United States
Environmental Protection
Agency

A Catalog of Ground Water Flow Solutions for Plume Diving Calculations

RESEARCH AND DEVELOPMENT

EPA 600/R-07/122
September 2007

A Catalog of Ground Water Flow Solutions for Plume Diving Calculations

by

Vikenti M. Gorokhovski
Senior Service America Incorporated
Silver Springs, Maryland 20910

James W. Weaver
Ecosystems Research Division
National Exposure Research Laboratory
Office of Research and Development
United States Environmental Protection Agency
Athens, Georgia 30605

U.S. Environmental Protection Agency
Office of Research and Development
Washington, DC 20460

Notice

Abstract

This report focuses on the problem of "diving plumes", a term which generally refers to plumes that go deeper into aquifers with distance from their sources. As noted by Weaver and Wilson (2000), plumes may dive for several reasons: aquifer recharge supplying clean water above the plume, aquifer stratigraphy controlling the transport direction, relatively deep pumping causing downward gradients, and the possibility of oxygenated recharge water selectively enhancing biodegradation in the upper portion of plumes. This document presents the mathematical basis of software for real-time development and refinement of site conceptual models. The emphasis in the work is on evaluation of ground water flow patterns and the proper placement of vertical sample intervals. Lack of consideration of plume diving could result in underestimation of the extent of contamination at these sites. The basics of the one-dimensional model are presented first. Solutions are then developed for flow with three sets of boundary conditions. The methodology is then extended to piecewise heterogeneous domains that allow for more flexibility in the solution. A type of inverse problem is solved that uses measured heads as a substitute for the recharge rate. The analytical method is extended to aquifers with sloping bases. Lastly a numerical model is presented for heterogeneous aquifers with uneven bases. The equations for streamlines that define plume diving and travel time are given. A comparison is made to solutions for a homogeneous aquifer and a piecewise heterogeneous aquifer.

Acknowledgments

The information in this document has been funded in part by the United States Environmental Protection Agency under cooperative agreement CQ 831622 to Senior Service America Inc., Silver Spring, MD. It has been subjected to the Agency's peer and administrative review, and it has been approved for publication as an EPA document. Mention of trade names of commercial products does not constitute endorsement or recommendation for use.

Foreword

The National Exposure Research Laboratory's Ecosystems Research Division (ERD) in Athens, Georgia, conducts research on organic and inorganic chemicals, greenhouse gas biogeochemical cycles, and land use perturbations that create direct and indirect, chemical and non-chemical stresses, exposures, and potential risks to humans and ecosystems. ERD develops, tests, applies and provides technical support for exposure and ecosystem response models used for assessing and managing risks to humans and ecosystems, within a watershed / regional context.

The Regulatory Support Branch (RSB) conducts problem-driven and applied research, develops technology tools, and provides technical support to customer Program and Regional Offices, States, Municipalities, and Tribes. Models are distributed and supported via the EPA Center for Exposure Assessment Modeling (CEAM) and through access to Internet tools (www.epa.gov/athens/onsite).

Proper assessment of ground water contaminant plumes requires decisions on where and how deep to sample. These plumes follow ground water flow paths that are controlled by the average aerial recharge, localized recharge and discharge zones, stratigraphy, and potential oxygen-enhanced biodegradation of upper contaminated zones. Methods for estimating plume diving can be used to make informed choices on locations for vertical sampling. This report provides a suite of solutions for the ground water flow equations that form the basis of plume diving calculation. The document is intended to serve as a reference for plume diving calculations embedded in several models, including the Internet tools at http://www.epa.gov/athens/onsite.

Eric J. Weber, Ph.D.
Acting Director
Ecosystems Research Division
Athens, Georgia

Leaking Underground Storage Tank Assessment Report Series

A series of research reports is planned to present data and models for leaking underground storage tank risk assessments. To date these include:

1. Gasoline Composition

Weaver, James W., Lewis Jordan and Daniel B. Hall, 2005, Predicted Ground Water, Soil and Soil Gas Impacts from US Gasolines, 2004: First Analysis of the Autumnal Data, United States Environmental Protection Agency, Washington, D.C., EPA/600/R-05/032.

2. Simulation Models

Gorokhovski, Vikenti M. and James W. Weaver, 2007, A Catalog of Ground Water Flow Solutions for Plume Diving Calculations, United States Environmental Protection Agency, Washington, D.C., EPA/600/R-07/122

Weaver, James W., 2004, On-line Tools for Assessing Petroleum Releases, United States Environmental Protection Agency, Washington, D.C., EPA 600/R-04/101.

3. Model Background and Evaluation

Weaver, James W. and C. S. Sosik, 2007, Assessment of Modeling Reports for Petroleum Release and Brownfields Sites, United States Environmental Protection Agency, Washington, D.C., EPA 600/R-07/101.

As more reports are added to the series, they may be found on EPA's web site at: https://cfpub.epa.gov/si/index.cfm.

Contents

1. Introduction

EPA and other organizations advocate improved approaches to site assessment (US EPA, 2003, 2004, ITRC, 2003). EPA's office of Underground Storage Tanks developed and published a framework for expedited site assessment (US EPA, 1997). Their purpose was to streamline corrective action at release sites, improve data collection and reduce the cost of remediation. Their approach emphasized a flexible sampling plan, field-generated data, on-site interpretation by senior staff working at the field sites.

Another widely adopted framework is called the Triad, which includes systematic planning of goals for all site activities, dynamic work strategies that allow for real-time decision making in the field, and real-time data gathering and assessment. EPA (2004) further elaborated that real-time measurement includes rapid sampling, geophysical analysis, and on-site data management software. One purpose of these technologies (EPA, 2003) is to allow for real-time development and refinement of the conceptual site model (CSM). Cheaper and faster site cleanups have been reported through use of the Triad approach (EPA 2005, 2006, ITRC, 2003).

This document presents the mathematical basis of software for real-time development and refinement of site conceptual models. The emphasis in the work is on evaluation of ground water flow patterns and the proper placement of vertical sample intervals.

The work focuses on the problem of "diving plumes", a term which generally refers to plumes that go deeper into aquifers with distance from their sources. As noted by Weaver and Wilson (2000), plumes may dive for several reasons: aquifer recharge supplying clean water above the plume, aquifer stratigraphy controlling the transport direction, relatively deep pumping causing downward gradients, and the possibility of oxygenated recharge water selectively enhancing biodegradation in the upper portion of plumes.

Weaver et al. (1996) presented results from a site on Long Island New York where a plume dived due to a local feature in the landscape (a gravel pit) and due to diffuse areal recharge. The first of these can contribute to localized recharge impacts and occurs over small horizontal distances. Weaver et al., (2002) reported on a trichloroethene plume that dove approximately 2 m as it emerged from below a paved parking lot. Runoff from roofs and a paved parking lot discharged into an unlined ditch that ran perpendicularly above the plume.

