COEFM(I,NNL)=B
      HDD(I)=P
      ELSE
      IF((ICOL(NNL).EQ.1).AND.(ICOL(NNR).EQ.1))THEN
      A=HCONS*(-HPS(I,1)-HPS(I,2))/DX**2-CONDB/THICK(I)-EFPORO/DT
      B=HCONS*HPS(I,2)/DX**2-HCONS*SLOPB/(2.0*DX)
      C=HCONS*HPS(I,1)/DX**2+HCONS*SLOPB/(2.0*DX)
      P=-SOURCE(I)+CONDB-EFPORO*HDP(I)/DT
      COEFM(I,I)=A
      COEFM(I,NNL)=B
      COEFM(I,NNR)=C
      HDD(I)=P
      ELSE
      ENDIF
      ENDIF
      ENDIF
15    CONTINUE
      CALL ELIM(HDD,NOCOL)
      DO 991 I=1,NOBAR
      DO 991 J=1,NOCOL
```

```
      NNL=LCONL(J)
      NNR=LCONR(J)
      IF(HDD(J).LT.0.0)HDD(J)=0.0
      IF(HDD(J).GT.HCOL(I,J))HDD(J)=HCOL(I,J)
      GRAD(I,J)=ABS(HDD(NNL)-HDD(NNR))/(2.0*DX)
 991  CONTINUE
      RETURN
      END




C
C THIS SUBROUTINE COMPUTES LEACHATE ACCRETION IN THE
C UNSATURATED ZONE
C
      SUBROUTINE SOURCH(NOROW,NOCOL,MOND,THETAF,THETAP,VDISTB,
     +SOURCHK,ICRNO,NONODE,NODE,FC,DRAIN,NODUN,NODCOL,BC,LINKNB,
     +DIFFUS,DX,LINKNR,LINKNL,DZ,THETAS,SAIE,HCONS,DRC)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NT=100,NC=50,NR=20)
      REAL *8 VDISTB(NT,1),THETAF(NT),THETAP(NT),SOURCHK(NC),SOLRIGHT(NR
     +),CCF(NR,NR),CCP(NR,NR),DRAIN(NC),BC(NC),STORE(NC,NR),DIFFUS(NT,4)
      INTEGER MOND(NT),NODUN(NC),NODCOL(NC,NR),ICRNO(NR,NC),NODE(NT),LIN
     +KNB(NT),LINKNR(NT),LINKNL(NT)
      DO 30 I=1,NOCOL
      NODUN(I)=0
 30   CONTINUE
      DO 10 I=1,NOROW
      DO 10 J=1,NOCOL
      NN=ICRNO(I,J)
      IF(NN.EQ.0)GOTO 10
      IF(MOND(NN).NE.1)THEN
      NODUN(J)=NODUN(J)+1
      NODCOL(J,NODUN(J))=NN
      ELSE
      ENDIF
 10   CONTINUE
      DO 40 I=1,NOCOL
      NOUN=NODUN(I)
      SOURCHK(I)=0.0
      DO 20 J=1,NOUN
      NN1=NODCOL(I,J)
      DO 432 IJ=1,NONODE
      IF(NODE(IJ).EQ.NN1)NI1=IJ
 432  CONTINUE
      STORPOT=THETAS-THETAP(NI1)
      IF(THETAP(NI1).GT.FC)THEN
      GRAV=THETAS-FC
      SOURCHK(I)=SOURCHK(I)+DRC*(1.0-STORPOT/GRAV)**3
```

```
      ELSE
      DCHK=HCONS*DABS(SAIE)*(THETAP(NI1)**4.50)
      SOURCHK(I)=SOURCHK(I)+DCHK
      ENDIF
 20   CONTINUE
      SOURCHK(I)=SOURCHK(I)/NOUN
 40   CONTINUE
      DO 80 I=1,NOCOL
      NOUN=NODUN(I)
      DRAN=0.0
      DO 80 J=1,NOUN
      IF(J.EQ.NOUN)GOTO 80
      NN1=NODCOL(I,J)
      NN2=NODCOL(I,J+1)
      VVDIST=VDISTB(NN2,1)-VDISTB(NN1,1)
      DO 430 IJ=1,NONODE
      IF(NODE(IJ).EQ.NN1)NI1=IJ
      IF(NODE(IJ).EQ.NN2)NI2=IJ
 430  CONTINUE
      AREAF=0.5*(THETAF(NI1)+THETAF(NI2))*VVDIST
      AREAP=0.5*(THETAP(NI1)+THETAP(NI2))*VVDIST
      STORE(I,J)=AREAF-AREAP
 80   CONTINUE
      DO 50 I=1,NOCOL
      NOUN=NODUN(I)-1
      IF(NOUN.LT.0)THEN
      DRAIN(I)=0.0
      GOTO 50
      ELSE
      ENDIF
      IF(NOUN.EQ.0)THEN
      NN1=NODCOL(I,NODUN(I))
      NN2=LINKNB(NN1)
      NNR=LINKNR(NN1)
      NNL=LINKNL(NN1)
      DO 61 K=1,NONODE
      IF(NODE(K).EQ.NN1)N1=K
      IF(NODE(K).EQ.NNL)NL1=K
      IF(NODE(K).EQ.NNR)NR1=K
 61   CONTINUE
      DR1=(THETAF(N1)-THETAP(N1))*(VDISTB(NN1,1)-VDISTB(NN2,1))/2.
      IF((NNL.EQ.0).OR.(NNR.EQ.0))THEN
      DR2=0.0
      DR3=0.0
      ELSE
      DR2=DIFFUS(N1,2)*(THETAF(NR1)-THETAF(N1))*DZ/DX
      DR3=DIFFUS(N1,4)*(THETAF(NL1)-THETAF(N1))*DZ/DX
      ENDIF
      DR= BC(I)-DR1-DR2-DR3
      DRAIN(I)=DR
```

```
      GOTO 50
      ELSE
      ENDIF
      J=NOUN
60    IF(J.EQ.NOUN)THEN
      NN=NODCOL(I,J)
      NNR=LINKNR(NN)
      NNL=LINKNL(NN)
      DO 63 K=1,NONODE
      IF(NODE(K).EQ.NN)N1=K
      IF(NODE(K).EQ.NNL)NL1=K
      IF(NODE(K).EQ.NNR)NR1=K
63    CONTINUE
      IF((NNL.EQ.0).OR.(NNR.EQ.0))THEN
      DR2=0.0
      DR3=0.0
      ELSE
      DR2=DIFFUS(N1,2)*(THETAF(NR1)-THETAF(N1))*DZ/DX
      DR3=DIFFUS(N1,4)*(THETAF(NL1)-THETAF(N1))*DZ/DX
      ENDIF
      DR=BC(I)-DR2-DR3-STORE(I,J)
      ELSE
      NN=NODCOL(I,J)
      NNL=LINKNL(NN)
      NNR=LINKNR(NN)
      DO 64 K=1,NONODE
      IF(NODE(K).EQ.NN)N1=K
      IF(NODE(K).EQ.NNL)NL1=K
      IF(NODE(K).EQ.NNR)NR1=K
64    CONTINUE
      IF((NNL.EQ.0).OR.(NNR.EQ.0))THEN
      DR2=0.0
      DR3=0.0
      ELSE
      DR2=DIFFUS(N1,2)*(THETAF(NR1)-THETAF(N1))*DZ/DX
      DR3=DIFFUS(N1,4)*(THETAF(NL1)-THETAF(N1))*DZ/DX
      ENDIF
      DR=DR-DR2-DR3-STORE(I,J)
      ENDIF
      J=J-1
      IF(J.GE.1)GOTO 60
      DRAIN(I)=DR
50    CONTINUE
      DO 70 I=1,NOCOL
      IF((SOURCHK(I)).GT.(DRAIN(I)))SOURCHK(I)=DRAIN(I)
70    CONTINUE
      RETURN
      END
```

```
C
C  THIS SUBROUTINE SOLVES SIMULTANEOUS EQUATIONS
C  USING GAUSS-JORDAN ELIMINATION METHOD
C
      SUBROUTINE ELIM(COLMAT,NOBEL)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER (NT=100)
      COMMON /COFMN/ COEFF
      REAL*8 COEFF(NT,NT),PP(NT),COLMAT(NT)
      DO 947 I=1,NOBEL
      IF(I.EQ.NOBEL)GOTO 956
      BIG=DABS(COEFF(I,I))
      I1=I+1
      DO 951 L=I1,NOBEL
      IF(DABS(COEFF(L,I)).GT.BIG)BIG=DABS(COEFF(L,I))
 951  CONTINUE
      DO 952 L=I,NOBEL
      IF(DABS(COEFF(L,I)).EQ.BIG)NR=L
 952  CONTINUE
      DO 953 M=1,NOBEL
      PP(M)=COEFF(I,M)
 953  CONTINUE
      NON1=NOBEL+1
      PP(NON1)=COLMAT(I)
      DO 954 M=1,NOBEL
      COEFF(I,M)=COEFF(NR,M)
 954  CONTINUE
      COLMAT(I)=COLMAT(NR)
      DO 955 M=1,NOBEL
      COEFF(NR,M)=PP(M)
 955  CONTINUE
      COLMAT(NR)=PP(NON1)
 956  DEN=COEFF(I,I)
      DO 948 J=1,NOBEL
      COEFF(I,J)=COEFF(I,J)/DEN
 948  CONTINUE
      COLMAT(I)=COLMAT(I)/DEN
      DO 949 K=1,NOBEL
      IF(K.EQ.I)GOTO 949
      IF(COEFF(K,I).EQ.0.0)GOTO 949
      CON=COEFF(K,I)
      DO 950 J=1,NOBEL
      IF(DABS(COEFF(I,J)).LT.1.0E-30)THEN
      COEFF(I,J)=1.0E-30
      ELSE
      COEFF(K,J)=COEFF(K,J)-COEFF(I,J)*CON
      ENDIF
 950  CONTINUE
      COLMAT(K)=COLMAT(K)-COLMAT(I)*CON
```

```
 949  CONTINUE
 947  CONTINUE
      RETURN
      END




C
C  THIS SUBROUTINE COMPUTES LATERAL FLOW UPON THE BARRIER
C  AND VERTICAL FLOW THROUGH THE BARRIER
C
      SUBROUTINEFLOW(NOCOL,NOBAR,HDD,SLOPBR,HCONS,J1,II,RAINN,THICK,CON
     +DB)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /MFLO/ AQV,FLOWL,FLOWR,FLOWVV,FLOWLV,FLOWLI,BEFFICY,
     +SMEFICY,EFFICY
      COMMON /MONFO/ GRAD,BIG
      PARAMETER(NT=100,NC=50,NR=50)
      REAL *8 HDD(NC),BIG(1),AVSLPL(1),AVSLPR(1),SLOPB
     +R(NC),FLOWL(1),FLOWR(1),GRAD(1,NC),AVGDL(1),AVGDR(1),FLOWVV(1),FLO
     +WLV(1),FLOWLI(1),BEFFICY(2),SMEFICY(2),AVHDL(1),AVHDR(1),THICK(NC)
      INTEGER MX(1)
      DO 350 I=1,NOBAR
      AVGDR(I)=0.0
      AVGDL(I)=0.0
 350  CONTINUE
      AQV=0.0
      DO 336 I=1,NOCOL
      QV=CONDB*(HDD(I)+THICK(I))/THICK(I)
      AQV=AQV+QV
 336  CONTINUE
      AQV=AQV/NOCOL
      IF(RAINN.NE.0.0)EFFICY=100.0-100.0*AQV/RAINN
      IF(RAINN.EQ.0.0)EFFICY=100.0
      IF(RAINN.NE.0.0)THEN
      IF(J1.EQ.1)THEN
      BEFFICY(II)=EFFICY
      SMEFICY(II)=EFFICY
      ELSE
      IF(EFFICY.GT.BEFFICY(II))BEFFICY(II)=EFFICY
      IF(EFFICY.LT.SMEFICY(II))SMEFICY(II)=EFFICY
      ENDIF
      ELSE
      ENDIF
      DO 505 I=1,NOBAR
      BIG(I)=HDD(1)
      DO 505 J=2,NOCOL
      IF(HDD(J).GT.BIG(I))BIG(I)=HDD(J)
 505  CONTINUE
```

```
      DO 510 I=1,NOBAR
      DO 510 J=1,NOCOL
      IF(HDD(J).EQ.BIG(I))THEN
      MX(I)=J
      ELSE
      GOTO 510
      ENDIF
  510 CONTINUE
      DO 515 I=1,NOBAR
      KN=MX(I)
      DO 520 J=1,KN
      AVHDL(I)=AVHDL(I)+HDD(J)
      AVSLPL(I)=AVSLPL(I)+SLOPBR(J)
      AVGDL(I)=AVGDL(I)+GRAD(I,J)
  520 CONTINUE
      AVHDL(I)=AVHDL(I)/KN
      AVSLPL(I)=AVSLPL(I)/KN
      AVGDL(I)=AVGDL(I)/KN
      DO 525 J=KN,NOCOL
      AVHDR(I)=AVHDR(I)+HDD(J)
      AVSLPR(I)=AVSLPR(I)+SLOPBR(J)
      AVGDR(I)=AVGDR(I)+GRAD(I,J)
  525 CONTINUE
      KN1=NOCOL-KN+1
      AVHDR(I)=AVHDR(I)/KN1
      AVSLPR(I)=AVSLPR(I)/KN1
      AVGDR(I)=AVGDR(I)/KN1
  515 CONTINUE
      DO 540 I=1,NOBAR
  541 FLOWL(I)=HCONS*BIG(I)*(AVGDL(I)+AVSLPL(I))/2.0
      FLOWR(I)=HCONS*BIG(I)*(AVGDR(I)+AVSLPR(I))/2.0
  540 CONTINUE
      RETURN
      END


C
C THIS SUBROUTINE COMPUTES MONTHLY VALUES OF THE FLOW PARAMETERS
C
      SUBROUTINE SELECT(NOBAR,FLOWL,FLOWR,AQV,J1,II,AVTHET,RAIN,AVIN,TAV
     +RUN,ACEVAT,TAVSOR,BIG,BBIG,TAVINF,FLONET,CAPINF,FLORUN)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /SILA/ FLOMLL,FLOMLR,FLOWMV,BIGLAT,SMALAT,AVMOST,RAINM,
     +PERCM,RUNM,EVAM,RECHM,BIGVER,SMAVER
      REAL *8 FLOMLL(1,2,12),FLOMLR(1,2,12),FLOWMV(1,2,12),BIGLAT(1),SMA
     +LAT(1),FLOWL(1),FLOWR(1),AVMOST(1,2,12),RAINM(2,12),PERCM(2
     +,12),RUNM(2,12),EVAM(2,12),RAIN(2,365),RECHM(1,2,12),BIGVER(1),SMA
     +VER(1),BIG(1),BBIG(1,12),CAPINF(2,12),FLORUN(2,12)
```