The second effect frequently requires either longer distances, longer times of transport or high rates of recharge to be active and thus is typically a greater concern for more mobile, less readily biodegradable compounds like methyl tert-butyl ether (MtBE) (e.g., Dernbach, 2000). Diving plumes were evident in the data presented for the Borden Landfill (MacFarlane et al., 1983) and the USGS Cape Cod field site (LeBlanc et al., 1991). These effects are consequences of the hydrologic system, which can produce both downward and upward flows. Landmeyer et al. (1998) presented data from a gasoline release site where an MTBE plume followed the regional flow pattern, first diving deeper into the aquifer, and further down-gradient, becoming shallower as the ground water reached a surficial discharge point. Wilson et al. (2000) presented results from a degrading MTBE plume where the maximum concentrations were co-located with the maximum hydraulic conductivities in a vertical section, illustrating important aspects of stratigraphic variation. Wilson et al. (2005) evaluated a site where plume diving was controlled by geochemical conditions that prevented biodegradation of methyl tert-butyl ether (MTBE) and a flow path dominated by stratigraphic differences in aquifer materials. Nichols and Roth (2006) reviewed the various factors that may contribute to plume diving, and suggests methods for evaluating the potential for plume dive at a site that are based on the average recharge rate and flow rate in the aquifer.

Part of the problem with diving plumes is the potential for mischaracterization of sites and the potential to miss downgradient contamination. An interactive example was provided by Weaver (2004) on the EPA web site.[1] This example shows that the placement and the length of a well screen (or shorter sampling interval) plays important roles in determining the observed concentration of a contaminant. A screen placed above a contaminant plume may find little or no contamination and a long screen may sample waters with varying contaminant concentrations, so that the mixed value is lower than the maximum. Therefore well-bore dilution and screen placement are equally important in proper characterization.

Complex ground water flow and transport models have been developed for simulation of contaminant transport problems. The need for these models is self-evident from the complex three-dimensional nature of the subsurface environment. These models, however, require much data and their application depends on the availability of extensive calibration data. For many cases, and for early stages of site characterization, far fewer data are available for plume and aquifer delineation and for parameter adjustment during calibration. Given these problems, the utility of models under these circumstances should be carefully considered. Conversely, "screening" is often said to be an objective of simplified models with limited data. For a simplified model to serve as a screening model, it first must be demonstrated that the model has the ability to simulate all significant phenomena at the site and be sufficient for the suggested purpose. In this paper we develop a simplified model, we compare it against a more complex model and field results, and finally provide a suggestion on how to incorporate the model into site characterization activities.

Bear (1972) summarized one-dimensional steady-state solutions for unconfined flow in homogeneous aquifers for flow between two aquifers, radial flow, flow on an inclined base, flow in horizontal and vertically stratified aquifers, and flow with recharge. See also Polubarinova – Kochina, 1962 and Strack, 1989. These solutions invoke the Dupuit-Forchheimer assumption that equipotentials are vertical. From these solutions, Weaver (2004) developed a method for plume diving estimation that uses piece-wise application of analytical solutions to allow for non-uniform recharge and aquifer

[1] http://www.epa.gov/athens/onsite

properties on a horizontal aquifer base. Analytical solutions for transient flow in aquifers with sloping bases and various boundary conditions were developed by Childs (1971), Brutsaert (1994), Telyakovskiy and Allen (2006), and for problems including recharge by Chapman (1994), Verhorst and Troch (2000), and Hantush and Marino (2001). Work in the area of hillslope hydrology has focused on the problem of flow over an uneven aquifer base with recharge from a three-dimensional soil block, representing flow in a catchment to a stream or river. An approach to reduce the geometric complexity to a one-dimensional form was developed by Fan and Bras (1998) and has been expanded (Verhorst and Troch (2000), Troch et al. (2002)) to allow for variable base slopes (Hilberts et al., 2004) with constant recharge. See Hilberts et al., 2004, especially, for a review of developments in this area. Most recently, Steward (2007) presented a piecewise heterogeneous aquifer solution for an aquifer with a stepped base.

This document contains a series of solutions of ground water flow for plume diving calculations. The resulting models (see http://www.epa.gov/athens/onsite) are intended for use in planning and conducting site assessments. An important question for these activities is "At what depth should the sample be taken?" The solutions start with a simple solution of one-dimensional ground water flow in an aquifer with an horizontal base. Boundary conditions can be applied in the traditional way at the ends of the domain, but to provide flexibility solutions that provide a means to set internal boundary conditions were developed. This approach is crucial to the assessment strategy developed by Weaver and Gorokhovski (2007).

Beyond these developments, the methods can be extended for piecewise heterogeneous aquifers (see Weaver, 2004). This approach greatly increases the flexibility of the solutions as parameters can be varied along the flow domain.

The two greatest weaknesses in the approach are the need to specify recharge rates and the restriction to an aquifer with a flat base. An alternative procedure can be developed that allows measured water levels in the aquifer to substitute for recharge rates. When an uneven base is required for an aquifer, the most advantageous approach is to use a numerical model. Thus there can be flexible specification of hydraulic conductivity, the recharge rate, and the base of the aquifer, on as small a scale as needed. The numerical model was tested against a simple homogeneous flow system and a piecewise heterogeneous system. The errors were insignificant for the water table elevation and the upper bound of the contaminant plume for both cases.

The document is organized by solution. The basics of the one-dimensional model are presented first. Solutions are then developed for flow with three sets of boundary conditions. The methodology is then extended to piecewise heterogeneous domains that allow for more flexibility in the solution. A type of inverse problem is solved that uses measured heads as a substitute for the recharge rate. The analytical method is extended to aquifers with sloping bases. Lastly a numerical model is presented for heterogeneous aquifers with uneven bases. The equations for streamlines that define plume diving and travel time are given. A comparison is made to solutions for a homogeneous aquifer and a piecewise heterogeneous aquifer.

2. Basic Model

The basic model describes ground water flow in a unconfined (phreatic) aquifer with a horizontal base. Figure 2.1 illustrates flow according to the standard approach for a Dupuit-Forchheimer model (see Bear, 1972, page 361). Assuming that water and soil are incompressible and the validity of the Conservation Law, we can write for a small interval [$\boldsymbol{x, x+\Delta x}$] (Figure 2.1) the following balance equation:

(2.1) $\boldsymbol{\Delta Q = N\,\Delta x}$

where $\boldsymbol{\Delta Q}$ is the change in ground water flow per unit width over the segment $\boldsymbol{\Delta x}$ [$L^3/L/T$], $\boldsymbol{N}$ is the recharge rate [$L^3/L^2/T$].

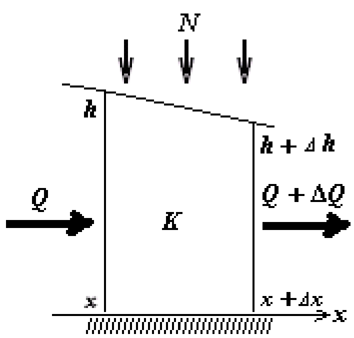


Figure 2.1. Schematic for deducing the ordinary differential equation for one-dimensional, steady-state ground water flow.

To introduce the thickness of the unconfined aquifer, $\boldsymbol{h}$ [L] into the model, we assume that the flow is horizontal, and specific discharge $\boldsymbol{q}$ [$L^3/L^2/T$] does not depend on depth (the Dupuit – Forchheimer assumption or approximation). Then we can rewrite *Equation 2.1* as

(2.2) $\boldsymbol{\Delta(qh) = N\,\Delta x}$

Making transition to limits with $\boldsymbol{\Delta x}$ going to zero, we obtain the ordinary differential equation (ODE)

$$\frac{d(qh)}{dx} = N \tag{2.3}$$

Equation 2.3 includes two unknown variables $\boldsymbol{q}$ and $\boldsymbol{h}$. To exclude one of them, we have to relate them. So our next assumption is Darcy's Law relating specific discharge and hydraulic gradient:

(2.4) $$\boldsymbol{q = -K\frac{dh}{dx}}$$

where $\boldsymbol{K}$ is the hydraulic conductivity [L/T]. Substituting the right hand side of *Equation 2.4* for $\boldsymbol{q}$ in *Equation 2.3*, we finally obtain the basic equation governing one-dimensional horizontal phreatic flow:

(2.5) $$\boldsymbol{\frac{d\left(Kh\frac{dh}{dx}\right)}{dx} = -N}$$

When solved for $\boldsymbol{h(x)}$, it yields thickness of aquifers with free water tables. The assumptions fit best aquifers with small absolute values of the water table gradients and not very low permeability.