```
DO 585 I=1,NOBAR
IF(J1.LE.31)THEN
IF(J1.EQ.1)BBIG(I,1)=0.0
FLOMLL(I,II,1)=FLOMLL(I,II,1)+FLOWL(I)
FLOMLR(I,II,1)=FLOMLR(I,II,1)+FLOWR(I)
FLOWMV(I,II,1)=FLOWMV(I,II,1)+AQV
AVMOST(I,II,1)=AVMOST(I,II,1)+AVTHET
RECHM(I,II,1)=RECHM(I,II,1)+TAVSOR
IF(BIG(I).GT.BBIG(I,1))BBIG(I,1)=BIG(I)
ELSE
IF((J1.GT.31).AND.(J1.LE.59))THEN
IF(J1.EQ.32)BBIG(I,2)=0.0
IF(J1.EQ.32)AVMOST(I,II,1)=AVMOST(I,II,1)/31.0
FLOMLL(I,II,2)=FLOMLL(I,II,2)+FLOWL(I)
FLOMLR(I,II,2)=FLOMLR(I,II,2)+FLOWR(I)
FLOWMV(I,II,2)=FLOWMV(I,II,2)+AQV
AVMOST(I,II,2)=AVMOST(I,II,2)+AVTHET
RECHM(I,II,2)=RECHM(I,II,2)+TAVSOR
IF(BIG(I).GT.BBIG(I,2))BBIG(I,2)=BIG(I)
ELSE
IF((J1.GT.59).AND.(J1.LE.90))THEN
IF(J1.EQ.60)BBIG(I,3)=0.0
IF(J1.EQ.60)AVMOST(I,II,2)=AVMOST(I,II,2)/28.0
FLOMLL(I,II,3)=FLOMLL(I,II,3)+FLOWL(I)
FLOMLR(I,II,3)=FLOMLR(I,II,3)+FLOWR(I)
FLOWMV(I,II,3)=FLOWMV(I,II,3)+AQV
AVMOST(I,II,3)=AVMOST(I,II,3)+AVTHET
RECHM(I,II,3)=RECHM(I,II,3)+TAVSOR
IF(BIG(I).GT.BBIG(I,3))BBIG(I,3)=BIG(I)
ELSE
IF((J1.GT.90).AND.(J1.LE.120))THEN
IF(J1.EQ.91)BBIG(I,4)=0.0
IF(J1.EQ.91)AVMOST(I,II,3)=AVMOST(I,II,3)/31.0
FLOMLL(I,II,4)=FLOMLL(I,II,4)+FLOWL(I)
FLOMLR(I,II,4)=FLOMLR(I,II,4)+FLOWR(I)
FLOWMV(I,II,4)=FLOWMV(I,II,4)+AQV
AVMOST(I,II,4)=AVMOST(I,II,4)+AVTHET
RECHM(I,II,4)=RECHM(I,II,4)+TAVSOR
IF(BIG(I).GT.BBIG(I,4))BBIG(I,4)=BIG(I)
ELSE
IF((J1.GT.120).AND.(J1.LE.151))THEN
IF(J1.EQ.121)BBIG(I,5)=0.0
IF(J1.EQ.121)AVMOST(I,II,4)=AVMOST(I,II,4)/30.0
FLOMLL(I,II,5)=FLOMLL(I,II,5)+FLOWL(I)
FLOMLR(I,II,5)=FLOMLR(I,II,5)+FLOWR(I)
FLOWMV(I,II,5)=FLOWMV(I,II,5)+AQV
AVMOST(I,II,5)=AVMOST(I,II,5)+AVTHET
RECHM(I,II,5)=RECHM(I,II,5)+TAVSOR
IF(BIG(I).GT.BBIG(I,5))BBIG(I,5)=BIG(I)
ELSE
```

```
IF((J1.GT.151).AND.(J1.LE.181))THEN
IF(J1.EQ.152)BBIG(I,6)=0.0
IF(J1.EQ.152)AVMOST(I,II,5)=AVMOST(I,II,5)/31.0
FLOMLL(I,II,6)=FLOMLL(I,II,6)+FLOWL(I)
FLOMLR(I,II,6)=FLOMLR(I,II,6)+FLOWR(I)
FLOWMV(I,II,6)=FLOWMV(I,II,6)+AQV
AVMOST(I,II,6)=AVMOST(I,II,6)+AVTHET
RECHM(I,II,6)=RECHM(I,II,6)+TAVSOR
IF(BIG(I).GT.BBIG(I,6))BBIG(I,6)=BIG(I)
ELSE
IF((J1.GT.181).AND.(J1.LE.212))THEN
IF(J1.EQ.182)BBIG(I,7)=0.0
IF(J1.EQ.182)AVMOST(I,II,6)=AVMOST(I,II,6)/30.0
FLOMLL(I,II,7)=FLOMLL(I,II,7)+FLOWL(I)
FLOMLR(I,II,7)=FLOMLR(I,II,7)+FLOWR(I)
FLOWMV(I,II,7)=FLOWMV(I,II,7)+AQV
AVMOST(I,II,7)=AVMOST(I,II,7)+AVTHET
RECHM(I,II,7)=RECHM(I,II,7)+TAVSOR
IF(BIG(I).GT.BBIG(I,7))BBIG(I,7)=BIG(I)
ELSE
IF((J1.GT.212).AND.(J1.LE.243))THEN
IF(J1.EQ.213)BBIG(I,8)=0.0
IF(J1.EQ.213)AVMOST(I,II,7)=AVMOST(I,II,7)/31.0
FLOMLL(I,II,8)=FLOMLL(I,II,8)+FLOWL(I)
FLOMLR(I,II,8)=FLOMLR(I,II,8)+FLOWR(I)
FLOWMV(I,II,8)=FLOWMV(I,II,8)+AQV
AVMOST(I,II,8)=AVMOST(I,II,8)+AVTHET
RECHM(I,II,8)=RECHM(I,II,8)+TAVSOR
IF(BIG(I).GT.BBIG(I,8))BBIG(I,8)=BIG(I)
ELSE
IF((J1.GT.243).AND.(J1.LE.273))THEN
IF(J1.EQ.244)BBIG(I,9)=0.0
IF(J1.EQ.244)AVMOST(I,II,8)=AVMOST(I,II,8)/31.0
FLOMLL(I,II,9)=FLOMLL(I,II,9)+FLOWL(I)
FLOMLR(I,II,9)=FLOMLR(I,II,9)+FLOWR(I)
FLOWMV(I,II,9)=FLOWMV(I,II,9)+AQV
AVMOST(I,II,9)=AVMOST(I,II,9)+AVTHET
RECHM(I,II,9)=RECHM(I,II,9)+TAVSOR
IF(BIG(I).GT.BBIG(I,9))BBIG(I,9)=BIG(I)
ELSE
IF((J1.GT.273).AND.(J1.LE.304))THEN
IF(J1.EQ.274)BBIG(I,10)=0.0
IF(J1.EQ.274)AVMOST(I,II,9)=AVMOST(I,II,9)/30.0
FLOMLL(I,II,10)=FLOMLL(I,II,10)+FLOWL(I)
FLOMLR(I,II,10)=FLOMLR(I,II,10)+FLOWR(I)
FLOWMV(I,II,10)=FLOWMV(I,II,10)+AQV
AVMOST(I,II,10)=AVMOST(I,II,10)+AVTHET
RECHM(I,II,10)=RECHM(I,II,10)+TAVSOR
IF(BIG(I).GT.BBIG(I,10))BBIG(I,10)=BIG(I)
ELSE
```

```
      IF((J1.GT.304).AND.(J1.LE.334))THEN
      IF(J1.EQ.305)BBIG(I,11)=0.0
      IF(J1.EQ.305)AVMOST(I,II,10)=AVMOST(I,II,10)/31.0
      FLOMLL(I,II,11)=FLOMLL(I,II,11)+FLOWL(I)
      FLOMLR(I,II,11)=FLOMLR(I,II,11)+FLOWR(I)
      FLOWMV(I,II,11)=FLOWMV(I,II,11)+AQV
      AVMOST(I,II,11)=AVMOST(I,II,11)+AVTHET
      RECHM(I,II,11)=RECHM(I,II,11)+TAVSOR
      IF(BIG(I).GT.BBIG(I,11))BBIG(I,11)=BIG(I)
      ELSE
      IF(J1.EQ.335)BBIG(I,12)=0.0
      IF(J1.EQ.335)AVMOST(I,II,11)=AVMOST(I,II,11)/30.0
      FLOMLL(I,II,12)=FLOMLL(I,II,12)+FLOWL(I)
      FLOMLR(I,II,12)=FLOMLR(I,II,12)+FLOWR(I)
      FLOWMV(I,II,12)=FLOWMV(I,II,12)+AQV
      AVMOST(I,II,12)=AVMOST(I,II,12)+AVTHET
      RECHM(I,II,12)=RECHM(I,II,12)+TAVSOR
      IF(BIG(I).GT.BBIG(I,12))BBIG(I,12)=BIG(I)
      IF(J1.EQ.365)AVMOST(I,II,12)=AVMOST(I,II,12)/31.0
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
  585 CONTINUE
      DO 595 I=1,NOBAR
      IF((II.EQ.1).AND.(J1.EQ.1))THEN
      BIGLAT(I)=FLOWL(I)
      SMALAT(I)=FLOWL(I)
      BIGVER(I)=AQV
      SMAVER(I)=AQV
      ELSE
      IF(FLOWL(I).GT.BIGLAT(I))BIGLAT(I)=FLOWL(I)
      IF(FLOWL(I).LT.SMALAT(I))SMALAT(I)=FLOWL(I)
      IF(AQV.GT.BIGVER(I))BIGVER(I)=AQV
      IF(AQV.LT.SMAVER(I))SMAVER(I)=AQV
      ENDIF
  595 CONTINUE
      IF(J1.LE.31)THEN
      RAINM(II,1)=RAINM(II,1)+RAIN(II,J1)
      PERCM(II,1)=PERCM(II,1)+AVIN
      RUNM(II,1)=RUNM(II,1)+TAVRUN
      EVAM(II,1)=EVAM(II,1)+ACEVAT
      CAPINF(II,1)=CAPINF(II,1)+TAVINF
```

```
FLORUN(II,1)=FLORUN(II,1)+FLONET
ELSE
IF((J1.GT.31).AND.(J1.LE.59))THEN
RAINM(II,2)=RAINM(II,2)+RAIN(II,J1)
PERCM(II,2)=PERCM(II,2)+AVIN
RUNM(II,2)=RUNM(II,2)+TAVRUN
EVAM(II,2)=EVAM(II,2)+ACEVAT
CAPINF(II,2)=CAPINF(II,2)+TAVINF
FLORUN(II,2)=FLORUN(II,2)+FLONET
ELSE
IF((J1.GT.59).AND.(J1.LE.90))THEN
RAINM(II,3)=RAINM(II,3)+RAIN(II,J1)
PERCM(II,3)=PERCM(II,3)+AVIN
RUNM(II,3)=RUNM(II,3)+TAVRUN
EVAM(II,3)=EVAM(II,3)+ACEVAT
CAPINF(II,3)=CAPINF(II,3)+TAVINF
FLORUN(II,3)=FLORUN(II,3)+FLONET
ELSE
IF((J1.GT.90).AND.(J1.LE.120))THEN
RAINM(II,4)=RAINM(II,4)+RAIN(II,J1)
PERCM(II,4)=PERCM(II,4)+AVIN
RUNM(II,4)=RUNM(II,4)+TAVRUN
EVAM(II,4)=EVAM(II,4)+ACEVAT
CAPINF(II,4)=CAPINF(II,4)+TAVINF
FLORUN(II,4)=FLORUN(II,4)+FLONET
ELSE
IF((J1.GT.120).AND.(J1.LE.151))THEN
RAINM(II,5)=RAINM(II,5)+RAIN(II,J1)
PERCM(II,5)=PERCM(II,5)+AVIN
RUNM(II,5)=RUNM(II,5)+TAVRUN
EVAM(II,5)=EVAM(II,5)+ACEVAT
CAPINF(II,5)=CAPINF(II,5)+TAVINF
FLORUN(II,5)=FLORUN(II,5)+FLONET
ELSE
IF((J1.GT.151).AND.(J1.LE.181))THEN
RAINM(II,6)=RAINM(II,6)+RAIN(II,J1)
PERCM(II,6)=PERCM(II,6)+AVIN
RUNM(II,6)=RUNM(II,6)+TAVRUN
EVAM(II,6)=EVAM(II,6)+ACEVAT
CAPINF(II,6)=CAPINF(II,6)+TAVINF
FLORUN(II,6)=FLORUN(II,6)+FLONET
ELSE
IF((J1.GT.181).AND.(J1.LE.212))THEN
RAINM(II,7)=RAINM(II,7)+RAIN(II,J1)
PERCM(II,7)=PERCM(II,7)+AVIN
RUNM(II,7)=RUNM(II,7)+TAVRUN
EVAM(II,7)=EVAM(II,7)+ACEVAT
CAPINF(II,7)=CAPINF(II,7)+TAVINF
FLORUN(II,7)=FLORUN(II,7)+FLONET
ELSE
```

```
IF((J1.GT.212).AND.(J1.LE.243))THEN
RAINM(II,8)=RAINM(II,8)+RAIN(II,J1)
PERCM(II,8)=PERCM(II,8)+AVIN
RUNM(II,8)=RUNM(II,8)+TAVRUN
EVAM(II,8)=EVAM(II,8)+ACEVAT
CAPINF(II,8)=CAPINF(II,8)+TAVINF
FLORUN(II,8)=FLORUN(II,8)+FLONET
ELSE
IF((J1.GT.243).AND.(J1.LE.273))THEN
RAINM(II,9)=RAINM(II,9)+RAIN(II,J1)
PERCM(II,9)=PERCM(II,9)+AVIN
RUNM(II,9)=RUNM(II,9)+TAVRUN
EVAM(II,9)=EVAM(II,9)+ACEVAT
CAPINF(II,9)=CAPINF(II,9)+TAVINF
FLORUN(II,9)=FLORUN(II,9)+FLONET
ELSE
IF((J1.GT.273).AND.(J1.LE.304))THEN
RAINM(II,10)=RAINM(II,10)+RAIN(II,J1)
PERCM(II,10)=PERCM(II,10)+AVIN
RUNM(II,10)=RUNM(II,10)+TAVRUN
EVAM(II,10)=EVAM(II,10)+ACEVAT
CAPINF(II,10)=CAPINF(II,10)+TAVINF
FLORUN(II,10)=FLORUN(II,10)+FLONET
ELSE
IF((J1.GT.304).AND.(J1.LE.334))THEN
RAINM(II,11)=RAINM(II,11)+RAIN(II,J1)
PERCM(II,11)=PERCM(II,11)+AVIN
RUNM(II,11)=RUNM(II,11)+TAVRUN
EVAM(II,11)=EVAM(II,11)+ACEVAT
CAPINF(II,11)=CAPINF(II,11)+TAVINF
FLORUN(II,11)=FLORUN(II,11)+FLONET
ELSE
RAINM(II,12)=RAINM(II,12)+RAIN(II,J1)
PERCM(II,12)=PERCM(II,12)+AVIN
RUNM(II,12)=RUNM(II,12)+TAVRUN
EVAM(II,12)=EVAM(II,12)+ACEVAT
CAPINF(II,12)=CAPINF(II,12)+TAVINF
FLORUN(II,12)=FLORUN(II,12)+FLONET
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
RETURN
```

```
      END

C
C THIS SUBROUTINE COMPUTES AVERAGE DAILY VALUES OF THE
C FLOW PARAMETERS
C
      SUBROUTINE FIND(NYEARS,NMONTH,NOBAR,FLOMLL,FLOMLR,FLOWMV,
     +BIGLAT,SMALAT,TDAYS,RAINM,RUNM,EVAM,RECHM,FLORUN,CAPINF,
     +ANFLOW,BALFLO)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /BIND/ FLOML,TOTML,TOMFLO,BIGEST,SMALST,TOTFL,TOTFV,
     +AVDALF,AVMOLF,AVMOVF,AVDAVF,AVDL,AVML,DAIFTO,TOTLL,TOTLR,
     +AVDLL,AVDLR,AVDALL,AVDALR,AVMLL,AVMLR,ANRAIN,ANPERC,
     +ANRUN,ANEVAT,AVDAIN,AVDAIC,AVDAUN,AVDAET,AVEFCY,ANVPE,ANVERT
      REAL *8 FLOML(1,2,12),TOTML(2,12),TOMFLO(2,12),TOTFL(1),TOTFV(1),A
     +VDALF(1),AVMOLF(1),AVMOVF(1),AVDAVF(1),FLOMLL(1,2,12),FLOMLR(1,2,1
     +2),FLOWMV(1,2,12),BIGLAT(1),SMALAT(1), TOTLL(1),TOTLR(1),AVDALL(1
     +),AVDALR(1),RAINM(2,12),RUNM(2,12),EVAM(2,12),ANRUN(2),ANPERC(2),A
     +NEVAT(2),ANRAIN(2),AVDAIN(2),AVDAIC(2),AVDAUN(2),AVDAET(2),VERFLO(
     +1,2),AVEFCY(2),PRENT(2),FLTENT(2),RECHM(2,12)
      REAL *8 ANFLOW(2),ANINF(2),FLORUN(2,12),CAPINF(2,12),ANVPE(2),ANVE
     +RT(2),BALFLO(2)
      TOTAL=0.0
      TOTALL=0.0
      TOTALR=0.0
      TOTAV=0.0
      DO 600 I=1,NOBAR
      DO 600 J=1,NYEARS
      DO 600 K=1,NMONTH
      FLOML(I,J,K)=(FLOMLL(I,J,K)+FLOMLR(I,J,K))/2.0
 600  CONTINUE
      DO 601 I=1,NYEARS
      DO 601 J=1,NMONTH
      IF(RECHM(I,J).LT.0.0)RECHM(I,J)=0.0
 601  CONTINUE
      DO 605 I=1,NOBAR
      DO 605 J=1,NYEARS
      DO 605 K=1,NMONTH
      TOTML(J,K)=TOTML(J,K)+FLOML(I,J,K)
 605  CONTINUE
      DO 610 I=1,NYEARS
      ANRAIN(I)=0.0
      ANPERC(I)=0.0
      ANRUN(I)=0.0
      ANEVAT(I)=0.0
      ANINF(I)=0.0
      ANFLOW(I)=0.0
      DO 610 J=1,NMONTH
      TOMFLO(I,J)=TOTML(I,J)+FLOWMV(NOBAR,I,J)
```

```
      ANRAIN(I)=ANRAIN(I)+RAINM(I,J)
      ANRUN(I)=ANRUN(I)+RUNM(I,J)
      ANEVAT(I)=ANEVAT(I)+EVAM(I,J)
      ANINF(I)=ANINF(I)+CAPINF(I,J)
      ANFLOW(I)=ANFLOW(I)+FLORUN(I,J)
  610 CONTINUE
      DO 611 I=1,NYEARS
      AVDAIN(I)=ANRAIN(I)/TDAYS
      AVDAUN(I)=ANRUN(I)/TDAYS
      AVDAET(I)=ANEVAT(I)/TDAYS
      ANPERC(I)=ANRAIN(I)-ANRUN(I)-ANEVAT(I)
      AVDAIC(I)=AVDAIN(I)-AVDAUN(I)-AVDAET(I)
  611 CONTINUE
      DO 615 I=1,NOBAR
      BIGEST=BIGEST+BIGLAT(I)
      SMALST=SMALST+SMALAT(I)
  615 CONTINUE
      DO 621 I=1,NOBAR
      DO 621 J=1,NYEARS
      VERFLO(I,J)=0.0
      DO 621 K=1,NMONTH
      VERFLO(I,J)=VERFLO(I,J)+FLOWMV(I,J,K)
  621 CONTINUE
      DO 620 I=1,NOBAR
      DO 620 J=1,NYEARS
      DO 620 K=1,NMONTH
      TOTFL(I)=TOTFL(I)+FLOML(I,J,K)
      TOTLL(I)=TOTLL(I)+FLOMLL(I,J,K)
      TOTLR(I)=TOTLR(I)+FLOMLR(I,J,K)
      TOTFV(I)=TOTFV(I)+FLOWMV(I,J,K)
  620 CONTINUE
      TOTDAY=NYEARS*TDAYS
      TOTMO=NYEARS*NMONTH
      DO 625 I=1,NOBAR
      AVDALF(I)=TOTFL(I)/TOTDAY
      AVDALL(I)=TOTLL(I)/TOTDAY
      AVDALR(I)=TOTLR(I)/TOTDAY
      AVMOLF(I)=TOTFL(I)/TOTMO
      AVMOVF(I)=TOTFV(I)/TOTMO
      AVDAVF(I)=TOTFV(I)/TOTDAY
  625 CONTINUE
      DO 626 I=1,NYEARS
      IF(ANRAIN(I).EQ.0.0)THEN
      AVEFCY(I)=100.0
      PRENT(I)=0.0
      ELSE
      AVEFCY(I)=100.0-100.0*VERFLO(1,I)/ANRAIN(I)
      ENDIF
      BALFLO(I)=ANVPE(I)-TOTLL(1)-ANVERT(I)
  626 CONTINUE
```

```
      DO 630 I=1,NOBAR
      TOTAL=TOTAL+TOTFL(I)
      TOTALL=TOTALL+TOTLL(I)
      TOTALR=TOTALR+TOTLR(I)
      TOTAV=TOTAV+TOTFV(I)
630   CONTINUE
      AVDL=TOTAL/TOTDAY
      AVML=TOTAL/TOTMO
      AVDLL=TOTALL/TOTDAY
      AVDLR=TOTALR/TOTDAY
      AVMLL=TOTALL/TOTMO
      AVMLR=TOTALR/TOTMO
      DAIFTO=AVDL+AVDAVF(NOBAR)
      RETURN
      END
```

APPENDIX D

FORTRAN PROGRAM FOR DIRECT GREEN'S FUNCTION BOUNDARY INTEGRAL SOLUTION

```
C
C DIRECT GREEN'S FUNCTION BOUNDARY INTEGRAL SOLUTION FOR
C TWO-DIMENSIONAL UNSATURATED MOISTURE-FLOW EQUATION
C                        MAIN PROGRAM
C
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /COFMN/ COEFF
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 X(NV),Y(NV),BC(NB),COEFF(NB,NB),COLMAT(NB),XIN(NI),YIN(NI),
     +SOLIN(NI),XM(ND),YM(ND),THETAP(ND),DIFFUS(ND),GRAVIT(ND),AREA(NT),
     +DTDX(ND),DTDZ(ND),THETPP(ND)
      INTEGER KODE(ND),IBOUND(NB),NINT(5),NOSING(ND),ISING(ND,9),LINKNO(
     +NT,3)
      EXTERNAL DATA,MATRIX,ELIM,OUTPUT,SOLINT,FLUXIN,CAMPINT,
     +NONLIN,GRAVIN
      OPEN(2,FILE='BEMLAP.DAT',STATUS='OLD')
      OPEN(3,FILE='BEMLAP.OUT',STATUS='UNKNOWN')
      OPEN(4,FILE='INTBOU.OUT',STATUS='UNKNOWN')
      OPEN(5,FILE='INFLUX.OUT',STATUS='UNKNOWN')
      OPEN(6,FILE='INFLUZ.OUT',STATUS='UNKNOWN')
      CALL DATA(NOBEL,NINTER,KODE,X,Y,BC,XIN,YIN,THETAS,SLOPMC,SAIE,HCON
     +S,THETIN,NOEL,DT,TIME,NOSING,ISING,LINKNO)
      NOTEL=NINTER+NOBEL
      I1=0
      I2=0
      DO 10 I=1,NOTEL
      IF((KODE(I).EQ.0).OR.(KODE(I).EQ.1))THEN
      I1=I1+1
      IBOUND(I1)=I
      ELSE
      I2=I2+1
      NINT(I2)=I
      ENDIF
 10   CONTINUE
      X(NOBEL+1)=X(1)
      Y(NOBEL+1)=Y(1)
      DO 25 I=1,NOBEL
      N=IBOUND(I)
      XM(N)=(X(I)+X(I+1))/2.0
      YM(N)=(Y(I)+Y(I+1))/2.0
 25   CONTINUE
      DO 80 I=1,NOEL
      K=1
```

```
      K1=LINKNO(I,K)
      IF((KODE(K1).EQ.0).OR.(KODE(K1).EQ.1))THEN
      XK1=XM(K1)
      YK1=YM(K1)
      ELSE
      XK1=XIN(K1)
      YK1=YIN(K1)
      ENDIF
      K2=LINKNO(I,K+1)
      IF((KODE(K2).EQ.0).OR.(KODE(K2).EQ.1))THEN
      XK2=XM(K2)
      YK2=YM(K2)
      ELSE
      XK2=XIN(K2)
      YK2=YIN(K2)
      ENDIF
      K3=LINKNO(I,K+2)
      IF((KODE(K3).EQ.0).OR.(KODE(K3).EQ.1))THEN
      XK3=XM(K3)
      YK3=YM(K3)
      ELSE
      XK3=XIN(K3)
      YK3=YIN(K3)
      ENDIF
      ARM1=XK2*YK3-YK2*XK3
      ARM2=-XK1*(YK3-YK2)
      ARM3=YK1*(XK3-XK2)
      AREA(I)=DABS(ARM1+ARM2+ARM3)/2.0
80    CONTINUE
      WRITE(4,47)
47    FORMAT(5X,'TIME',5X,'NODE #1',5X,'NODE #2',5X,'NODE #3',5X,'NODE #
     +4',5X,'NODE #5',/)
      WRITE(5,47)
      WRITE(6,47)
      TT=0.0
12    T=0.0
11    T=T+DT
      TT=TT+DT
      J1=TT/DT
      DO 45 I=1,NOBEL
      N=IBOUND(I)
      IF(KODE(N).EQ.1)BC(I)=0.001
      IF(KODE(N).EQ.0)BC(I)=0.3+0.0982*T-0.0265*T**2+0.00107*T**3+0.0000
     +595*T**4
      IF(J1.EQ.1)THEN
      IF(KODE(N).EQ.1)THEN
      THETAP(N)=THETIN
      THETPP(N)=THETAP(N)
      ELSE
      THETAP(N)=BC(I)
```

```
      THETPP(N)=THETAP(N)
      ENDIF
      ELSE
      IF(KODE(N).EQ.1)THEN
      THETPP(N)=THETAP(N)
      THETAP(N)=COLMAT(I)
      ELSE
      THETPP(N)=THETAP(N)
      THETAP(N)=BC(I)
      ENDIF
      ENDIF
45    CONTINUE
      DO 46 I=1,NINTER
      N=NINT(I)
      IF(J1.EQ.1)THEN
      THETAP(N)=THETIN
      THETPP(N)=THETAP(N)
      ELSE
      THETPP(N)=THETAP(N)
      THETAP(N)=SOLIN(I)
      ENDIF
46    CONTINUE
      DO 40 I=1,NOBEL
      N=IBOUND(I)
      DIFF1=-SLOPMC*HCONS*SAIE/THETAS
      TPP=THETAP(N)
      DIFF2=(TPP/THETAS)**(SLOPMC+2.0)
      DIFFUS(N)=DIFF1*DIFF2
40    CONTINUE
      DO 41 I=1,NINTER
      N=NINT(I)
      DIFF1=-SLOPMC*HCONS*SAIE/THETAS
      TPP=THETAP(N)
      DIFF2=(TPP/THETAS)**(SLOPMC+2.0)
      DIFFUS(N)=DIFF1*DIFF2
41    CONTINUE
      DO 20 I=1,NOBEL
      N=IBOUND(I)
      N=IBOUND(I)
      IF(J1.EQ.1)THEN
      GRAVIT(N)=0.0
      ELSE
      CALL NONLIN(X(I),X(I+1),Y(I),Y(I+1),BC(I),KODE(I),COLMAT(I),SLOPMC
     +,HCONS,THETAS,GRAVIT(N))
      ENDIF
20    CONTINUE
      DO 23 I=1,NINTER
      N=NINT(I)
      IF(J1.EQ.1)THEN
      GRAVIT(N)=0.0
```

```
      ELSE
      CALL GRAVIN(DTDZ(N),SLOPMC,HCONS,THETAS,SOLIN(I),GRAVIT(N))
      ENDIF
 23   CONTINUE
      CALL MATRIX(NOBEL,KODE,X,Y,BC,COEFF,COLMAT,THETAP,XM,YM,IBOUND,DIF
     +FUS,DT,AREA,XIN,YIN,LINKNO,NOEL)
      IIN=0
      DO 21 I=1,NOBEL
      N=IBOUND(I)
      CALL CAMPINT(I,N,LINKNO,XIN,YIN,BBB,NOEL,XM,YM,AREA,IIN,KODE,DIFFU
     +S,DT,GRAVIT,NOSING,ISING)
      COLMAT(I)=COLMAT(I)-BBB
 21   CONTINUE
      CALL ELIM(COLMAT,NOBEL)
      CALL SOLINT(NOBEL,NINTER,X,Y,XIN,YIN,BC,COLMAT,SOLIN,KODE,IBOUND,N
     +INT,DT,THETAP,DIFFUS,GRAVIT,AREA,NOSING,ISING,NOEL,LINKNO,XM,YM)
      CALL FLUXIN(NOBEL,NINTER,X,Y,XIN,YIN,BC,COLMAT,DTDX,DTDZ,KODE,LINK
     +NO,NOEL,AREA,XM,YM,NINT,IBOUND,THETAP,DIFFUS,GRAVIT,DT)
      CALL OUTPUT(NOBEL,NINTER,BC,COLMAT,SOLIN,KODE,IBOUND,NINT,TT,DTDX,
     +DTDZ)
      IF(T.LT.TIME)GOTO 11
      IF(TT.LT.30.0)GOTO 12
      STOP
      END



C
C THIS SUBROUTINE READS DATA FROM THE INPUT FILE
C
      SUBROUTINE DATA(NOBEL,NINTER,KODE,X,Y,BC,XIN,YIN,THETAS,SLOPMC,SAI
     +E,HCONS,THETIN,NOEL,DT,TIME,NOSING,ISING,LINKNO)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 X(NV),Y(NV),BC(NB),XIN(NI),YIN(NI)
      INTEGER KODE(ND),ISING(ND,9),NOSING(ND),LINKNO(NT,3)
      READ(2,*)NOBEL,NINTER,NEXEL,NOEL
      READ(2,*)THETAS,SLOPMC,SAIE,HCONS,THETIN,DT,TIME
      DO 10 I=1,NOBEL
      READ(2,*)X(I),Y(I)
 10   CONTINUE
      DO 20 I=1,NOEL
      READ(2,*)(LINKNO(I,K),K=1,3)
 20   CONTINUE
      NOTEL=NINTER+NEXEL
      DO 30 I=1,NINTER
      READ(2,*)XIN(I),YIN(I)
 30   CONTINUE
      DO 40 I=1,NOTEL
```

```
      READ(2,*)KODE(I),NOSING(I)
 40   CONTINUE
      DO 45 I=1,NOTEL
      MS=NOSING(I)
      READ(2,*)(ISING(I,K),K=1,MS)
 45   CONTINUE
      RETURN
      END
```

```
C
C THIS SUBROUTINE COMPUTES THE GRAVITY TERM AT THE BOUNDARY
C
      SUBROUTINE NONLIN(XX1,XX2,YY1,YY2,BC1,K1,COLMAZ,SLOPMC,HCONS,THETA
     +S,GVIT)
      IMPLICIT REAL *8(A-H,O-Z)
      R2=DSQRT((XX2-XX1)**2+(YY2-YY1)**2)
      CO=(XX1-XX2)/R2
      EX1=2.0*SLOPMC+3.0
      EX2=2.0*SLOPMC+2.0
      GRA1=EX1*HCONS/(THETAS**EX1)
      IF(K1.EQ.1)THEN
      GVIT=GRA1*(COLMAZ**EX2)*BC1*CO
      ELSE
      GVIT=GRA1*(BC1**EX2)*COLMAZ*CO
      ENDIF
      RETURN
      END
```

```
C
C THIS SUBROUTINE EVALUATES THE GRAVITY TERM IN THE DOMAIN
C
      SUBROUTINE GRAVIN(DTZZ,SLOPMC,HCONS,THETAS,SOLNN,GVIT)
      IMPLICIT REAL *8(A-H,O-Z)
      EX1=2.0*SLOPMC+3.0
      EX2=2.0*SLOPMC+2.0
      GRA1=EX1*HCONS/(THETAS**EX1)
      GVIT=GRA1*(SOLNN**EX2)*DTZZ
      RETURN
      END
```

```
C
C THIS SUBROUTINE COMPUTES THE ELEMENTS OF THE MATRICES FOR BIM
C
      SUBROUTINE MATRIX(NOBEL,KODE,X,Y,BC,COEFF,COLMAT,THETAP,XM,YM,IB
     +OUND,DIFFUS,DT,AREA,XIN,YIN,LINKNO,NOEL)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 X(NV),Y(NV),BC(NB),COEFF(NB,NB),COLMAT(NB),XM(ND),YM(ND),TH
     +ETAP(ND),DIFFUS(ND),AREA(NT),XIN(NI),YIN(NI)
      INTEGER KODE(ND),IBOUND(NB),LINKNO(NT,3)
      EXTERNAL INTEG,SOURCE
      IN=0
      DO 20 I=1,NOBEL
      N=IBOUND(I)
      COLMAT(I)=0.0
      DO 20 J=1,NOBEL
      M=IBOUND(J)
      CALL INTEG(I,J,X(J),X(J+1),Y(J),Y(J+1),XM(N),YM(N),HH,GG,IN,DIFFUS
     +(M),DT)
      IF(KODE(N).EQ.1)THEN
      IF(I.EQ.J)THEN
      COEFF(I,J)=0.50
      ELSE
      IF(KODE(M).EQ.1)COEFF(I,J)=HH
      IF(KODE(M).EQ.0)COEFF(I,J)=-GG
      ENDIF
      ELSE
      IF(KODE(M).EQ.1)COEFF(I,J)=HH
      IF(KODE(M).EQ.0)COEFF(I,J)=-GG
      ENDIF
      IF(KODE(M).EQ.1)COLMAT(I)=COLMAT(I)+BC(J)*GG
      IF((KODE(M).EQ.0).AND.(I.NE.J))COLMAT(I)=COLMAT(I)-BC(J)*HH
      IF((KODE(M).EQ.0).AND.(I.EQ.J))COLMAT(I)=COLMAT(I)-BC(J)*0.5
 20   CONTINUE
      IIN=0
      DO 30 I=1,NOBEL
      N=IBOUND(I)
      CALL SOURCE(I,N,LINKNO,XIN,YIN,BBB,NOEL,XM,YM,AREA,IIN,KODE,THETAP
     +,DIFFUS,DT)
      COLMAT(I)=COLMAT(I)+BBB
 30   CONTINUE
      RETURN
      END
```