2.1. Assumptions and Limitations

This governing equation is based on a conceptual model consisting of an unconfined aquifer where the base is defined. The perceived saturated thickness of the aquifer (h) depends on knowledge of the aquifer base. In practice this would be defined through coring of the aquifer. In cases where the aquifer base is not clearly definable, then a series of simulations are needed to account for uncertainty in thickness.

The analytical solutions presented in Sections 3 through 6 require the assumption of a horizontal aquifer base and homogeneous aquifer properties. Section 7 describes a model where the aquifer base can slope in an arbitrary manner. If the aquifer hydraulic conductivity varies over the vertical, this parameter can be averaged and the average value used in the solution (Bear, 1972, page 376) for ground water flows. Other standard assumptions apply: Darcy's Law is valid, aquifer consolidation is ignored and ground water is incompressible.

Flow is one-dimensional in the conceptual model. For single-layer systems, as here, vertical flows are unimportant in Dupuit-Forchheimer models. These correspond to field situations where the length of the flow system is at least two times greater than the aquifer thickness (Bear, 1972, page 365), and there are no cones of depression or seepages faces in the area of interest. Thus flow is assumed to occur in a horizontal plane and as noted above only one of the two remaining dimensions are simulated. The second of these is explicitly ignored, so that the models do not reproduce multidimensional behavior as would occur in well fields, for example. Along a stream line, however, the models applies. The key to field application is the delineation of the streamline pathway. In other cases the models may still be useful, particularly for high velocity aquifers where flows are dominant in one direction. In other cases, the one-dimensional simulation may give a ball-park estimate of plume diving, that is subject to verification and possible refinement through additional data collection and model application.

3. Homogeneous Aquifer with a Horizontal Base and Uniform Recharge

In the case of homogeneous aquifers, that is, aquifers with uniform permeability, *Equation 2.5* can be written as

(3.1) $$\frac{d\left(h\frac{dh}{dx}\right)}{dx} = -\frac{N}{K}$$

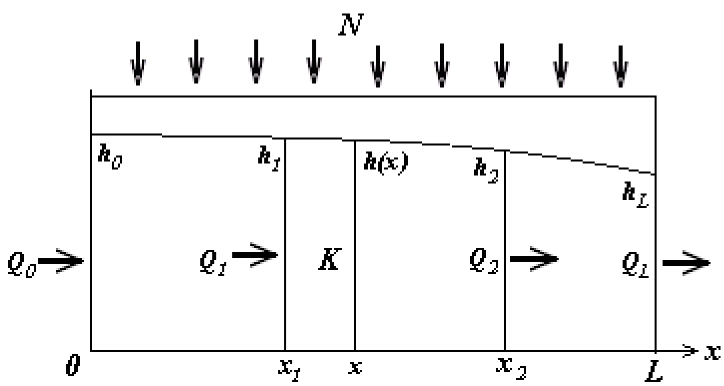


Figure 3.1: One ─ dimensional steady state flow on interval **[*0, L*]**

With ***K*** and ***N*** constant, the general solution to *Equation 3.1* is

(3.2) $$h^2 = -W\ x^2 + bx + c$$

where ***b*** and ***c*** are arbitrary constants and ***W*** is the quotient ***N/K***. To specify their values, we need to impose two boundary conditions. Substituting in *Equation 3.2* that at ***x = 0, h = h₀***

(3.2a) $$c = h_0^2 .$$

For the second coefficient, ***b***, we differentiate *Equation 3.2* and multiply the result by ***K***:

$$2hK\frac{dh}{dx} = -2Nx + Kb$$

Where we have made use of the relationship

$$\frac{d(h^2)}{dx} = 2h\frac{dh}{dx}$$

Again substituting $x = 0$ in the above equation reveals that

(3.2b) $2hK\frac{dh}{dx} = Kb$ or $b = -2\frac{Q_0}{K}$

where Q_0 is the flow per unit width of the aquifer at $x = 0$.

There are three possible ways to assign boundary conditions for *Equation 3.1* (or *Equation 3.2*) for an arbitrary interval $[x_1, x_2]$.

3.1. Case 1: Two head boundaries

Boundary conditions are given as the thickness of the aquifer at the ends of the segment of interest: $h(x_1) = h_1$ and $h(x_2) = h_2$. The solution is (Bear, 1972, p.380):

(3.3) $$h^2 = h_1^2 - \left(\frac{h_1^2 - h_2^2}{x_2 - x_1} - W(x_2 - x_1)\right)(x - x_1) - W(x - x_1)^2$$

where $W = \frac{N}{K}$. By noting the similar form of Equations 3.2 and 3.3, than

$$c = h_1^2$$

$$b = -2\frac{Q_1}{K}$$

3.2. Case 2: One head and One Flux Boundary

Boundary conditions are given at $x = x_1$ as h_1 and at x_2 as $Kh\frac{dh}{dx}\Big|_{x_2} = -Q_2$. The first integration of *Equation (3.1)* yields

(3.4) $$Kh\frac{dh}{dx} = -N(x - x_1) + C$$

It follows from *Equation 3.4* that

$$-Q_2 = -N(x_2 - x_1) + C\,.$$

and

(3.4a) $C = -Q_2 + N(x_2 - x_1) = -Q_1$

With condition $h(x_1) = h_1$, we finally obtain

(3.5) $h^2 = h_1^2 - 2\dfrac{Q_1}{K}(x - x_1) - W(x - x_1)^2$

3.3. Case 3: Two Flux Boundaries

Boundary conditions are given as $Kh\dfrac{dh}{dx}\Big|_{x_1} = -Q_1$ and $Kh\dfrac{dh}{dx}\Big|_{x_2} = -Q_2$. In this formulation, Equation 3.1 does not have a unique solution, since the above boundary conditions do not permit obtaining coefficient ***c*** in *Equation 3.2*. Therefore, these boundary conditions are excluded from further consideration.

3.4. Extrapolation from a small interval to a large domain

The above *Cases 1*and *2* are solutions for the interval $[x_1, x_2]$ chosen arbitrarily within the interval of interest: $[0, L]$. However, the above solutions, or speaking more exactly, their boundary conditions, permit obtaining solution for interval $[0, L]$. To this end, let us consider the case when the boundary conditions are given at one location as $h_2 = h(x_2)$ and

$$Kh\frac{dh}{dx}\Big|_{x_2} = -Q_2$$

Based on *Equation 3.5*, we can write

(3.6) $h_2^2 = h_1^2 - 2\dfrac{Q_1}{K}(x_2 - x_1) - W(x_2 - x_1)^2$

Therefore,

$$h_1^2 = h_2^2 + 2\frac{Q_1}{K}(x_2 - x_1) + W(x_2 - x_1)^2$$

with $Q_1 = Q_2 - N(x_2 - x_1)$. The head at the left hand boundary, h_0, can be found from

$$h_0^2 = h_2^2 + 2\frac{Q_0}{K}x_2 + Wx_2^2$$

with $Q_0 = Q_2 - Nx$.