```
C
C THIS SUBROUTINE COMPUTES THE DOMAIN INTEGRAL
C ASSOCIATED WITH THE GRAVITY TERM
C
      SUBROUTINE CAMPINT(I,N,LINKNO,XIN,YIN,BCM,NOEL,XM,YM,AREA,IIN,KODE
     +,DIFFUS,DT,GRAVIT,NOSING,ISING)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 XIN(NI),YIN(NI),AREA(NT),XM(ND),YM(ND),DIFFUS(ND),GRAVIT(ND
     +),TC1(3),TC2(3),TC3(3),OMEG(3),XCO(3),YCO(3)
      INTEGER LINKNO(NT,3),KODE(ND),NOSING(ND),ISING(ND,9)
      EXTERNAL ERROR1
      CC=0.5771
      OMEG(1)=1.0/3.0
      OMEG(2)=1.0/3.0
      OMEG(3)=1.0/3.0
      TC1(1)=0.0
      TC1(2)=0.0
      TC1(3)=1.0
      TC2(1)=0.0
      TC2(2)=1.0
      TC2(3)=0.0
      TC3(1)=1.0
      TC3(2)=0.0
      TC3(3)=0.0
      BCM=0.0
      MS=NOSING(N)
      DO 20 J=1,NOEL
      K1=LINKNO(J,1)
      IF((KODE(K1).EQ.0).OR.(KODE(K1).EQ.1))THEN
      XC1=XM(K1)
      YC1=YM(K1)
      ELSE
      XC1=XIN(K1)
      YC1=YIN(K1)
      ENDIF
      K2=LINKNO(J,2)
      IF((KODE(K2).EQ.0).OR.(KODE(K2).EQ.1))THEN
      XC2=XM(K2)
      YC2=YM(K2)
      ELSE
      XC2=XIN(K2)
      YC2=YIN(K2)
      ENDIF
      K3=LINKNO(J,3)
      IF((KODE(K3).EQ.0).OR.(KODE(K3).EQ.1))THEN
      XC3=XM(K3)
      YC3=YM(K3)
      ELSE
      XC3=XIN(K3)
```

```
      YC3=YIN(K3)
      ENDIF
      AVDIFF=(DIFFUS(K1)+DIFFUS(K2)+DIFFUS(K3))/3.0
      AVGRAV=(GRAVIT(K1)+GRAVIT(K2)+GRAVIT(K3))/3.0
      NSING=0
      DO 15 KS=1,MS
      IF(ISING(N,KS).EQ.J)NSING=J
15    CONTINUE
      IF(NSING.EQ.J)THEN
      R2=DSQRT((XC2-XC3)**2+(YC2-YC3)**2)
      CO=(XC3-XC2)/R2
      SI=(YC3-YC2)/R2
      SINGIN1=0.0
      SINGI21=0.0
      DO 16 K=1,3
      K1=LINKNO(J,K)
      XCO(K)=TC1(K)*XC1+TC2(K)*XC2+TC3(K)*XC3
      YCO(K)=TC1(K)*YC1+TC2(K)*YC2+TC3(K)*YC3
      IF(IIN.EQ.0)THEN
      RAD=DSQRT((XM(N)-XCO(K))**2+(YM(N)-YCO(K))**2)
      ELSE
      RAD=DSQRT((XIN(I)-XCO(K))**2+(YIN(I)-YCO(K))**2)
      ENDIF
      DENLO=4.0*DIFFUS(K1)*DT
      DENO=4.0*3.14*DIFFUS(K1)
      SINGIN1=SINGIN1-AREA(J)*OMEG(K)*GRAVIT(K1)*CC/DENO
      SINGI21=SINGI21-AREA(J)*OMEG(K)*GRAVIT(K1)*DLOG(1.0/DENLO)/DENO
16    CONTINUE
      AX=(XC2-XC3)/2.0
      AY=(YC2-YC3)/2.0
      IF(AX.EQ.0.0)THEN
      DIST=DABS(XC1-XC3)
      ELSE
      TA=AY/AX
      DIST=DABS(TA*XC1-YC1+YC3-TA*XC3)/DSQRT(TA**2+1.0)
      ENDIF
      SIG=(XC3-XC1)*(YC2-YC1)-(XC2-XC1)*(YC3-YC1)
      IF(SIG.LT.0.0)THEN
      DIST=-DIST
      ELSE
      ENDIF
      SIC1=YC1*SI+XC1*CO
      ETA1=XC1*SI-YC1*CO
      SIC2=YC2*SI+XC2*CO
      ETA2=XC2*SI-YC2*CO
      SIC3=YC3*SI+XC3*CO
      ETA3=XC3*SI-YC3*CO
      ALPHA=(SIC3-SIC1)/(ETA3-ETA1)
      BETA=(SIC2-SIC1)/(ETA2-ETA1)
      BRAC1=(ALPHA-BETA)*(DLOG(DIST)-1.5)
```

```
      BRAC2=ALPHA*(DLOG(1.0+ALPHA**2))/2.0
      BRAC3=BETA*(DLOG(1.0+BETA**2))/2.0
      BRAC4=ATAN(ALPHA)-ATAN(BETA)
      PRO=2.0*AVGRAV/(4.0*AVDIFF*3.14)
      SINGI22=-PRO*(BRAC1+BRAC2-BRAC3+BRAC4)*(DIST**2)/2.0
      BCM=BCM+SINGIN1+SINGI21+SINGI22
      ELSE
      BCM1=0.0
      DO 21 K=1,3
      K1=LINKNO(J,K)
      XCO(K)=TC1(K)*XC1+TC2(K)*XC2+TC3(K)*XC3
      YCO(K)=TC1(K)*YC1+TC2(K)*YC2+TC3(K)*YC3
      IF(IIN.EQ.0)THEN
      RAD=DSQRT((XM(N)-XCO(K))**2+(YM(N)-YCO(K))**2)
      ELSE
      RAD=DSQRT((XIN(I)-XCO(K))**2+(YIN(I)-YCO(K))**2)
      ENDIF
      A=RAD**2/(4.0*DIFFUS(K1)*DT)
      DENO=4.0*3.14*DIFFUS(K1)
      BCM1=BCM1+OMEG(K)*GRAVIT(K1)*ERROR1(A)/DENO
 21   CONTINUE
      BCM=BCM+AREA(J)*BCM1
      ENDIF
 20   CONTINUE
      RETURN
      END


C
C THIS SUBROUTINE SOLVES SIMULTANEOUS EQUATIONS BY
C GAUSS-JORDAN ELIMINATION METHOD
C
      SUBROUTINE ELIM(COLMAT,NOBEL)
      IMPLICIT REAL *8(A-H,O-Z)
      COMMON /COFMN/ COEFF
      PARAMETER (NB=20,NT=28)
      REAL*8 COEFF(NB,NB),PP(NT),COLMAT(NB)
      DO 947 I=1,NOBEL
      IF(I.EQ.NOBEL)GOTO 956
      BIG=DABS(COEFF(I,I))
      I1=I+1
      DO 951 L=I1,NOBEL
      IF(DABS(COEFF(L,I)).GT.BIG)BIG=DABS(COEFF(L,I))
 951  CONTINUE
      DO 952 L=I,NOBEL
      IF(DABS(COEFF(L,I)).EQ.BIG)NR=L
 952  CONTINUE
      DO 953 M=1,NOBEL
```

```
      PP(M)=COEFF(I,M)
  953 CONTINUE
      NON1=NOBEL+1
      PP(NON1)=COLMAT(I)
      DO 954 M=1,NOBEL
      COEFF(I,M)=COEFF(NR,M)
  954 CONTINUE
      COLMAT(I)=COLMAT(NR)
      DO 955 M=1,NOBEL
      COEFF(NR,M)=PP(M)
  955 CONTINUE
      COLMAT(NR)=PP(NON1)
  956 DEN=COEFF(I,I)
      DO 948 J=1,NOBEL
      COEFF(I,J)=COEFF(I,J)/DEN
  948 CONTINUE
      COLMAT(I)=COLMAT(I)/DEN
      DO 949 K=1,NOBEL
      IF(K.EQ.I)GOTO 949
      IF(COEFF(K,I).EQ.0.0)GOTO 949
      CON=COEFF(K,I)
      DO 950 J=1,NOBEL
      IF(DABS(COEFF(I,J)).LT.1.0E-30)THEN
      COEFF(I,J)=1.0E-30
      ELSE
      COEFF(K,J)=COEFF(K,J)-COEFF(I,J)*CON
      ENDIF
  950 CONTINUE
      COLMAT(K)=COLMAT(K)-COLMAT(I)*CON
  949 CONTINUE
  947 CONTINUE
      RETURN
      END


C
C THIS SUBROUTINE COMPUTES THE MOISTURE CONTENTS IN THE DOMAIN
C
      SUBROUTINE SOLINT(NOBEL,NINTER,X,Y,XIN,YIN,BC,COLMAT,SOLIN,KODE,IB
     +OUND,NINT,DT,THETAP,DIFFUS,GRAVIT,AREA,NOSING,ISING,NOEL,LINKNO,XM
     +,YM)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 X(NV),Y(NV),XIN(NI),YIN(NI),SOLIN(NI),BC(NB),COLMAT(NB),THE
     +TAP(ND),DIFFUS(ND),GRAVIT(ND),AREA(NT),XM(NB),YM(NB)
      INTEGER KODE(ND),IBOUND(NB),NINT(5),NOSING(ND),ISING(ND,9),LINKNO(
     +NT,3)
      EXTERNAL INTEG,CAMPINT,SOURCE
```

```
      IN=1
      IIN=1
      DO 10 I=1,NINTER
      N=NINT(I)
      CALL SOURCE(I,N,LINKNO,XIN,YIN,BCB,NOEL,XM,YM,AREA,IIN,KODE,THETAP
     +,DIFFUS,DT)
      SOLIN(I)=0.0
      DO 11 J=1,NOBEL
      M=IBOUND(J)
      CALL INTEG(I,J,X(J),X(J+1),Y(J),Y(J+1),XIN(I),YIN(I),AA,BB,IN,DIFF
     +US(M),DT)
      IF(KODE(M).EQ.1)SOLIN(I)=SOLIN(I)+BC(J)*BB-COLMAT(J)*AA
      IF(KODE(M).EQ.0)SOLIN(I)=SOLIN(I)+COLMAT(J)*BB-BC(J)*AA
 11   CONTINUE
      SOLIN(I)=SOLIN(I)+BCB
 10   CONTINUE
      DO 20 I=1,NINTER
      N=NINT(I)
      IIN=1
      CALL CAMPINT(I,N,LINKNO,XIN,YIN,BBB,NOEL,XM,YM,AREA,IIN,KODE,DIFFU
     +S,DT,GRAVIT,NOSING,ISING)
      SOLIN(I)=SOLIN(I)-BBB
 20   CONTINUE
      RETURN
      END



C
C THIS SUBROUTINE COMPUTES THE MOISTURE-FLUX IN THE DOMAIN
C
      SUBROUTINEFLUXIN(NOBEL,NINTER,X,Y,XIN,YIN,BC,COLMAT,DTDX,DTDZ,KOD
     +E,LINKNO,NOEL,AREA,XM,YM,NINT,IBOUND,THETAP,DIFFUS,GRAVIT,DT)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 X(NV),Y(NV),XIN(NI),YIN(NI),DTDX(ND),DTDZ(ND),BC(NB),COLMAT
     +(NB),AREA(NT),XM(NB),YM(NB),THETAP(ND),DIFFUS(ND),GRAVIT(ND)
      INTEGER KODE(ND),LINKNO(NT,3),NINT(5),IBOUND(NB)
      EXTERNAL FINTEG,FSOURC
      IN=1
      IIN=1
      DO 10 I=1,NINTER
      N=NINT(I)
      ID=0
      CALL FSOURC(I,LINKNO,XM,YM,XIN,YIN,BCCX,BCCZ,NOEL,AREA,KODE,THETAP
     +,DIFFUS,ID,DT)
      ID=1
      CALLFSOURC(I,LINKNO,XM,YM,XIN,YIN,BCGX,BCGZ,NOEL,AREA,KODE,GRAVIT
     +,DIFFUS,ID,DT)
```

```
      DTDX(N)=0.0
      DTDZ(N)=0.0
      DO 11 J=1,NOBEL
      M=IBOUND(J)
      CALL FINTEG(X(J),X(J+1),Y(J),Y(J+1),XIN(I),YIN(I),AF,BF,AFZ,BFZ,DI
     +FFUS(M),DT)
      IF(KODE(M).EQ.1) DTDX(N)=DTDX(N)+BC(J)*BF-COLMAT(J)*AF
      IF(KODE(M).EQ.1)DTDZ(N)=DTDZ(N)+BC(J)*BFZ-COLMAT(J)*AFZ
      IF(KODE(M).EQ.0) DTDX(N)= DTDX(N)+COLMAT(J)*BF-BC(J)*AF
      IF(KODE(M).EQ.0)DTDZ(N)=DTDZ(N)+COLMAT(J)*BFZ-BC(J)*AFZ
 11   CONTINUE
      DTDX(N)=DTDX(N)+BCCX-BCGX
      DTDZ(N)=DTDZ(N)+BCCZ-BCGZ
 10   CONTINUE
      RETURN
      END


C
C THIS FUNCTION COMPUTES THE EXPONENTIAL INTEGRAL
C
      REAL*8 FUNCTION ERROR1(Z)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 Z
      IF(Z.LE.1.0)THEN
      A0=-0.57721566
      A1=0.99999193
      A2=-0.24991055
      A3=0.05519968
      A4=-0.00976004
      A5=0.00107857
      ERROR1=-DLOG(Z)+A0+A1*Z+A2*Z**2+A3*Z**3+A4*Z**4+A5*Z**5
      ELSE
      A1=8.5733287401
      A2=18.059016973
      A3=8.6347608925
      A4=0.2677737343
      B1=9.5733223454
      B2=25.6329561486
      B3=21.0996530827
      B4=3.9584969228
      XNUM=Z**4+A1*Z**3+A2*Z**2+A3*Z+A4
      XEN1=DEXP(-Z)
      DEN2=Z**4+B1*Z**3+B2*Z**2+B3*Z+B4
      DENO=Z*DEN2
      ERROR1=XEN1*XNUM/DENO
      ENDIF
      RETURN
```

```
      END



C
C THIS FUNCTION COMPUTES THE SERIES TO EVALUATE THE
C SINGULAR INTEGRAL AT THE BOUNDARY
C
      REAL*8 FUNCTION ERROR2(Z)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 Z
      C=0.5771
      D=DLOG(Z)
      N=0
      CC=0.0
      SUM=0.0
      DD=1.0
 10   N=N+1
      CC=CC+1.0
      DD=DD*CC
      DENO=N*(2.0*N+1.0)*DD
      PROD=((-1)**(N-1))*(Z**N)/DENO
      SUM=SUM+PROD
      IF(DABS(PROD).LT.1.0E-9)THEN
      GOTO 11
      ELSE
      GOTO 10
      ENDIF
 11   ERROR2=-C-D+2.0+SUM
      RETURN
      END



C
C THIS SUBROUTINE EVALUATES THE BOUNDARY INTEGRAL
C
      SUBROUTINE INTEG(I,J,XX1,XX2,YY1,YY2,XP,YP,HH,GG,IN,DIFFS1,DT)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 SCIE(4),OME(4),XCO(4),YCO(4)
      EXTERNAL ERROR1,ERROR2
      SCIE(1)=0.86113631
      SCIE(2)=-SCIE(1)
      SCIE(3)=0.33998104
      SCIE(4)=-SCIE(3)
      OME(1)=0.34785485
      OME(2)=OME(1)
      OME(3)=0.65214515
```

```
      OME(4)=OME(3)
      AX=(XX2-XX1)/2.0
      BX=(XX2+XX1)/2.0
      AY=(YY2-YY1)/2.0
      BY=(YY2+YY1)/2.0
      IF(((I-J).EQ.0).AND.(IN.EQ.0))THEN
      DELS=2.0*DSQRT(AX**2+AY**2)
      ALPHA=DELS**2/(16.0*DIFFS1*DT)
      GG=DELS*ERROR2(ALPHA)/(4.0*3.14)
      HH=0.0
      ELSE
      IF(AX.EQ.0.0)THEN
      DIST=DABS(XP-XX1)
      ELSE
      TA=AY/AX
      DIST=DABS((TA*XP-YP+YY1-TA*XX1)/DSQRT(TA**2+1))
      ENDIF
      SIG=(XX1-XP)*(YY2-YP)-(XX2-XP)*(YY1-YP)
      IF(SIG.LT.0.0)THEN
      DIST=-DIST
      ELSE
      GOTO 32
      ENDIF
 32   GG=0.0
      HH=0.0
      AXY=DSQRT(AX**2+AY**2)
      DO 15 K=1,4
      XCO(K)=AX*SCIE(K)+BX
      YCO(K)=AY*SCIE(K)+BY
      RAD=DSQRT((XP-XCO(K))**2+(YP-YCO(K))**2)
      A=(RAD**2)/(4.0*DIFFS1*DT)
      BRAC=44.0*RAD**2
      GG=GG+7.0*ERROR1(A)*OME(K)*AXY/88.0
      HH=HH-7.0*(DEXP(-A)*DIST*OME(K)*AXY)/BRAC
 15   CONTINUE
      ENDIF
      RETURN
      END


C
C THIS SUBROUTINE EVALUATES THE BOUNDARY INTEGRAL
C FOR MOISTURE-FLUX
C
      SUBROUTINE FINTEG(XX1,XX2,YY1,YY2,XP,YP,HHX,GGX,HHZ,GGZ,DIFFS1,DT)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 SCIE(4),OME(4),XCO(4),YCO(4)
      SCIE(1)=0.86113631
```

```
      SCIE(2)=-SCIE(1)
      SCIE(3)=0.33998104
      SCIE(4)=-SCIE(3)
      OME(1)=0.34785485
      OME(2)=OME(1)
      OME(3)=0.65214515
      OME(4)=OME(3)
      AX=(XX2-XX1)/2.0
      BX=(XX2+XX1)/2.0
      AY=(YY2-YY1)/2.0
      BY=(YY2+YY1)/2.0
      R2=DSQRT((XX2-XX1)**2+(YY2-YY1)**2)
      CO=(XX1-XX2)/R2
      SI=(YY1-YY2)/R2
      IF(AX.EQ.0.0)THEN
      DIST=DABS(XP-XX1)
      ELSE
      TA=AY/AX
      DIST=DABS((TA*XP-YP+YY1-TA*XX1)/SQRT(TA**2+1))
      ENDIF
      SIG=(XX1-XP)*(YY2-YP)-(XX2-XP)*(YY1-YP)
      IF(SIG.LT.0.0)THEN
      DIST=-DIST
      ELSE
      GOTO 32
      ENDIF
32    GGX=0.0
      GGZ=0.0
      HHX=0.0
      HHZ=0.0
      DO 15 K=1,4
      XCO(K)=AX*SCIE(K)+BX
      YCO(K)=AY*SCIE(K)+BY
      XXX=-XCO(K)+XP
      ZZZ=-YCO(K)+YP
      RAD=DSQRT((XP-XCO(K))**2+(YP-YCO(K))**2)
      SEE=-SI*(YCO(K)-YP)-CO*(XCO(K)-XP)
34    DRDX=-(-SEE*CO-DIST*SI)/RAD
      DRDZ=-(DIST*CO-SEE*SI)/RAD
      A=RAD**2/(4.0*DIFFS1*DT)
      GGX=GGX-7.0*(DRDX/RAD)*DEXP(-A)*OME(K)*DSQRT(AX**2+AY**2)/44.0
      GGZ=GGZ-7.0*(DRDZ/RAD)*DEXP(-A)*OME(K)*DSQRT(AX**2+AY**2)/44.0
      BHX1=DIST*DRDX/((RAD**3)*3.14)
      BHX2=DEXP(-A)*(1.0+A)
      BHX3=BHX1*BHX2
      BHX4=SI*DEXP(-A)/((RAD**2)*6.28)
      BHX=BHX3-BHX4
      HHX=HHX+BHX*OME(K)*DSQRT(AX**2+AY**2)
      BHZ1=DIST*DRDZ/((RAD**3)*3.14)
      BHZ2=DEXP(-A)*(1.0+A)
```

```
      BHZ3=BHZ1*BHZ2
      BHZ4=-CO*DEXP(-A)/((RAD**2)*6.28)
      BHZ=BHZ3-BHZ4
      HHZ=HHZ+BHZ*OME(K)*DSQRT(AX**2+AY**2)
 15   CONTINUE
      RETURN
      END




C
C THIS SUBROUTINE EVALUATES THE DOMAIN INTEGRAL
C ASSOCIATED WITH THE INITIAL CONDITION
C
      SUBROUTINE SOURCE(I,N,LINKNO,XIN,YIN,BCM,NOEL,XM,YM,AREA,IIN,KODE,
     +THETAP,DIFFUS,DT)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,ND=25)
      REAL*8 XIN(NI),YIN(NI),AREA(NT),OMEG(3),XCO(4),YCO(4),TC1(3),TC2(3
     +),TC3(3),XM(ND),YM(ND),DIFFUS(ND),THETAP(ND)
      INTEGER LINKNO(NT,3),KODE(ND)
      OMEG(1)=1.0/3.0
      OMEG(2)=1.0/3.0
      OMEG(3)=1.0/3.0
      TC1(1)=0.5
      TC1(2)=0.5
      TC1(3)=0.0
      TC2(1)=0.0
      TC2(2)=0.5
      TC2(3)=0.5
      TC3(1)=0.5
      TC3(2)=0.0
      TC3(3)=0.5
      BCM=0.0
      DO 20 J=1,NOEL
      K1=LINKNO(J,1)
      IF((KODE(K1).EQ.0).OR.(KODE(K1).EQ.1))THEN
      XC1=XM(K1)
      YC1=YM(K1)
      ELSE
      XC1=XIN(K1)
      YC1=YIN(K1)
      ENDIF
      K2=LINKNO(J,2)
      IF((KODE(K2).EQ.0).OR.(KODE(K2).EQ.1))THEN
      XC2=XM(K2)
      YC2=YM(K2)
      ELSE
      XC2=XIN(K2)
```

```
      YC2=YIN(K2)
      ENDIF
      K3=LINKNO(J,3)
      IF((KODE(K3).EQ.0).OR.(KODE(K3).EQ.1))THEN
      XC3=XM(K3)
      YC3=YM(K3)
      ELSE
      XC3=XIN(K3)
      YC3=YIN(K3)
      ENDIF
      BCM1=0.0
      DO 21 K=1,3
      K1=LINKNO(J,K)
      IF(K.NE.3)THEN
      K2=LINKNO(J,K+1)
      ELSE
      K2=LINKNO(J,1)
      ENDIF
      XCO(K)=TC1(K)*XC1+TC2(K)*XC2+TC3(K)*XC3
      YCO(K)=TC1(K)*YC1+TC2(K)*YC2+TC3(K)*YC3
      IF(IIN.EQ.0)THEN
      RAD=DSQRT((XM(N)-XCO(K))**2+(YM(N)-YCO(K))**2)
      ELSE
      RAD=DSQRT((XIN(I)-XCO(K))**2+(YIN(I)-YCO(K))**2)
      ENDIF
      AVDIFF=(DIFFUS(K1)+DIFFUS(K2))/2.0
      AVTHET=(THETAP(K1)+THETAP(K2))/2.0
      A=RAD**2/(4.0*AVDIFF*DT)
      DENO=4.0*3.14*AVDIFF*DT
      BCM1=BCM1+OMEG(K)*AVTHET*DEXP(-A)/DENO
 21   CONTINUE
      BCM=BCM+AREA(J)*BCM1
 20   CONTINUE
      RETURN
      END



C
C THIS SUBROUTINE COMPUTES THE DOMAIN INTEGRAL FOR MOISTURE-FLUX
C
      SUBROUTINE FSOURC(I,LINKNO,XM,YM,XIN,YIN,BCMX,BCMZ,NOEL,AREA,KODE,
     +THEGRA,DIFFUS,ID,DT)
      IMPLICIT REAL*8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,ND=25,NT=28)
      REAL*8 XIN(NI),YIN(NI),AREA(NT),XM(ND),YM(ND),THEGRA(ND),DIFFUS(ND
     +)
      INTEGER LINKNO(NT,3),KODE(ND)
      TC1=1.0/3.0
```

```
      TC2=1.0/3.0
      TC3=1.0/3.0
      BCMX=0.0
      BCMZ=0.0
      DO 20 J=1,NOEL
      K1=LINKNO(J,1)
      IF((KODE(K1).EQ.0).OR.(KODE(K1).EQ.1))THEN
      XC1=XM(K1)
      YC1=YM(K1)
      ELSE
      XC1=XIN(K1)
      YC1=YIN(K1)
      ENDIF
      K2=LINKNO(J,2)
      IF((KODE(K2).EQ.0).OR.(KODE(K2).EQ.1))THEN
      XC2=XM(K2)
      YC2=YM(K2)
      ELSE
      XC2=XIN(K2)
      YC2=YIN(K2)
      ENDIF
      K3=LINKNO(J,3)
      IF((KODE(K3).EQ.0).OR.(KODE(K3).EQ.1))THEN
      XC3=XM(K3)
      YC3=YM(K3)
      ELSE
      XC3=XIN(K3)
      YC3=YIN(K3)
      ENDIF
      AVDIFG=(THEGRA(K1)+THEGRA(K2)+THEGRA(K3))/3.0
      AVDIFF=(DIFFUS(K1)+DIFFUS(K2)+DIFFUS(K3))/3.0
      XCO=TC1*XC1+TC2*XC2+TC3*XC3
      YCO=TC1*YC1+TC2*YC2+TC3*YC3
      XXX=-(XCO-XIN(I))
      ZZZ=-(YCO-YIN(I))
      RAD=DSQRT((XIN(I)-XCO)**2+(YIN(I)-YCO)**2)
      DRDX=XXX/RAD
      DRDZ=ZZZ/RAD
      A=RAD**2/(4.0*AVDIFF*DT)
      IF(ID.EQ.0)THEN
      BCMX1=-AVDIFG*DEXP(-A)*XXX/(8.0*3.14*(AVDIFF**2)*(DT**2))
      BCMZ1=-AVDIFG*DEXP(-A)*ZZZ/(8.0*3.14*(AVDIFF**2)*(DT**2))
      ELSE
      BCMX1=-7.0*AVDIFG*DEXP(-A)*XXX/((RAD**2)*AVDIFF*44.0)
      BCMZ1=-7.0*AVDIFG*DEXP(-A)*ZZZ/((RAD**2)*AVDIFF*44.0)
      ENDIF
      BCMX=BCMX+AREA(J)*BCMX1
      BCMZ=BCMZ+AREA(J)*BCMZ1
20    CONTINUE
      RETURN
```

```
      END




C
C THIS SUBROUTINE WRITES THE VALUES IN THE OUTPUT FILES
C
      SUBROUTINEOUTPUT(NOBEL,NINTER,BC,COLMAT,SOLIN,KODE,IBOUND,NINT,TT
     +,DTDX,DTDZ)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,ND=25)
      REAL*8 BC(NB),COLMAT(NB),SOLIN(NI),DTDX(ND),DTDZ(ND)
      INTEGER KODE(ND),IBOUND(NB),NINT(NI)
      WRITE(3,37)TT
 37   FORMAT(1X,'TIME  = ',F5.2)
      WRITE(3,36)
 36   FORMAT(5X,'BOUNDARY ELEMENT SOLUTION OF POISSON'S EQUATION
     + USING',
     +/,12X,'BOUNDARY INTEGRATION FOR THE SOURCE TERM',/,5X,'__________
     +_____________________________________',/////)
      WRITE(3,35)
 35   FORMAT(5X,'TABLE : SOLUTION  AT THE BOUNDARY ',/,5X,'___________
     +_________________________',/)
      WRITE(3,15)
 15   FORMAT(5X,'NODE NO.',5X,'POTENTIAL',10X,'FLUX',/,5X,'________',5X
     +,'_________',5X,'___________'/)
      DO 20 I=1,NOBEL
      N=IBOUND(I)
      IF(KODE(N).EQ.0)WRITE(3,25)N,BC(I),COLMAT(I)
 25   FORMAT(7X,I2,7X,F10.5,6X,F10.5)
      IF(KODE(N).EQ.1)WRITE(3,25)N,COLMAT(I),BC(I)
 20   CONTINUE
      WRITE(3,21)
 21   FORMAT(5X,'____________________________________________',/////)
      WRITE(3,45)
 45   FORMAT(5X,'TABLE : SOLUTION IN THE DOMAIN',/,5X,'________________
     +_____________________________________')
      WRITE(3,40)
 40   FORMAT(/,5X,'NODE NO.',5X,'POTENTIAL',5X,'FLUX IN X-DIR.',4X,'FLUX
     + IN Z-DIR.',/,5X,'________',5X,'_________',5X,'_____________',4X,
     +'_____________',/)
      DO 30 I=1,NINTER
      N=NINT(I)
      WRITE(3,50)N,SOLIN(I)
 50   FORMAT(8X,I2,6X,F10.5)
 30   CONTINUE
      WRITE(3,31)
 31   FORMAT(5X,'_______________________________________________________
     +____',/////)
```

```
      WRITE(4,70)TT,SOLIN(1),SOLIN(2),SOLIN(3),SOLIN(4),SOLIN(5)
70    FORMAT(6(3X,F10.5))
      IF(TT.EQ.1.0)THEN
      WRITE(5,73)
73    FORMAT(4X,'NODE NO.',4X,'FLUX IN X-DIR',4X,'FLUX IN Z-DIR')
      DO 71 I=1,NINTER
      WRITE(5,72)I,DTDX(I),DTDZ(I)
72    FORMAT(5X,I2,5X,F10.5,5X,F10.5)
71    CONTINUE
      ELSE
      ENDIF
      RETURN
      END
```

APPENDIX E

FORTRAN PROGRAM FOR PERTURBATION GREEN'S FUNCTION BOUNDARY INTEGRAL SOLUTION

```
C
C PERTURBATION GREEN'S FUNCTION BOUNDARY INTEGRAL SOLUTION FOR
C TWO-DIMENSIONAL UNSATURATED MOISTURE-FLOW EQUATION
C                        MAIN PROGRAM
C
    IMPLICIT REAL *8(A-H,O-Z)
    COMMON /COFMN/ COEFF
    PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
    REAL*8 X(NV),Y(NV),BC(NB),COEFF(NB,NB),COLMAT(NB),XIN(NI),YIN(NI),
   +SOLIN(NI),XM(ND),YM(ND),THETAP(ND),DIFFUS(ND),THETPP(ND),RHMAT(NB)
   +,GRAVIT(ND),AREA(NT),DTDX(ND),DTDZ(ND),THETAF(ND),BCC(NB)
    INTEGER KODE(ND),IBOUND(NB),NINT(5),NOSING(ND),ISING(ND,9),LINKNO(
   +NT,3)
    EXTERNAL DATA,MATRIX,ELIM,OUTPUT,SOLINT,FLUXIN,CAMPINT,NONLIN,
   +GRAVIN
    OPEN(2,FILE='BEMLAP.DAT',STATUS='OLD')
    OPEN(3,FILE='BEMLAP.OUT',STATUS='UNKNOWN')
    OPEN(4,FILE='INTBOU.OUT',STATUS='UNKNOWN')
    CALL DATA(NOBEL,NINTER,KODE,X,Y,BCC,XIN,YIN,THETAS,SLOPMC,SAIE,HCO
   +NS,THETIN,NOEL,DT,TIME,NOSING,ISING,LINKNO)
    NOTEL=NINTER+NOBEL
    I1=0
    I2=0
    DO 10 I=1,NOTEL
    IF((KODE(I).EQ.0).OR.(KODE(I).EQ.1))THEN
    I1=I1+1
    IBOUND(I1)=I
    ELSE
    I2=I2+1
    NINT(I2)=I
    ENDIF
10  CONTINUE
    WRITE(4,47)
47  FORMAT(5X,'TIME',5X,'NODE #1',5X,'NODE #2',5X,'NODE #3',5X,'NODE #
   +4',5X,'NODE #5',/)
    TT=0.0
12  T=0.0
11  T=T+DT
    TT=TT+DT
    WRITE(*,*)'TT=',TT
    J1=TT/DT
    DO 45 I=1,NOBEL
```