Thus, if boundary conditions are assigned as in Case 2, the problem for arbitrary interval **[x_1, x_2]** can be extended on interval **[$0, L$]** easily. If the boundary conditions are given as in Case 1, it can be reduced to Case 2 first, using, for example, equation

(3.7) $$Q_1 = \frac{K}{2(x_2 - x_1)}\left(h_1^2 - h_2^2\right) - \frac{N(x_2 - x_1)}{2}$$

Then the solution can be extended on the interval **[x_1, x_2]**.

3.5. *Flows with a Ground Water Divide*

In the case where Q_0 and Q_n have different signs ($Q_0 Q_n < 0$), there must be a water divide within the segment of interest [$0, L$]. It is at the location where flux is equal to zero: $Q_0 - Nx_{WD} = 0$. The location can be evaluated using the following formula (Bear, 1972):

(3.8) $$x_{WD} = \frac{L}{2} - \frac{h_0^2 - h_L^2}{2WL}$$

The Matlab Code ‘D1_Methodology’ implementing the above solution and example applications are presented in Appendix 1.

4. Piecewise Aquifer with a Horizontal Base

In the context of this work, heterogeneity of an aquifer means that either the hydraulic conductivity, or recharge, or both depend on the x-coordinate. Because of relative sparseness of available data, such aquifers are considered here as being piecewise homogeneous. That is, within a homogeneous segment of the aquifer, both the hydraulic conductivity and the recharge are constant. In each aquifer segment shown in Figure 4.1, ground water flow is described by equation

(4.1) $$\frac{d\left(K_i\ {}_ih\frac{d\,{}_ih}{dx}\right)}{dx} = -N_i$$

where the leading subscript i refers to an individual segment. To elaborate further, the notation for this section uses a leading subscript to denote the function of head in a given segment i: ${}_ih$. The trailing subscript refers to a head value at a given location: $h_i = h(x_i)$.

To obtain an unique solution to *Equation 4.1*, we use the usual (external) boundary conditions, for instance $h_0 = h(0)$ and $h_L = h(L)$, but also internal boundary conditions between homogeneous sections of the aquifer (Weaver, 2004, appendix 3)

(4.2a) $\quad {}_{i-1}h(x_i) = {}_i h(x_i)$

(4.2b) $\quad K_{i-1}\; {}_{i-1}h \left.\dfrac{d\,{}_{i-1}h}{dx}\right|_{x=x_i} = K_i\; {}_i h \left.\dfrac{d\,{}_i h}{dx}\right|_{x=x_i}$

These conditions express the continuity of the water table and the flux (the Law of Conservation).

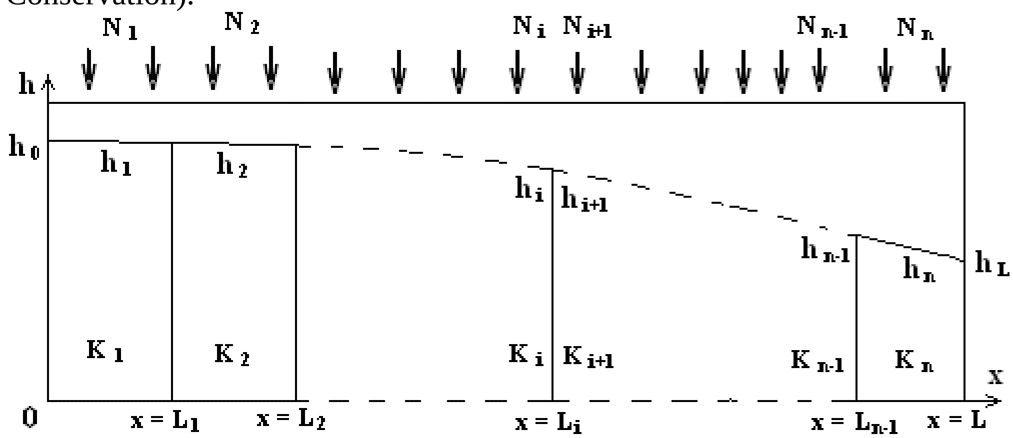


Figure 4.1: Heterogeneous aquifer on a horizontal base.

Note that the object consisting of n sections has $2(n-1)$ internal boundaries and two external ones at $x=0$ and $x=L$.

4.1. Solution Per Aquifer Segment

The solution to *Equation 4.*1 within a homogeneous interval $[x_{i-1}, x_i]$, within *the* i^{th} segment, the flow is described by equation

(4.3) $\quad {}_i h^2(x) = -W_i\left(x - L_{i-1}\right)^2 + A_{2i-1}\left(x - L_{i-1}\right) + A_{2i-2}$

where $W_i = \dfrac{N_i}{K_i}$, and A_{2i-1} and A_{2i-2} are unknown coefficients.

Thus for the first segment, we have

$${}_1 h^2(x) = -W_1\left(x - L_0\right)^2 + A_1\left(x - L_0\right) + A_0$$

Since $L_0=0$ and using the boundary condition at $x = 0$, we can rewrite the above equation as

$$ {}_1h^2(x) = -W_1 x^2 + A_1 x + h_0^2 $$

For the second segment, we have the equation

$$ {}_2h^2(x) = -W_2 \left(x - L_1\right)^2 + A_3 \left(x - L_1\right) + A_2 $$

For the third segment, we have the equation

$$ {}_3h^2(x) = -W_3 \left(x - L_2\right)^2 + A_5 \left(x - L_1\right) + A_4 $$

And so on. To find unknown coefficients A_i, the *Internal Conditions (4.2)* can be used. Thus, the following equations apply to the first inner boundary:

$$ -W_1 D_1^2 + A_1 D_1 + h_0^2 = A_2 $$
$$ -2N_1 D_1 + K_1 A_1 = K_2 A_3 $$

where $D_1 = L_1 - L_0 = L_1$. For the second inner boundary, we get the following equations:

$$ -W_2 D_2^2 + A_3 D_2 + A_2 = A_4 $$
$$ -2N_2 D_2 + K_2 A_3 = K_3 A_5 $$

where $D_2 = L_2 - L_1$.

For the inner boundary number i, there are the two following equations:

$$ -W_i D_i^2 + A_{2i-1} D_i + A_{2i-2} = A_{2i} $$
$$ -2N_i D_i + K_i A_{2i-1} = K_{i+1} A_{2i+1} $$

where $D_i = L_i - L_{i-1}$.

Only one equation is associated with the last boundary number n:

$$ -W_n D_n^2 + A_{2n-1} D_n + A_{2n-2} = h_L^2 $$

In this equation, the boundary condition at $x = L$ is used. Thus we have $2n-1$ equations to find $2n-1$ unknown coefficients A_i.