```
      N=IBOUND(I)
      IF(KODE(N).EQ.1)BCC(I)=0.001
      IF(KODE(N).EQ.0)BCC(I)=0.3+0.0982*T-0.0265*T**2+0.00107*T**3+0.000
     +0595*T**4
      IF(J1.EQ.1)THEN
      IF(KODE(N).EQ.1)THEN
      THETAP(N)=THETIN
      THETPP(N)=THETAP(N)
      ELSE
      THETAP(N)=BCC(I)
      THETPP(N)=THETAP(N)
      ENDIF
      ELSE
      IF(KODE(N).EQ.1)THEN
      THETPP(N)=THETAP(N)
      THETAP(N)=THETAF(N)
      ELSE
      THETPP(N)=THETAP(N)
      THETAP(N)=BCC(I)
      ENDIF
      ENDIF
 45   CONTINUE
      DO 49 I=1,NINTER
      N=NINT(I)
      IF(J1.EQ.1)THEN
      THETAP(N)=THETIN
      THETPP(N)=THETAP(N)
      ELSE
      THETPP(N)=THETAP(N)
      THETAP(N)=THETAF(N)
      ENDIF
 49   CONTINUE
      DO 40 I=1,NOBEL
      N=IBOUND(I)
      DIFF1=-SLOPMC*HCONS*SAIE/THETAS
      TPP=THETAP(N)
      DIFF2=(TPP/THETAS)**(SLOPMC+2.0)
      DIFFUS(N)=DIFF1*DIFF2
 40   CONTINUE
      DO 41 I=1,NINTER
      N=NINT(I)
      DIFF1=-SLOPMC*HCONS*SAIE/THETAS
      TPP=THETAP(N)
      DIFF2=(TPP/THETAS)**(SLOPMC+2.0)
      DIFFUS(N)=DIFF1*DIFF2
 41   CONTINUE
      DO 46 I=1,NOBEL
      N=IBOUND(I)
      THETAF(N)=0.0
 46   CONTINUE
```

```
      DO 48 I=1,NINTER
      N=NINT(I)
      THETAF(N)=0.0
48    CONTINUE
      II=0
22    II=II+1
      DO 31 I=1,NOBEL
      N=IBOUND(I)
      IF(II.EQ.1)THEN
      BC(I)=BCC(I)
      ELSE
      BC(I)=0.0
      THETAP(N)=0.0
      ENDIF
31    CONTINUE
      DO 27 I=1,NINTER
      N=NINT(I)
      IF(II.GT.1)THETAP(N)=0.0
27    CONTINUE
      CALLMATRIX(NOBEL,KODE,X,Y,BC,COEFF,RHMAT,THETAP,XM,YM,IBOUND,DIFF
     +US,DT,AREA,XIN,YIN,LINKNO,NOEL)
      DO 20 I=1,NOBEL
      N=IBOUND(I)
      N=IBOUND(I)
      IF((J1.EQ.1).AND.(II.EQ.1))THEN
      GRAVIT(N)=0.0
      ELSE
      CALL NONLIN(X(I),X(I+1),Y(I),Y(I+1),BC(I),KODE(I),COLMAT(I),SLOPMC
     +,HCONS,THETAS,GRAVIT(N))
      ENDIF
20    CONTINUE
      DO 23 I=1,NINTER
      N=NINT(I)
      IF((J1.EQ.1).AND.(II.EQ.1))THEN
      GRAVIT(N)=0.0
      ELSE
      CALL GRAVIN(DTDZ(N),SLOPMC,HCONS,THETAS,SOLIN(I),GRAVIT(N))
      ENDIF
23    CONTINUE
      IIN=0
      DO 21 I=1,NOBEL
      N=IBOUND(I)
      CALL CAMPINT(I,N,LINKNO,XIN,YIN,BBB,NOEL,XM,YM,AREA,IIN,KODE,DIFFU
     +S,DT,GRAVIT,NOSING,ISING)
      COLMAT(I)=RHMAT(I)-BBB
21    CONTINUE
      CALL ELIM(COLMAT,NOBEL)
      CALL SOLINT(NOBEL,NINTER,X,Y,XIN,YIN,BC,COLMAT,SOLIN,KODE,IBOUND,N
     +INT,DT,THETAP,DIFFUS,GRAVIT,AREA,NOSING,ISING,NOEL,LINKNO,XM,YM)
      CALL FLUXIN(NOBEL,NINTER,X,Y,XIN,YIN,BC,COLMAT,DTDX,DTDZ,KODE,LINK
```

```
     +NO,NOEL,AREA,XM,YM,NINT,IBOUND,THETAP,DIFFUS,GRAVIT,DT)
      DO 24 I=1,NINTER
      N=NINT(I)
      THETAF(N)=THETAF(N)+SOLIN(I)
 24   CONTINUE
      DO 26 I=1,NOBEL
      N=IBOUND(I)
      IF(KODE(N).EQ.1)THEN
      THETAF(N)=THETAF(N)+COLMAT(I)
      ELSE
      THETAF(N)=THETAF(N)+BC(I)
      ENDIF
 26   CONTINUE
      TSLN=0.0
      DO 28 I=1,NINTER
      TSLN=TSLN+SOLIN(I)
 28   CONTINUE
      DO 29 I=1,NOBEL
      N=IBOUND(I)
      IF(KODE(N).EQ.1)TSLN=TSLN+COLMAT(I)
      IF(KODE(N).EQ.0)TSLN=TSLN+BC(I)
 29   CONTINUE
      IF(DABS(TSLN).GT.1.0E-5)GOTO 22
      CALL OUTPUT(NOBEL,NINTER,BC,COLMAT,THETAF,KODE,IBOUND,NINT,TT)
      IF(T.LT.TIME)GOTO 11
      IF(TT.LT.30.0)GOTO 12
      STOP
      END



C
C THIS SUBROUTINE COMPUTES THE MATRICES IN THE PGF SOLUTION
C
      SUBROUTINE MATRIX(NOBEL,KODE,X,Y,BC,COEFF,RHMAT,THETAP,XM,YM,
     +IBOUND,DIFFUS,DT,AREA,XIN,YIN,LINKNO,NOEL)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NI=5,NT=28,NV=21,ND=25)
      REAL*8 X(NV),Y(NV),BC(NB),COEFF(NB,NB),RHMAT(NB),XM(ND),YM(ND),THE
     +TAP(ND),DIFFUS(ND),AREA(NT),XIN(NI),YIN(NI)
      INTEGER KODE(ND),IBOUND(NB),LINKNO(NT,3)
      EXTERNAL INTEG,SOURCE
      X(NOBEL+1)=X(1)
      Y(NOBEL+1)=Y(1)
      DO 25 I=1,NOBEL
      N=IBOUND(I)
      XM(N)=(X(I)+X(I+1))/2.0
      YM(N)=(Y(I)+Y(I+1))/2.0
```

```
25  CONTINUE
    IN=0
    DO 20 I=1,NOBEL
    N=IBOUND(I)
    RHMAT(I)=0.0
    DO 20 J=1,NOBEL
    M=IBOUND(J)
    CALL INTEG(I,J,X(J),X(J+1),Y(J),Y(J+1),XM(N),YM(N),HH,GG,IN,DIFFUS
   +(M),DT)
    IF(KODE(N).EQ.1)THEN
    IF(I.EQ.J)THEN
    COEFF(I,J)=0.50
    ELSE
    IF(KODE(M).EQ.1)COEFF(I,J)=HH
    IF(KODE(M).EQ.0)COEFF(I,J)=-GG
    ENDIF
    ELSE
    IF(KODE(M).EQ.1)COEFF(I,J)=HH
    IF(KODE(M).EQ.0)COEFF(I,J)=-GG
    ENDIF
    IF(KODE(M).EQ.1)RHMAT(I)=RHMAT(I)+BC(J)*GG
    IF((KODE(M).EQ.0).AND.(I.NE.J))RHMAT(I)=RHMAT(I)-BC(J)*HH
    IF((KODE(M).EQ.0).AND.(I.EQ.J))RHMAT(I)=RHMAT(I)-BC(J)*0.5
20  CONTINUE
    DO 10 I=1,NOEL
    K=1
    K1=LINKNO(I,K)
    IF((KODE(K1).EQ.0).OR.(KODE(K1).EQ.1))THEN
    XK1=XM(K1)
    YK1=YM(K1)
    ELSE
    XK1=XIN(K1)
    YK1=YIN(K1)
    ENDIF
    K2=LINKNO(I,K+1)
    IF((KODE(K2).EQ.0).OR.(KODE(K2).EQ.1))THEN
    XK2=XM(K2)
    YK2=YM(K2)
    ELSE
    XK2=XIN(K2)
    YK2=YIN(K2)
    ENDIF
    K3=LINKNO(I,K+2)
    IF((KODE(K3).EQ.0).OR.(KODE(K3).EQ.1))THEN
    XK3=XM(K3)
    YK3=YM(K3)
    ELSE
    XK3=XIN(K3)
    YK3=YIN(K3)
    ENDIF
```

```
      ARM1=XK2*YK3-YK2*XK3
      ARM2=-XK1*(YK3-YK2)
      ARM3=YK1*(XK3-XK2)
      AREA(I)=DABS(ARM1+ARM2+ARM3)/2.0
 10   CONTINUE
      IIN=0
      DO 30 I=1,NOBEL
      N=IBOUND(I)
      CALL SOURCE(I,N,LINKNO,XIN,YIN,BBB,NOEL,XM,YM,AREA,IIN,KODE,THETAP
     +,DIFFUS,DT)
      RHMAT(I)=RHMAT(I)+BBB
 30   CONTINUE
      RETURN
      END



C
C THIS SUBROUTINE WRITES THE VALUES IN THE OUTPUT FILES
C
      SUBROUTINE OUTPUT(NOBEL,NINTER,BC,COLMAT,THETAF,KODE,IBOUND,
     +NINT,TT)
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER(NB=20,NV=21,NI=5,NT=28,ND=25)
      REAL*8 BC(NB),COLMAT(NB),THETAF(ND)
      INTEGER KODE(ND),IBOUND(NB),NINT(NI)
      WRITE(3,37)TT
 37   FORMAT(1X,'TIME = ',F5.2)
      WRITE(3,36)
 36   FORMAT(5X,'BOUNDARY ELEMENT SOLUTION OF POISSON'S
     +EQUATION USING',/,12X,'BOUNDARY INTEGRATION FOR THE
     +SOURCE TERM',/,5X,'___________________________________
     +__________________',/////)
      WRITE(3,35)
 35   FORMAT(5X,'TABLE : SOLUTION  AT THE BOUNDARY ',/,5X,'_____________
     +____________________________',/)
      WRITE(3,15)
 15   FORMAT(5X,'NODE NO.',5X,'POTENTIAL',10X,'FLUX',/,5X,'________',5X
     +,'__________',5X,'____________'/)
      DO 20 I=1,NOBEL
      N=IBOUND(I)
      IF(KODE(N).EQ.0)WRITE(3,25)N,BC(I),COLMAT(I)
 25   FORMAT(7X,I2,7X,F10.5,6X,F10.5)
      IF(KODE(N).EQ.1)WRITE(3,25)N,COLMAT(I),BC(I)
 20   CONTINUE
      WRITE(3,21)
 21   FORMAT(5X,'________________________________________',/////)
      WRITE(3,45)
```

```
45  FORMAT(5X,'TABLE : SOLUTION IN THE DOMAIN',/,5X,'_______________
   +_________________________________________')
    WRITE(3,40)
40  FORMAT(/,5X,'NODE NO.',5X,'POTENTIAL',5X,'FLUX IN X-DIR.',4X,'FLUX
   + IN Z-DIR.',/,5X,'________',5X,'_________',5X,'_____________',4X,
   +'_____________',/)
    DO 30 I=1,NINTER
    N=NINT(I)
    WRITE(3,50)N,THETAF(N)
50  FORMAT(8X,I2,6X,F10.5)
30  CONTINUE
    WRITE(3,31)
31  FORMAT(5X,'_____________________________________________
   +____',/////)
    WRITE(4,70)TT,THETAF(1),THETAF(2),THETAF(3),THETAF(4),THETAF(5)
70  FORMAT(6(3X,F10.5))
    RETURN
    END
```

```
C The program also includes the SUBROUTINES CAMPINT, NONLIN, GRAVIN,
C INTEG, ELIM, DATA, SOLINT, FLUXIN, FINTEG, SOURCE, FSOURC and
C FUNCTIONS ERROR1 AND ERROR2.
C The rest of the program is the same as in the DGF solution.
```

APPENDIX F

FORTRAN PROGRAMS FOR ANALYTICAL SOLUTIONS

```
C
C ANALYTICAL SOLUTION FOR TWO-DIMENSIONAL MOISTURE-FLOW PROBLEM
C
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 THETA(10),X(10),Z(10),N,M,THETAP(10),DIFFUS(10),DTDZ(10),PP
     +(10),DTDX(10)
      EXTERNAL ANIN
      OPEN(2,FILE='ANAL1.DAT',STATUS='UNKNOWN')
      OPEN(3,FILE='ANAL1.OUT',STATUS='UNKNOWN')
      OPEN(4,FILE='COMP1.OUT',STATUS='UNKNOWN')
      OPEN(5,FILE='COMPX.OUT',STATUS='UNKNOWN')
      OPEN(6,FILE='COMPZ.OUT',STATUS='UNKNOWN')
      CALL ANIN(XX,ZZ,AA,BB,CC,DD,NINTER,X,Z,THETIN,SLOPMC,HCONS,DT,SAIE
     +,THETAS)
      WRITE(4,47)
 47   FORMAT(5X,'TIME',5X,'NODE #1',5X,'NODE #2',5X,'NODE #3',5X,'NODE #
     +4',5X,'NODE #5',/)
      WRITE(6,47)
      WRITE(5,47)
      TT=0.0
 17   T=0.0
 15   T=T+DT
      TT=TT+DT
      J1=TT/DT
      BB=0.3+0.0982*T-0.0265*T**2+0.00107*T**3+0.0000595*T**4
      AA=0.3+0.0982*T-0.0265*T**2+0.00107*T**3+0.0000595*T**4
      DO 25 I=1,NINTER
      IF(J1.EQ.1)THEN
      THETAP(I)=THETIN
      ELSE
      THETAP(I)=THETA(I)
      ENDIF
      IF(J1.EQ.1)THEN
      PP(I)=0.0
      ELSE
      GRAV1=(2.0*SLOPMC+3.0)*HCONS*DTDZ(I)
      GRAV2=THETAP(I)**(2.0*SLOPMC+2.0)
      GRAV3=THETAS**(2.0*SLOPMC+3.0)
      PP(I)=GRAV1*GRAV2/GRAV3
      ENDIF
 25   CONTINUE
      DO 10 I=1,NINTER
```

```
      DIFF1=-SLOPMC*HCONS*SAIE/THETAS
      DIFF2=(THETAP(I)/THETAS)**(SLOPMC+2.0)
      DIFFUS(I)=DIFF1*DIFF2
      N=-1
      COSMN=0.0
      SINMN=0.0
      SINMXX=0.0
20    N=N+1
      M=-1
      SINM=0.0
      SINMX=0.0
21    M=M+1
      OMEGAN=(2.0*N+1.0)*22.0/(14.0*ZZ)
      OMEGAM=(2.0*M+1.0)*22.0/(14.0*XX)
      CHAT=4.0*CC/((2.0*N+1.0)*3.14)
      SINNT=DSIN((2.0*N+1.0)*22.0/14.0)
      BETAN1=2.0*(ZZ**2)*SINNT/((2.0*N+1.0)*3.14)
      BETAN2=(2.0*ZZ/((2.0*N+1.0)*3.14))**2
      BETAN3=BB-AA-DD*(BETAN1-BETAN2)/ZZ
      BETAN=4.0*(BETAN3)/((2.0*N+1.0)*3.14)
      GAMAN1=THETAP(I)-AA-DD*(BETAN1-BETAN2)/ZZ
      GAMAN=4.0*GAMAN1/((2.0*N+1.0)*3.14)
      SINMT=DSIN((2.0*M+1.0)*22.0/14.0)
      GAMHA1=2.0*(XX**2)*SINMT/((2.0*M+1.0)*3.14)
      GAMHA2=(2.0*XX/((2.0*M+1.0)*3.14))**2
      GAMHA3=GAMAN-BETAN-CHAT*(GAMHA1-GAMHA2)/XX
      GAMHAT=4.0*GAMHA3/((2.0*M+1.0)*3.14)
      C2HAT=DIFFUS(I)*(OMEGAN**2)*CHAT
      ALPHAC=DIFFUS(I)*CHAT/XX
      ALPHAB=DIFFUS(I)*(OMEGAN**2)*BETAN
      ALPHAD=DIFFUS(I)*DD/ZZ
      TNBAR=4.0*(ALPHAD-PP(I))/((2.0*N+1.0)*3.14)
      TNHAT=ALPHAC-ALPHAB-C2HAT*(X(I)**2)/(2.0*XX)+TNBAR
      TMXT1=TNBAR+ALPHAC-ALPHAB-C2HAT*(GAMHA1-GAMHA2)/XX
      TMN=4.0*(TMXT1)/((2.0*M+1.0)*3.14)
      ALPHAT=DIFFUS(I)*(OMEGAN**2+OMEGAM**2)
      CMT=TMN/ALPHAT+GAMHAT*DEXP(-ALPHAT*DT)
      SINM1=CMT*DSIN(OMEGAM*X(I))
      SINMX1=CMT*OMEGAM*DCOS(OMEGAM*X(I))
      SINM=SINM+SINM1
      SINMX=SINMX+SINMX1
      IF(DABS(SINM1).GT.1.0E-10)GOTO 21
      SINMM=SINM+BETAN+CHAT*(X(I)**2)/(2.0*XX)
      SINMN1=SINMM*DSIN(OMEGAN*Z(I))
      COSMN1=SINMM*OMEGAN*DCOS(OMEGAN*Z(I))
      COSMN=COSMN+COSMN1
      SINMN=SINMN+SINMN1
      SINMNX=SINMX+CHAT*X(I)/XX
      SINMNX1=SINMNX*DSIN(OMEGAN*Z(I))
      SINMXX=SINMXX+SINMNX1
```

```
      IF(DABS(SINMN1).GT.1.0E-8)GOTO 20
      IF(DABS(COSMN1).GT.1.0E-8)GOTO 20
      IF(DABS(SINMNX1).GT.1.0E-8)GOTO 20
      THETA(I)=SINMN+AA+(Z(I)**2)*DD/(2.0*ZZ)
      DTDZ(I)=COSMN+Z(I)*DD/ZZ
      DTDX(I)=SINMXX
 10   CONTINUE
      WRITE(*,*)'TT=',TT
      WRITE(4,16)TT,THETA(1),THETA(2),THETA(3),THETA(4),THETA(5)
      WRITE(6,16)TT,DTDZ(1),DTDZ(2),DTDZ(3),DTDZ(4),DTDZ(5)
      WRITE(5,16)TT,DTDX(1),DTDX(2),DTDX(3),DTDX(4),DTDX(5)
 16   FORMAT(6(3X,F10.5))
      WRITE(3,*)'TT=',TT
      WRITE(3,12)
 12   FORMAT(8X,'ANALYTICAL SOLUTION FOR RECHARD'S EQUATION',/,11X,'WITH
     + FLUX AT THE UPPER BOUNDARY',/,8X,'______________________________
     +____________________',/////)
      WRITE(3,11)
 11   FORMAT(5X,'TABLE : SOLUTION FOR INTERNAL NODES',/,5X,'___________
     +_______________________________________',/)
      WRITE(3,2)
 2    FORMAT(5X,'NODE NO.',3X,'POTENTIAL',3X,'FLUX IN X-DIR',3X,'FLUX IN
     + Z-DIR',/,5X,'________',3X,'_________',3X,'_____________',3X,'____
     +________',/)
      DO 30 I=1,NINTER
      WRITE(3,1)I,THETA(I)
 1    FORMAT(5X,I3,5X,F10.5)
 30   CONTINUE
      WRITE(3,31)
 31   FORMAT(5X,'_______________________________________________'//)
      IF(T.LT.10.)GOTO 15
      IF(TT.LT.30.0)GOTO 17
 35   STOP
      END
```

```
C
C THIS SUBROUTINE READS DATA FOR TWO-DIMENSIONAL ANALYTICAL
C SOLUTION OF THE UNSATURATED MOISTURE-FLOW EQUATION
C
      SUBROUTINE ANIN(XX,ZZ,AA,BB,CC,DD,NINTER,X,Z,THETIN,SLOPMC,HCONS,D
     +T,SAIE,THETAS)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 X(10),Z(10)
      READ(2,*)NINTER
      READ(2,*)XX,ZZ,THETIN,SLOPMC,HCONS,DT,SAIE,THETAS
      READ(2,*)AA,BB,CC,DD
      DO 10 I=1,NINTER
```

```
      READ(2,*)X(I),Z(I)
 10   CONTINUE
      RETURN
      END




C
C ANALYTICAL SOLUTION FOR ONE-DIMENSIONAL MOISTURE-FLOW PROBLEM
C
      IMPLICIT REAL *8(A-H,O-Z)
      REAL *8 THETAF(50),ZZ(50),THETAP(50),DIFFUS(50)
      EXTERNAL DATA,ERROR
      OPEN(2,FILE='DATA.DAT',STATUS='UNKNOWN')
      OPEN(4,FILE='ANAL.OUT',STATUS='UNKNOWN')
      CALL DATA(BC,THETIN,TIME,DT,SLOPMC,SAIE,THETAS,NODE,HCONS,ZZ)
      WRITE(4,47)
 47   FORMAT(5X,'TIME',5X,'NODE  4',5X,'NODE  7',5X,'NODE 10',5X,'NODE 1
     +3',5X,'NODE 16',/)
      TT=0.0
 12   T=0.0
 11   T=T+DT
      TT=TT+DT
      J1=TT/DT
      BC=0.3+0.0982*T-0.0265*(T**2)+0.00107*(T**3)+0.0000595*(T**4)
      DO 45 I=1,NODE
      IF(J1.EQ.1)THEN
      THETAP(I)=THETIN
      ELSE
      THETAP(I)=THETAF(I)
      ENDIF
 45   CONTINUE
      DO 40 I=1,NODE
      DIFF1=-SLOPMC*HCONS*SAIE/THETAS
      TPP=THETAP(I)
      DIFF2=(TPP/THETAS)**(SLOPMC+2.0)
      DIFFUS(I)=DIFF1*DIFF2
 40   CONTINUE
      DO 10 I=1,NODE
      ARG=ZZ(I)/(2.0*DSQRT(DIFFUS(I)*T))
      THETAF(I)=BC+(THETAP(I)-BC)*ERROR(ARG)
 10   CONTINUE
      WRITE(4,70)TT,THETAF(4),THETAF(7),THETAF(10),THETAF(13),THETAF(16)
 70   FORMAT(6(3X,F10.5))
      IF(T.LT.TIME)GOTO 11
      IF(TT.LT.30.0)GOTO 12
      STOP
      END
```

```
C
C THIS SUBROUTINE COMPUTES ERROR FUNCTION FOR ONE-DIMENSIONAL
C ANALYTICAL SOLUTION OF THE MOISTURE-FLOW EQUATION
C
      REAL*8 FUNCTION ERROR(Z)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL*8 Z
      PP=0.3275911
      A1=0.254829592
      A2=-0.284496736
      A3=1.421413741
      A4=-1.453152027
      A5=1.061405429
      TT=1.0/(1.0+PP*Z)
      ERF1Z=A1*TT + A2*TT**2 + A3*TT**3 + A4*TT**4 + A5*TT**5
      ERFCZ=ERF1Z*DEXP(-Z**2)
      ERROR=1.0-ERFCZ
      RETURN
      END
```

```
C
C THIS SUBROUTINE READS DATA FOR ONE-DIMENSIONAL ANALYTICAL SOLUTION
C OF THE UNSATURATED MOISTURE-FLOW EQUATION
C
      SUBROUTINE DATA(THETIN,TIME,DT,SLOPMC,SAIE,THETAS,NODE,HCONS,ZZ)
      IMPLICIT REAL *8(A-H,O-Z)
      REAL *8 ZZ(50)
      READ(2,*)NODE,HCONS,THETAS,SAIE,SLOPMC,DT,TIME,THETIN
      DO 10 I=1,NODE
      READ(2,*)ZZ(I)
      ZZ(I)=0.6*ZZ(I)
 10   CONTINUE
      RETURN
      END
```

```
C
C ANALYTICAL SOLUTION FOR SIMPLIFIED LEACHATE FLOW
C EQUATION IN THE SATURATED ZONE
C
      IMPLICIT REAL *8(A-H,O-Z)
      PARAMETER (NT=25)
```

```
      INTEGER LCONL(NT),LCONR(NT),ICOL(NT)
      REAL *8 THICK(NT),HDIN(NT),SOURCE(NT),HDP(NT),HPP(NT),HPS(NT),HDD(
     +NT),X(NT)
      EXTERNAL ELIM,SATDAT
      OPEN(2,FILE='SAT.DAT',STATUS='OLD')
      OPEN(3,FILE='SAT.OUT',STATUS='UNKNOWN')
      OPEN(4,FILE='COMPAN.OUT',STATUS='UNKNOWN')
      CALL SATDAT(THICK,CONDB,DX,HCONS,NOCOL,LCONL,LCONR,ICOL,EFPORO,DT,
     +SOURCE,HDIN,SLOPE,BCONL,BCONR,TIME,EPS)
      AVTHICK=0.0
      XLENG=DX*(NOCOL+1.0)
      DO 55 I=1,NOCOL
      X(I)=I*DX
 55   CONTINUE
      T=0.0
 11   T=T+1.0
      J1=T/1.0
      DO 45 I=1,NOCOL
      IF(J1.EQ.1)THEN
      HDP(I)=HDIN(I)
      HPP(I)=HDP(I)
      ELSE
      HPP(I)=HDP(I)
      HDP(I)=HDD(I)
      ENDIF
 45   CONTINUE
      DO 10 I=1,NOCOL
      HPS(I)=(HDP(I)+HPP(I))/2.0
 10   CONTINUE
      DO 35 I=1,NOCOL
      HDD(I)=0.0
 35   CONTINUE
      DO 15 I=1,NOCOL
      N=-1
 31   N=N+1
      OMEGA=(2.0*N+1.0)*22.0/(14.0*XLENG)
      XSINX=((1.0/OMEGA)**2)*((-1.0)**N)
      COSX=14.0*XLENG*((-1.0)**N)/((2.0*N+1.0)*22.0)
      XCOSX=14.0*(XLENG**2)*((-1.0)**N)/((2.0*N+1.0)*22.0)-(1.0/OMEGA)**
     +2
      X2COSX=14.0*(XLENG**3)*((-1.0)**N)/((2.0*N+1.0)*22.0)-28.0*XLENG*X
     +SINX/((2.0*N+1.0)*22.0)
      CNO=2.0*HDP(I)*COSX/XLENG-2.0*BCONL*XCOSX/XLENG+2.0*BCONL*X2COSX/(
     +XLENG**2)-2.0*BCONR*X2COSX/(XLENG**3)
      ALPHA=HCONS*HPS(I)/EFPORO
      ALP2HAT=ALPHA*(OMEGA**2)
      PB1=CONDB
      PHAT=(-SOURCE(I)+PB1)/EFPORO
      BETA=-CONDB/(THICK(I)*EFPORO)
      TN1=4.0*ALPHA*BCONR*COSX/(XLENG**3)
```

```
      TN2=4.0*ALPHA*BCONL*COSX/(XLENG**2)
      TN3=2.0*BETA*BCONL*XCOSX/XLENG
      TN4=2.0*BETA*BCONL*X2COSX/XLENG**2
      TN5=2.0*BETA*BCONR*X2COSX/XLENG**3
      TN6=2.0*PHAT*COSX/XLENG
      TNHAT=TN1-TN2+TN3-TN4+TN5-TN6
      ALPHAT=ALP2HAT-BETA
      GG=CNO-TNHAT/ALPHAT
      CNT=TNHAT/ALPHAT+GG*DEXP(-ALPHAT*DT)
      HDST1=CNT*DCOS(OMEGA*X(I))
      HDD(I)=HDD(I)+HDST1
      IF(DABS(HDST1).GT.1.0E-10)GOTO 31
      AX=BCONL*X(I)*(1.0-X(I)/(XLENG))+BCONR*(X(I)**2)/(XLENG**2)
      HDD(I)=HDD(I)+AX
 15   CONTINUE
      IF(J1.EQ.1)THEN
      WRITE(4,42)
 42   FORMAT(1X,'TIME',2X,'# 8',2X,'# 9',2X,'# 10',2X,'# 11',2X,'# 12',2
     +X,'# 13',/)
      ELSE
      ENDIF
      WRITE(4,41)T,HDD(8),HDD(9),HDD(10),HDD(11),HDD(12),HDD(13)
 41   FORMAT(1X,F5.2,6(F10.2))
      WRITE(3,50)T
 50   FORMAT(5X,'TIME T = ',F5.2,1X,'DAYS',//,5X,'NODE NO.',10X,'MOUND H
     +EAD (FT)')
      DO 52 I=1,NOCOL
      WRITE(3,51)I,HDD(I)
 51   FORMAT(5X,I2,8X,F10.2)
 52   CONTINUE
      IF(T.LT.TIME)GOTO 11
      STOP
      END




C
C THIS SUBROUTINE READS DATA FOR THE ANALYTICAL SOLUTION
C OF THE SATURATED LEACHATE FLOW EQUATION
C
      SUBROUTINE SATDAT(THICK,CONDB,DX,HCONS,NOCOL,LCONL,LCONR,ICOL,
     +EFPORO,DT,SOURCE,HDIN,SLOPE,BCONL,BCONR,TIME,EPS)
      IMPLICIT REAL*8(A-H,O-Z)
      PARAMETER(NT=25)
      REAL *8 THICK(NT),SOURCE(NT),HDIN(NT)
      INTEGER LCONL(NT),LCONR(NT),ICOL(NT)
      READ(2,*)CONDB,DX,DT,HCONS,EFPORO,SLOPE
      READ(2,*)BCONL,BCONR,TIME,EPS
```