Let us rewrite the above equations in the system with known terms put in the right hand side and terms including unknowns in the left hand side. Besides we group them: First n equations presents *Boundary Condition 4.2a* and following $n-1$ equation represent *Boundary Condition 4.2b:*

$$
(4.4)\quad
\begin{array}{lll}
1: & -D_1A_1 + A_2 & = h_0^2 - W_1D_1^2 \\
2: & A_2 + D_2A_3 - A_4 & = W_2D_2^2 \\
\text{-----} & \text{-----------------} & \text{----------} \\
i: & A_{2i-2} + D_iA_{2i-1} - A_{2i} & = W_iD_i^2 \\
\text{-----} & \text{-----------------} & \text{----------} \\
n: & A_{2n-2} + D_nA_{2n-1} & = h_L^2 + W_nD_n^2 \\
n+1: & K_1A_1 - K_2A_3 & = 2N_1D_1 \\
n+2: & K_2A_3 - K_3A_5 & = 2N_2D_2 \\
\text{-----} & \text{-----------------} & \text{----------} \\
n+i: & K_iA_{2i-1} - K_{i+1}A_{2i+1} & = 2N_iD_i \\
\text{-----} & \text{-----------------} & \text{----------} \\
n+n-1: & K_{n-1}A_{2n-3} - K_nA_{2n-1} & = 2N_{n-1}D_{n-1}
\end{array}
$$

Solution of *System 4.4*, coefficients A_i, is the sets of coefficients of *Equations 4.3* for every homogeneous segment.

4.2. Sequential Solution Method for Specified Flux Boundary

Let the external boundary conditions are assigned as $h_0 = h(0)$ and $Q_0 = Q(0)$, the flux at $x = 0$. Then within the first homogeneous segment the flow is governed by *Equation 3.8* that takes the form

$$(4.5)\quad {}_1h^2 = h_0^2 - 2\frac{Q_0}{K_1}x - W_1x^2$$

where $W_1 = \frac{N_1}{K_1}$. (Thereafter $W_i = \frac{N_i}{K_i}$.)

For the second segment, we have

$${}_2h^2 = h_1^2 - 2\frac{Q_1}{K_2}(x - L_1) - W_2(x - L_1)^2$$

Due to the internal *Boundary Conditions 4.2*,

$$h_1^2 = {}_1h(L_1)^2 \text{ and } Q_1 = Q(L_1) = Q_0 + N_1L_1$$

For the segment # i, we have

$$(4.6)\quad {}_ih^2 = h_{i-1}^2 - 2\frac{Q_{i-1}}{K_i}(x - L_{i-1}) - W_i(x - L_{i-1})^2$$

where

(4.7) $\boldsymbol{h}^2_{i\text{-}1} = {}_{i\text{-}1}\boldsymbol{h}(\boldsymbol{L}_{i\text{-}1})^2$ and $Q_{i\text{-}1} = Q(\boldsymbol{L}_{i\text{-}1}) = Q_{i\text{-}2} + N_{i\text{-}1}(\boldsymbol{L}_i - L_{i-1})$

Equations 4.5 and *4.6* describe a procedure that does not involve solving *System 4.4*. Starting with the most left boundary we obtain the formula for the first segment. Using this formula, we calculate the squared thickness of the aquifer at the end of the first segment. This squared thickness is the left boundary condition for the second segment. The flux at this boundary is $\boldsymbol{Q_1 = Q_0 + N_1 (L_2 - L_1)}$.

This algorithm is implemented in the code 'hrzAQT' presented in Appendix 2.

The code 'hrzAQT' works only within intervals [$\boldsymbol{x_1, x_2}$] for which thickness of aquifers at their ends, $\boldsymbol{h_1 = h_1}$ and $\boldsymbol{h_2 = h_2 (x_2)}$, are known. However in most cases, observations at the ends of the objects of interest are not available. Therefore, we need tool to overcome this obstacle. The numerical 'hrzD1_Unvrs' is developed to this end.

Code 'hrzD1_Unvrs1' permits solving system *of Equations 4.1* with internal boundary conditions presented by *Equations 4.2* and for two sets of 'outer' boundary conditions numerically:

(4.8a) $\boldsymbol{h(x_a)} = h_a$ and $\boldsymbol{h(x_b)} = h_b$, $\qquad \boldsymbol{x_1 \le x_a \le x_2}, \quad \boldsymbol{x_1 \le x_b \le x_2} \qquad x_1 \neq x_2$

(4.8a) $\boldsymbol{h(x_a) = h_a}$ and $\boldsymbol{Kh\frac{dh}{dx}\Big|_{x_b} = -Q_b}$, $\qquad \boldsymbol{x_1 \le x_a \le x_2}, \quad \boldsymbol{x_1 \le x_b \le x_2}$

These two sets of *Conditions 4.8* correspond to cases implemented in Code 'UnvhrzD1_Unvrs'**.**

4.3. Case 1: Two head boundaries

Equations 4.8a are used as boundary conditions. The lesser of $\boldsymbol{x_a}$ and $\boldsymbol{x_b}$ is denoted as $\boldsymbol{x_a}$ and the greater as $\boldsymbol{x_b}$. ($\boldsymbol{h_a}$ and $\boldsymbol{h_b}$ are also exchanged if necessary.) Then an auxiliary object, bounded by $\boldsymbol{x_a}$ and $\boldsymbol{x_b}$ is made up. For this object, system of Equations (4.4) is developed and solved. That permits calculating flux at point $\boldsymbol{x_a}$, $\boldsymbol{Q_a}$. Knowing $\mathbf{Q}(\boldsymbol{x_a})$ permits calculating influx $\boldsymbol{Q}$ at every segment boundary of the object. For example, if $\boldsymbol{x_a}$ belongs to segment $\boldsymbol{j}$, then

(4.9) $\boldsymbol{Q_1 = Q_a + N_j(L_j - x_a) + \sum_{j-1}^{2} N_{j-1}(L_j - L_{j-1})}$

Having known the flux at the ends of the segments, we can calculate $\boldsymbol{h_j^2 = h_j^2(L_j)}$. Thus, if $\boldsymbol{x_a}$ belongs to segment number $\boldsymbol{j}$, then

(4.10) $h_j^2 = h_a^2 + 2\frac{Q_j}{K_j}\left(x_a - L_j\right) + W_j\left(x_a - L_j\right)^2$

(4.11) $h_{j-1}^2 = h_j^2 + 2\frac{Q_{j-1}}{K_{j-1}}\left(L_j - L_{j-1}\right) + W_{j-1}\left(L_j - L_{j-1}\right)^2$

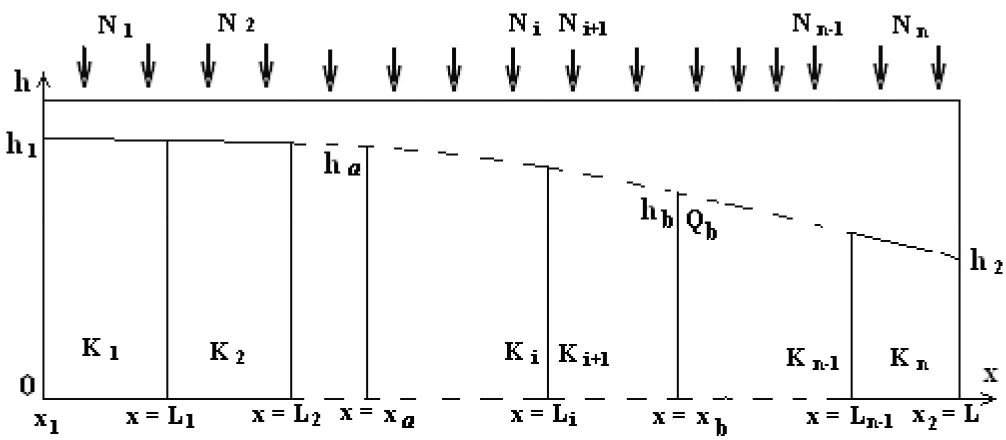


Figure 4.2: Heterogeneous aquifer with a horizontal base and two sets of boundary conditions

And so on, until h_1^2 will be obtained. Then we can calculate $h^2(x)$ for any x based on Equation 4.6 that we rewrite here as

(4.12) ${}_jh^2 = h_j^2 - 2\frac{Q_j}{K_j}\left(x - L_{ij}\right) - W_j\left(x - L_j\right)^2$

4.4. Case 2: One head and One Flux Boundary

Equations 4.8a are used as boundary conditions. The case differs from Case 1 with the fact that we do not need to make up an auxiliary object, develop system of equations like *Equations 4.4*, and solve it. Using knowledge of Q_b at x_b, we can immediately calculate the flux at the ends of the segments, using procedure described by an equation like *Equation 4.9*, and then *Equations 4.10* and *4.11*.

A listing of the code ‘hrzAQT’ and all included scripts providing the analytical solution to the above problem are presented in Appendix 2. A listing of the code ‘hrzD1_Unvrs’ and all included scripts providing the numerical solution to the above problem are presented in Appendix 3. This code is recommended as more flexible.

5. Inverse Problems of One-Dimensional Flow in an Unconfined Aquifer with a Horizontal Base

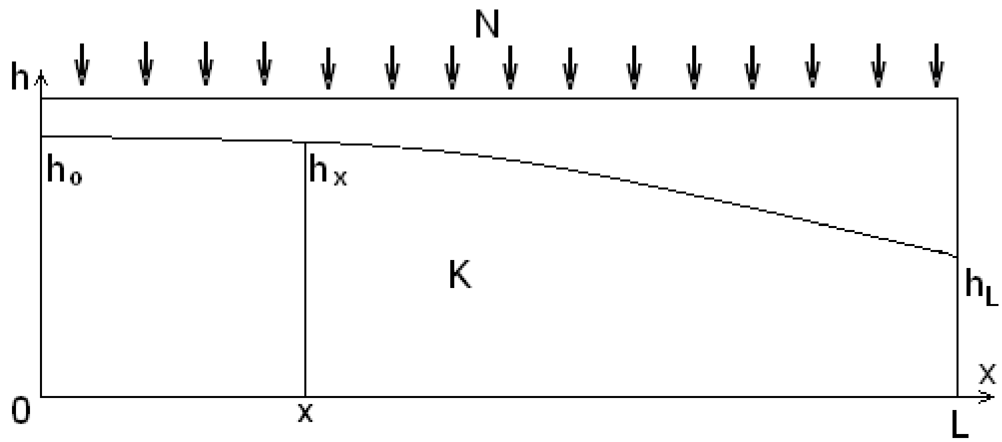


Figure 5.1: Homogeneous aquifer with a horizontal base

Case 1: Homogeneous Site One-dimensional ground water flow is described by equation

$$\textbf{(5.1)} \qquad \frac{d\left(Kh\frac{dh}{dx}\right)}{dx} = -N$$

With ***K*** and ***N*** constant, the general solution is

$$\textbf{(5.2)} \qquad h^2 = -\frac{N}{K}x^2 + bx + c$$

where ***b*** and ***c*** are arbitrary constant. Their choice depends on the boundary conditions (Bear, p.379).

5.1. Case 1a: Two Head Boundaries

The boundary conditions are given as $h(0)=h_0$ and $h(L)=h_L$. Then

$$\textbf{(5.3)} \qquad h^2 = h_0^2 - \left(\frac{h_0^2 - h_L^2}{L} - \frac{N}{K}(L-x)\right)x \qquad \text{(Bear, p.380)}$$

Differentiating *Equation 5. 3* and multipling the result by ***K/2***, we obtain:

$$(5.4) \qquad Kh\frac{dh}{dx} = -K\frac{h_0^2 - h_L^2}{2L} + \frac{N\,L}{2} - N\,x$$

Equation 5.4 can be used to express the discharge (flow) at location ***x*.** Thus at ***x = 0*** the discharge entering the site, Q_0, is

$$(5.5) \qquad Q_0 = -Kh\left.\frac{dh}{dx}\right|_0 = K\frac{h_0^2 - h_L^2}{2L} - \frac{N\,L}{2}$$

Therefore the incoming discharge can be estimated if ***N*** and ***K*** are known.

For location ***x = L***, *Equation 5.4* gives

$$(5.6) \qquad Q_L \; - \; Kh\left.\frac{dh}{dx}\right|_L = K\frac{h_0^2 - h_L^2}{2L} + \frac{N\,L}{2}$$

So two known heads do not supply sufficient information to estimate ***N*** and ***K***, as the difference of two discharges yields a trivial result:

$$Kh\left.\frac{dh}{dx}\right|_L - Kh\left.\frac{dh}{dx}\right|_0 = -N\,L$$

5.2. Case 1b: Three Known Heads

The head is known at three locations:

$$h(0) = h_0 \qquad h(L_1) = h_1 \qquad h(L_2) = h_2$$

With these three known heads, two equations can be written to describe flow:

$$(5.7a) \qquad h^2 = h_0^2 - \left(\frac{h_0^2 - h_1^2}{L_1} - \frac{N}{K}(L_1 - x)\right)x$$

$$(5.7b) \qquad h^2 = h_0^2 - \left(\frac{h_0^2 - h_1^2}{L_1} - \frac{N}{K}(L_2 - x)\right)x$$

There are also two equations for the discharge at location ***x = 0***:

$$(5.8a) \qquad Q_0 = -Kh\left.\frac{dh}{dx}\right|_0 = K\frac{h_0^2 - h_1^2}{2L_1} - \frac{N\,L_1}{2}$$

$$(5.8b) \qquad Q_0 = -Kh\left.\frac{dh}{dx}\right|_0 = K\frac{h_0^2 - h_1^2}{2L_2} - \frac{N\,L_2}{2}$$

Equations 5.8 permit finding ratio $\frac{N}{K}$ from the equality

$$\textbf{(5.9)} \quad -K\frac{h_0^2 - h_1^2}{2L_1} + \frac{N L_1}{2} = -K\frac{h_0^2 - h_2^2}{2L_2} + \frac{N L_2}{2}$$

Thus,

$$\textbf{(5.10)} \quad \frac{N}{K} = \frac{\dfrac{h_0^2 - h_1^2}{L_1} - \dfrac{h_0^2 - h_2^2}{L_2}}{(L_1 - L_2)} = \frac{\left(h_0^2 - h_1^2\right)L_2 - \left(h_0^2 - h_2^2\right)L_1}{L_1 L_2 (L_1 - L_2)}$$

5.3. Case 1c: A General Approach to Case 1b

General approach to *Case 1b*. A general solution of *Equation 5.1* can be written as

$$\textbf{(5.11)} \quad h^2 = a + bx + cx^2$$

To find coefficients ***a, b,*** and c, it suffices to know value of the above function at three points

$$h(0) = h_0 \qquad h(L_1) = h_1 \qquad h(L_2) = h_2$$

Since $\boldsymbol{h = h_0}$ at $\boldsymbol{x = 0,}$ the coefficient $\boldsymbol{a}$ is immediately known as being equal to $\boldsymbol{h_0^2}$. There are two equations for the remaining unknown coefficients, $\boldsymbol{b}$ and $\boldsymbol{c}$:

$$\textbf{(5.12)} \quad \begin{aligned} bL_1 + cL_1^2 &= h_1^2 - h_0^2 \\ bL_2 + cL_2^2 &= h_2^2 - h_0^2 \end{aligned}$$