```
      READ(2,*)NOCOL
      DO 10 I=1,NOCOL
      ICOL(I)=1
      READ(2,*)LCONL(I),LCONR(I),SOURCE(I),THICK(I),HDIN(I)
10    CONTINUE
      ICOL(NOCOL+1)=2
      ICOL(NOCOL+2)=2
      RETURN
      END
```

BIBLIOGRAPHY

Abbott, M. B., H. J. K. Anderson, K. Havno, K. Hogh J., Ul Kroszynski & IR Warren, 1982. "Research and Development for the Unsaturated Zone Component of the European Hydrologic System-Systeme Hydrologique European (SHE)." In: Engineering Applications of Computational Hydraulics, Vol. 1., Editor: M. B. Abbott and J. A. Cunge. Pitman Advanced Publishing Program. Boston-London-Melbourne. pp. 41-69.

Abnish, C. A., 1975. "Groundwater Recharge Simulation." J. Hydraulics Div. ASCE. Vol. 101. No. 9, September.

Abramowitz, M. and I. A. Stegan, 1970. Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables. U.S. Government Printing Office, Washington, D.C.

Aral, M. M. and Tang, Y., 1988. "A New Boundary Element Formulation for Time-Dependent Confined and Unconfined Aquifer Problems." Water Resour. Res. Vol. 24. No. 6. pp. 831-842.

Batu, V., 1982. "Time-dependent, Linearized Two-dimensional Infiltration and

Evapotranspiration from Nonuniform and Nonperiodic Strip Sources." Water Resour. Res. Vol. 18, No. 6. pp. 1725-1733.

Bear, Jacob, 1979. Hydraulics of Groundwater. McGraw-Hill Inc., USA. p. 196-198 (569p.).

Bear, Jacob, 1972. Dynamics of fluids in Porous Media. American Elsevier Publishing Company Inc. New York. p. 368-369 (764p.).

Bras, Rafael L., 1990. Hydrology : An Introduction to Hydrologic Science. Addison-Wesley Publishing Company, USA. pp. 485-488.

Brebbia C. A., 1978. The Boundary Element Method for Engineers. Addison-Wesley, Mass.

Brebbia, C. A., Telles J., and Wrobel, L., 1982. Boundary Element Methods - Theory and Application. Springer-Verlag. New York.

Brebbia, C., Telles, J., and Wrobel, L., 1984. Boundary Element Techniques. Springer-Verlag. 464p.

Bristow Keith L. and Williams John, 1987. "Sensitivity of simulated infiltration to changes in soil hydraulic properties." In : Infiltration Development and Application.

Editor : Yu-Si Fok. University of Hawaii at Manoa. USA. pp 555-564.

Brooks, R. H. and A. T. Corey, 1964. "Hydraulic Properties of Porous Media." Hydrology Paper No. 3. Colorado State University.

Brutsaert, W., 1967. "Some Methods of Calculating Unsaturated Permeability." Trans. ASCE, 10, pp. 400-404.

Buckingham, E., 1907. "Studies of the Movement of Soil Moisture." U.S. Dept. Agr. Bur. Soils Bull., No. 38.

Campbell G. S., 1974. "A Simple Method For Determining Unsaturated Conductivity from Moisture Retention Data." Soil Science Vol. 117, No. 6. pp. 311-314.

Carnahan, B., Luther, H. A., and James Wilkes O., 1972. Applied Numerical Methods. John-Wiley and Sons. 604p.

Cheng, A. H.-D., 1984. "Darcy's Law With Variable Permeability: A Boundary Integral Solution." Water Resour. Res. Vol. 20, No. 7. pp. 980-984.

Cheng, A. H-D and James A. Liggett, 1984. "Boundary Integral Equation Method for Linear Porous-Elasticity with Applications to Soil Consolidation." International

Journal for Numerical Methods in Engineering. Vol. 20, No. 2. pp. 255-278.

Cheng, A. H-D and James A. Liggett, 1984. "Boundary Integral Equation Method for Linear Porous-Elasticity with Applications to Fracture Propagation." International Journal for Numerical Methods in Engineering. Vol. 20, No. 2. pp. 279-296.

Childs, E. C. and N. Collis-George, 1959. "The Permeability of Porous Materials." Proc. Roy. Soc., A201, pp. 392-405.

Chow, Ven Te, David R. Maidment, and Larry W. Mays, 1988. Applied Hydrology. McGraw-Hill Book Company, New York, USA. p. 281-302 (572p.).

Clapp, R. B. and G. M. Hornberger, 1978. "Empirical Equations for Some Soil Hydraulic Properties." Water Res. Res., Vol. 14, No. 4. pp. 601-604.

Cooley, R. L., 1971. Finite Element Solutions for the Equations of Groundwater Flow: Desert Research Institute Univ. Nevada, Tech. Report Series, H-W. Hydrology and Water Resources Pub. No. 18. 134p.

Dahlquist, Germand and Bjork Ake, 1974. Numerical Methods. Prentice-Hall, Inc., New York, USA.

Dass, P., G. R. Tamke, and C. M. Stoffel, 1977. "Leachate Production at Sanitary

Landfill Sites." Journal of the Environmental Engineering Division. ASCE, Vol. 103, No. 6. pp. 981-988.

Demetracopoulos, Alexander C., George P. Korfiatis, Efstathios L. Bourodimos, and Edward G. Nawy, 1984. "Modeling for Design of Landfill Bottom Liners." Journal of Environmental Engineering. Vol. 110, No. 6. pp. 1084-1098.

Demetracopoulos, Alexander C., George P. Korfiatis, Efst. L. Bourodimos, and Edward G. Nawy,1986. "Unsaturated Flow Through Solid Waste Landfills : Model and Sensitivity Analysis." Water Resources Bulletin. American Water Resources Association. Vol. 22, No. 4. pp. 601-609.

Demetracopoulos, Alexander C., 1988. "Overview of Landfill Bottom Liner Hydraulics." Water Resources Bulletin. American Water Resources Association. Vol. 24, No. 1. pp. 49-56.

Eagleson, P. S., 1970. Dynamic Hydrology. McGraw-Hill Inc., New York, USA.

Fenn Dennis G., Keith J. Henley, and Truett V. Degere, 1975. "Use of the Water Balance for Predicting Leachate Generation from Solid Waste Disposal Sites." Report SW-168. U. S. Environmental Protection Agency. pp. 8-11.

Fogden, A., K. A. Landman, and L. R. White, 1988. "Contaminant Transport in

Fractured Media: Steady state Solution by a Boundary Integral Method." Vol. 24, No. 8. pp. 1384-1396.

Gardner, W. R., 1958. "Some Steady State Solutions of the Unsaturated Moisture Flow Equation with Application to Evapotranspiration from a Water Table." Soil Sci., 85, pp. 228-232.

Gee, J. R., 1981. "Prediction of Leachate Accumulation in Sanitary Landfills." Proceedings, 4th Annual Madison Conferences of Applied Research and Practice on Municipal and Industrial Waste. University of Wisconsin, Madison, Wis.

Gupta, S. K. and K. K. Tanji, 1976. "A Three-dimensional Galerkin Finite Element Solution of Flow through Multiaquifers in Sutter Basin, California." Water Resources Research. Vol. 12, No. 2. pp. 155-162.

Haverkamp, R., L. Rendon, and G. Vachaud, 1987. "Infiltration Equations and Their Applicability for Predictive Use." In : Infiltration Development and Application. Editor : Yu-Si Fok . University of Hawaii at Manoa, U. S. A. pp. 142-152.

Hillel Daniel, 1980. Application of Soil Physics. Academic Press Inc., USA. p. 40-43 (385p.).

Klute, A., 1972. "The Determination of the Hydraulic Conductivity and Diffusivity of Unsaturated Soils." Soil Sci., Vol. 113, No. 4. pp. 264-276.

Knisel, W. J., Jr., Editor, 1980. CREAMS : A Field Scale Model for Chemical Runoff and Erosion from Agricultural Management System. Vol. I, II, III, Draft copy, USDA-SEA, AR, Cns. Res.

Korfiatis, G. P., 1984 . Modeling the Moisture Transport Through Solid Waste Landfills. Ph.D. Dissertation, Rutgers University, Piscataway. NJ. 259 p.

Korfiatis, George P., Alexander C. Demetracopoulos, Efstathios L. Bourodimos, ans Edward G. Nawy, 1984. "Moisture Transport in a Solid Waste Column." Journal of Environmental Engineering, Vol. 110, No. 4. pp. 780-796.

Korfiatis, George P. and Alexander C. Demetracopoulos, 1986. "Flow Characteristics of Landfill Leachate Collection Systems and Liners." Journal of Environmental Engineering, Vol. 112, No. 3. pp. 538-550.

Kuppusamy, T., J. Sheng, J. C. Parker, and R. J. Lenhard, 1987."Finite-Element Analysis of Multiphase Immiscible Flow Through Soils." Water Resour. Res. Vol. 23, No. 4. pp. 625-631.

Lafe, O. E., Liggett, J. A., and Liu, P. L-F., 1981. "BIEM Solutions to Combinations of Leaky, Layered, Confined, Unconfined, Nonisotropic Aquifers." Water Resour.

Res. Vol. 17, No. 5. pp. 1431-1444.

Lafe, O. E. and Cheng, A. H. D., 1987. "A Perturbation Boundary Element Code for Steady state Groundwater Flow in Heterogeneous Aquifers." Water Resour. Res. Vol. 23, No. 6. pp. 1079-1084.

Lennon, G. P., Liu, P. L. F. and Liggett, J. H., 1980. "Boundary Integral Solution to Three Dimensional Unconfined Darcy's Flow." Water Resour. Res., Vol. 16. No. 4. pp. 651-658.

Liggett, J. A., 1977. "Location of Free Surface in Porous Media." J. Hydraulics Div. ASCE, Vol. 103, No. 4. pp. 353-365.

Liggett, J. A., 1982."Singular Cubature over Triangles." International Journal for Numerical Methods in Engineering. Vol. 18. pp. 1375-1384.

Liggett, J. A. and P. L-F. Liu, 1979. "Unsteady Flow in Confined Aquifers: A Comparison of Two Boundary Integral Methods." Water Resources Research. Vol. 15, No. 4. pp. 861-866.

Liu, P. L. F. and Liggett J. A., 1979. "Boundary Solution to Two Problem in Porous Media." J. Hydraulics Div., ASCE, Vol. 105, No. 3. pp. 171-183.

Liu, P. L. F., Cheng, A. H. D., Liggett, J. A., and Lee, J. H., 1981. "Boundary Integral Equation Solution to Moving Interface Between Two Fluids in Porous Media." Water Resources Res. Vol. 17, No. 5. pp. 1445-1452.

Lomen, D. O., and Warrick, A. W., 1974. "Time-dependent Linearized Infiltration, II, Line Sources." Soil Sci. Soc. Am. J., 38. pp. 568-572.

McKeon, T. J. and Chu, Wen Sen, 1987. "A Multigrid Model for Steady Flow in Partially Saturated Porous Media." Water Resour. Res. Vol. 23, No. 4. pp. 542-550.

Morse, P. M. and H. Feshbach, 1953. Methods of Theoretical Physics. Vol. 1. McGraw-Hill, New York.

Mualem, Y., 1976. "A New Model for Predicting the Hydraulic Conductivity of Unsaturated Porous Media." Water Resour. Res. Vol. 12, No. 3. pp. 513-522.

Neuman, S. P., 1973. "Saturated-Unsaturated Seepage by Finite Elements." J. Hyd. Div. ASCE. Vol. 99, No. 12. pp. 2233-2251.

Perrier, E. R. and Gibson A. C., 1980. Hydrogeologic Simulation on Solid Waste Disposal Sites. EPA - WS - 868, US EPA, Cincinnati, Ohio.

Philip, J. R., 1968. "Steady Infiltration from Buried Point Sources and Spherical Cavities." Water Resour. Res. Vol. 4, No. 5. pp. 1039-1047.

Philip, J, R., 1969. "Theory of Infiltration." In : Advances in Hydroscience, V. T. Chow (ed), Academi Press, New York. pp. 215-296.

Philip, J. R., 1971. "General Theorem on Steady Infiltration from Surface Sources with Application to Point and Line Sources." Soil Sci. Soc. Am. J. Vol. 35, No. 6. pp. 867-871.

Prickett, T. A. and Lonnquist, C. G., 1971. "Selected Digital Computer Techniques for Groundwater Resources Evaluation." Illinois State Water Survey Bull. 55. Urbana. 62p.

Remson, I., Hornberger, G. M., and Molz, F. J., 1971. Numerical Methods in Subsurface Hydrology. Wiley Interscience. New York. 389p.

Richtmeyer, R. D., and K. W. Morton, 1967. Difference Methods for Initial Value Problems. John-Wiley & Sons. New York. 405p.

Ritchie, J. T., 1972. "A model for Predicting Evaporation from Row Crop with Incomplete Cover." Water Resour. Res. Vol. 8, No. 5. pp. 1204-1213.

Rogowski, A. S., 1971. "Watershed Physics : Model of the Soil Moisture Characteristic." Water Resour. Res. Vol. 7, No. 6. pp. 1575-1582.

Rubin, J. and R. Steinhardt, 1963. "Soil Water Relations During Rain Infiltration, 1, Theory". Proc. of the Soil Science Society of America, 27, pp. 246-251.

Rulon, J. T., Rodway, R., and Freeze, R. A., 1985. "The Development of Multiple Seepage faces on Layered Slopes." Water Resour. Res. Vol. 21, No. 11. pp. 1625-1636.

Schroeder, P. R., A. C. Gibson, and M. D. Smolen, 1984. The Hydrlogic Evaluation of Landfill Performance (HELP) Model. Volume II, Documentation for Version 1. U. S. Army Engineer Water Ways Experiment Station, Vicksburg, M.S.. Produced by National Technical Information Service, U. S. Department of Commerce, Springfield, VA 22161. pp. 14-20.

Shanholtz, V. O., and J. B. Lilliard, 1970. A Soil Water Model for Two Contrasting Tillage Practices. Bulletin 38, Virginia Water Resources Research Center, VPISU, Blacksburg, VA. 217p.

Taigbenu, A. E., J. A. Liggett, and A. H-D. Cheng, 1984. "Boundary Integral Solution to Seawater Intrusion into Coastal Aquifers." Water Resour. Res. Vol. 20, No. 8. pp. 1150-1158.

Taigbenu, A. E. and J. A. Liggett, 1985. "Boundary Element Calculations of Diffusion Equation." ASCE Journal of Engineering Mechanics. Vol. 111, No. 3. pp. 311-328.

Taigbenu, A. E., 1985. A New Boundary Element Formulation Applied to Unsteady Aquifer Problems. Ph.D. Dissertation. Department of Civil Engineering, Cornell University, New York. 194 p.

Taigbenu A. E. and J. A. Liggett, 1986. "An Integral Formulation Applied to the Diffusion and Boussinesq Equations." International Journal for Numerical Methods in Engineering,. Vol. 23. pp. 1057-1079.

Taigbenu A. E. and J. A. Liggett, 1986. "An Integral Solution for the Diffusion-Advection Equation." Water Resour. Res. Vol. 22, No. 8. pp. 1237-1246.

Thomas, R. J., 1973. Groundwater Models. FAO Irrigation and Drainage Paper. UN. Rome.

Tysen, H. N. and Weber, E. M., 1963. "Use of Electronic Computers in the Simulation of the dynamic behavior of Groundwater Basins." Water Resour. Eng. Conf. Am. Soc. Civil Engr., May 13-17. Milwaukee, Wis.

Viessman, Warren J. R., John W. Knapp, Gary L. Lewis, and Terence E.

Harbaugh, 1977. Introduction to Hydrology. Harper and Row Publishers, New York. p. 372-374 (704p.).

Wanielista, Martin, 1990. Hydrology and Water Quality Control. John Wiley & Sons, Inc. New York. p. 57-58 (565p.).

Warrick, A. W., 1975. "Analytical Solutions to the One-dimensional Linearized Moisture Flow Equation for Arbitrary Input." Soil Sci. Vol. 120. pp. 79-84.

Wehran Eng. Co., 1983. Hydrologic Investigation, Fresh Kills Landfill Solid Waste Disposal Operation Plan. Vol. 1 . Prepared for: Parsons Brinckerhoff-Cosulich Consultants, One Penn Plaza, New York. p. 4-22 to 4-35.

Willis, Robert and Yeh Williams W-G, 1987. Groundwater Systems Planning and Management. Prentice-Hall, Inc. Eaglewood Cliffs, NJ 07632. 416p.

Wooding, R. A., 1968. "Steady Infiltration from a Circular Pond." Water Resour. Res., Vol. 4, No. 6. pp. 1259-1275.

Wright, S. J. and C. J. Miller, 1982. "Estimates of Seepage through Clay Covers on Hazardous waste Landfills." In Hazardous Waste Management for the 80's. T. L. Sweeney et al., eds. Ann Arbor Science Publishers, Ann Arbor, Mich., pp. 343-356.

Yeh, G. T. and D. S. Ward, 1981. FEMWASTE: A Finite-Element Model of Waste Transport Through Porous Saturated-Unsaturated Media. Oak Ridge National Laboratory, Tennessee. Environmental Sciences Division Publication No. 1462. 133p.

Yeh, G. T., 1987. FEMWATER: A Finite Element Model of Water Flow Through Saturated-Unsaturated Porous Media-First Revision. Oak Ridge National Laboratory, Tennessee. Environmental Sciences Division Publication No. 2943. 258p.

Yu-Si, F., 1987. "Evolution of Algebraic Equations." In : Infiltration Development and Application. Editor : Yu-Si Fok. University of Hawaii at Manoa, USA. pp. 38-48.

Zienkiewicz, O. C. and R. L. Taylor, 1989. Finite Element Method. 4th Edition. McGraw-Hill, New York. 648p.