Solution of *System 5.12* is given by determinants:

$$\textbf{(5.13)} \quad b = \frac{\begin{vmatrix} h_1^2 - h_0^2 & L_1^2 \\ h_2^2 - h_0^2 & L_2^2 \end{vmatrix}}{\begin{vmatrix} L_1 & L_1^2 \\ L_2 & L_2^2 \end{vmatrix}}, \qquad c = \frac{\begin{vmatrix} L_1 & h_1^2 - h_0^2 \\ L_2 & h_2^2 - h_0^2 \end{vmatrix}}{\begin{vmatrix} L_1 & L_1^2 \\ L_2 & L_2^2 \end{vmatrix}}$$

or

$$b = \frac{\left(h_1^2 - h_0^2\right)L_2^2 - \left(h_2^2 - h_0^2\right)L_1^2}{L_1 L_2\left(L_2 - L_1\right)}$$

(5.14)

$$c = \frac{\left(h_2^2 - h_0^2\right)L_1 - \left(h_1^2 - h_0^2\right)L_2}{L_1 L_2\left(L_2 - L_1\right)} = -\frac{\left(h_0^2 - h_1^2\right)L_2 - \left(h_0^2 - h_2^2\right)L_1}{L_1 L_2 (L_1 - L_2)}$$

Equation 5.14 shows that the unknown coefficients of *Equation 5.1* can be determined without knowledge of the recharge rate, N, and the hydraulic conductivity, K. The physical meanings of the coefficients b and c can be determined from *Equation 5. 3* which is slightly rewritten as

$$\textbf{(5.15)}\quad h^2 = h_0^2 - \left(\frac{h_0^2 - h_L^2}{L} - \frac{N}{K}L\right)x - \frac{N}{K}x^2$$

Thus,

$$\textbf{(5.16}\boldsymbol{a}\textbf{)}\qquad b = \frac{N}{K}L - \frac{h_0^2 - h_L^2}{L}, \qquad c = -\frac{N}{K}.$$

Differentiating *Equation 11*, we can also see that

$$\textbf{(5.16}\boldsymbol{b}\textbf{)}\ b = 2h_0 \left.\frac{dh}{dx}\right|_0 \qquad or \qquad Q_0 = -K\frac{b}{2} \qquad .$$

where here Q_0 is the discharge at $x = 0$.

It must be noted that because *Equation 5.1* is governed by the dimensionless parameter $\frac{N}{K}$, so the values of the individual parameters N and K cannot be identified in the above cases. To do so, knowledge of discharge at any one location is necessary.

Equation 5.16b can be used for finding gradient of the water table at $x = 0$. Additional options are given by *Equations 5.17* following from *Equations 5.5* and *5.6*:

$$\textbf{(5.17}\boldsymbol{a}\textbf{)}\ \left.\frac{dh}{dx}\right|_0 = \frac{1}{2}\left(\frac{N}{K} * \frac{L}{h_0} - \frac{h_0^2 - h_L^2}{h_0 L}\right) = -\frac{1}{2}\left(\frac{cL}{h_0} + \frac{h_0^2 - h_L^2}{h_0 L}\right)$$

$$\textbf{(5.17}\boldsymbol{b}\textbf{)}\ \left.\frac{dh}{dx}\right|_L = -\frac{1}{2}\left(\frac{N}{K} * \frac{L}{h_L} + \frac{h_0^2 - h_L^2}{h_L L}\right) = \frac{1}{2}\left(\frac{c\,L}{h_L} - \frac{h_0^2 - h_L^2}{h_L L}\right)$$

The gradient of the water table in any location x inside the interval with known boundary conditions follows from *Equation 5.4* as

$$\textbf{(5.18)}\quad \frac{dh}{dx}=\frac{1}{2}\left(\frac{N}{K}\frac{L-x}{h}-\frac{h_0^2-h_L^2}{hL}\right)=-\frac{1}{2}\left(\frac{h_0^2-h_L^2}{2hL}+c\frac{L-x}{h}\right)$$

If there are more than three observed heads, *Equation 5.11* can be rewritten as a regression, and its coefficients (***a, b,*** and ***c***) can be evaluated as such. However, all three parameter governing flow (Q_0, ***K***, and ***N***) cannot be calculated from the regression. One must be determined independently.

5.4. Cases 1d to 1g: Combinations of Boundary Conditions

Case 1d: The boundary conditions are given as $h(0) = h_0$ and $Kh\left.\frac{dh}{dx}\right|_0=-Q_0$, that is the thickness and discharge are known at location $x = 0$.

The first integration of *Equation 5.1* yields

$$\textbf{(5.19)}\quad Kh\frac{dh}{dx}=-Nx+C$$

It follows from the second boundary condition that $C = -Q_0$. So *Equation 5.19* can be written as

$$\textbf{(5.20)}\quad Kh\frac{dh}{dx}=-Nx-Q_0$$

The second integration yields the final result:

$$\textbf{(5.21)}\quad h^2=h_0^2-2\frac{Q_0}{K}x-\frac{N}{K}x^2$$

Note. If we assume that *Equations 5.11* and 5.*21* describe the same system, we can see again that

$$\textbf{(5.22)}\quad b=-2\frac{Q_0}{K}$$

Therefore if either the hydraulic conductivity, ***K***, or the discharge, Q_0, is known, then the other can be determined from

(5.23) $$2\frac{Q_0}{K}=\frac{\left(h_0^2-h_a^2\right)L_2^2-\left(h_0^2-h_2^2\right)L_1^2}{L_1L_2\left(L_2-L_1\right)}$$

In contrast, when we know values of h_0 and h_L only, we can use *Equation 5.3*. In this case

(5.24) $$2\frac{Q_0}{K}=-\frac{N}{K}L+\frac{h_0^2-h_L^2}{L}$$

and the parameter $\frac{N}{K}$ must be known also.

Case 1e: The boundary conditions are given as $h(0)=h_0$ and $Kh\left.\frac{dh}{dx}\right|_L=-Q_L$:

This case can be reduced to the previous one by calculating $Q_0=Q_L-NL$**.** Then *Equation 5.21* becomes

(5.25) $$h^2=h_0^2-2\frac{Q_L-NL}{K}x-\frac{N}{K}x^2$$

Case 1f: The boundary conditions are given as $h(L)=h_L$ and $Kh\left.\frac{dh}{dx}\right|_0=-Q_0$:

Integrating *Equation 5.1* and using the second boundary condition yields *Equation 5.19*. The second integration yields

(5.26) $$h^2=C-2\frac{Q_0}{K}x-\frac{N}{K}x^2$$

The arbitrary constant C is equal to

(5.27) $$C={h_L}^2+2\frac{Q_0}{K}L+\frac{N}{K}L^2$$

So finally we have

(5.28) $$h^2=h_L^2-2\frac{Q_0}{K}\left(x-L\right)-\frac{N}{K}\left(x^2-L^2\right)$$

Case 1g: The boundary conditions are given as $h(L)=h_L$ and $Kh\left.\frac{dh}{dx}\right|_L=-Q_L$:

This case can be reduced to the previous one by calculating $Q_0=Q_L-NL$**.** Then *Equation 5 27* becomes

(5.29) $$h^2=h_0^2-2\frac{Q_L-NL}{K}\left(x-L\right)-\frac{N}{K}\left(x^2-L^2\right)$$

6. Solution for One-Dimensional, Steady-State Flow in a Shallow Unconfined Aquifer with a Sloping Base

Differential equation describing one-dimensional steady- state flow in an unconfined shallow aquifer on an uneven base can be written based on the Dupuit assumptions as

(6.1) $$\frac{d}{dx}\left(K(x)(h(x)-y(x))\frac{dh(x)}{dx}\right)+N(x)=0$$

where ***h(x)*** is the elevation of the water table, ***K(x)*** is the hydraulic conductivity, ***N(x)*** is the recharge, and ***y(x)*** is the elevation of the aquifer base.

The first integral of Equation 6.1 is

(6.2) $$K(x)(h(x)-y(x))h'(x)=-\int_0^x N(x)dx-Q_0$$

where Q_0 is an arbitrary constant of integration. Its physical meaning is the flux at $x = 0$.

Equation 6.2 can be rewritten in terms of the thickness of the aquifer

(6.3) $b(x) = h(x) - y(x)$

In this case

(6.4) $h'(x) = y'(x) + b'(x)$

Thus, Equation 6.2 can be rewritten as

(6.5) $$K(x)b(x)(y'(x)+b'(x))=-\int_0^x N(x)dx-Q_0$$

6.1. Homogeneous Aquifer

Let us assume that we are looking for solution of Equation 6.1 (or Equation 6.5, that is the same) on interval [$0, L$] for a homogeneous aquifer. The aquifer is a homogeneous one if

$K(x) = K$= const. and $N(x) = N$ = const. We assume also that the aquifer base is changing linearly

(6.6) $y(x) = m\,x$.

Equation 6.6 means that $y' = m$ and that we assigned the elevation of the aquifer base equal to zero at $x = 0$.

Under these assumptions, we can rewrite Equation 6.5 as

(6.7) $$b(x)b'(x) + mb(x) = -\frac{Nx + Q_0}{K}$$

Equation 6.7 is not linear, but we can approximate it with a linear equation with respect to $b^2(x)$. Indeed,

(6.8) $$b(x)b'(x) + \frac{mb^2(x)}{b(x)} = \frac{d\left(b^2(x)\right)}{2dx} + \frac{m\left(b^2(x)\right)}{b(x)} \approx \frac{d\left(b^2(x)\right)}{2dx} + \frac{m\left(b^2(x)\right)}{\bar{b}}$$

where $\bar{b}$ is some value of the aquifer thickness from interval [$0, L$]. Substituting this approximation to the left hand side of Equation 6.7, we obtain

(6.9) $$\left(b^2\right)' + \frac{2m}{\bar{b}} b^2 \approx -2\frac{Nx + Q_0}{K}$$

Equation 6.9 is linear with respect to function $b^2 = b^2(x)$. Its generals is

(6.10) $$b^2 = -2e^{-\int_0^x \frac{2m}{\bar{b}} dx} \int_0^x \left(\frac{Nx + Q_0}{K} e^{\int_0^x \frac{2m}{\bar{b}} dx} \right) dx + C$$

Integrating exponents in Equation 10, we obtain the general solution of Equation 6.10 in the form

(6.11) $$b^2 = -2e^{-\frac{2m}{\bar{b}}x} \int_0^x \frac{Nx + Q_0}{K} e^{\frac{2m}{\bar{b}}x} dx + C$$

[Note that for $m = 0$, Equation 6.11 converts into the equation for the unconfined homogeneous aquifer on the horizontal base]

We need two boundary conditions to find the unknown arbitrary constants C and Q_0. We consider two cases here.

6.2. Case 1: Two Head Boundaries

The boundary conditions are assigned as the water table elevations at the ends of interval [$0, L$] for which we seek the solution that is

(B.C. 1.1) $b^2(0) = b_0^2$ and $b^2(L) = b_L^2$

Applying the condition at $x = 0$, we immediately obtain we can rewrite

$$\textbf{(6.12)}\quad b^2 = -2e^{-\frac{2m}{\bar{b}}x}\int_0^x \frac{Nx+Q_0}{K}e^{\frac{2m}{\bar{b}}x}dx + b_0^2$$

[Checking dimensions in Equation 6.12:

$$\left[L^2\right] = -2e^{[0]}\left(\int_0^x [L]e^{[0]}[L]\right) + \left[L^2\right]$$

Thus, with respect to dimensions, everything is OK]

Consider integral in Equations 6.12:

$$\textbf{(6.13)}\quad \int_0^x \frac{Nx+Q_0}{K}e^{\frac{2m}{\bar{b}}x}dx = \frac{N}{K}\int_0^x xe^{\frac{2m}{\bar{b}}x}dx + \frac{Q_0}{K}\int_0^x e^{\frac{2m}{\bar{b}}x}dx$$

Integration of the first term in Equation 6.16 yields

$$\textbf{(6.14a)}\ \int_0^x xe^{\frac{2m}{\bar{b}}x}dx = \frac{\bar{b}^2e^{\frac{2m}{\bar{b}}x}}{4m^2}\left(\frac{2m}{\bar{b}}x-1\right)\Bigg|_0^x = \frac{\bar{b}^2e^{\frac{2m}{\bar{b}}x}}{4m^2}\left(\frac{2m}{\bar{b}}x-1\right) + \frac{\bar{b}^2}{4m^2}$$

Integration of the second right hand term in term in Equation 6.13 yields

$$\textbf{(6.14b)}\ \int_0^x e^{\frac{2m}{\bar{b}}x}dx = \frac{\bar{b}}{2m}e^{\frac{2m}{\bar{b}}x}\Bigg|_0^x = \frac{\bar{b}}{2m}\left(e^{\frac{2m}{\bar{b}}x}-1\right)$$

Substituting these results in Equation 6.12

$$\textbf{(6.15)}\quad b^2 \approx -\left(\frac{N}{K}\bar{b}^2\frac{\frac{2m}{\bar{b}}x-1+e^{-\frac{2m}{\bar{b}}x}}{2m^2} + \frac{Q_0}{K}\bar{b}\frac{1-e^{-\frac{2m}{\bar{b}}x}}{m}\right) + b_0^2$$

To obtain the final solution to the thickness b, we have to find the unknown recharge $Q(0) = Q_0$. To this end, let us define the average thickness of the aquifer as

$$\textbf{(6.16)}\quad \bar{b} = \frac{h_0+h_L}{2} - m\frac{L}{2}$$

Then

$$(6{,}17)\quad Q_0 = K\frac{h_0 - h_L}{L}\bar{b} - \frac{NL}{2}$$

Thus finally, the solution to Equation 9 is

$$(6.18)\quad b^2 \approx \frac{N}{K}\bar{b}^2\frac{1 - \frac{2m}{\bar{b}}x - e^{-\frac{2m}{\bar{b}}x}}{2m^2} + \bar{b}\left(\frac{h_0 - h_L}{L}\bar{b} - \frac{N}{K}\frac{L}{2}\right)\frac{e^{-\frac{2m}{\bar{b}}x} - 1}{m} + b_0^2$$

Equation 6.18 solves the Case 1.1 problem.

The solution presented by Equation 6.18 must converge to the solution of the problem of the filtration in the aquifer with a horizontal base, that is, with $m = 0$. However straight forward substitution into Equation 6.18 leads to uncertainties $0/0$ in the first and second terms of Equation 6.18. Applying the L'Hopital's rule to the fractions containing m, we obtain

$$(6.19a)\quad \lim_{m\to 0}\frac{1 - \frac{2m}{\bar{b}}x - e^{-\frac{2m}{\bar{b}}x}}{2m^2} = \lim_{m\to 0}\frac{-\frac{2}{\bar{b}}x + \frac{2}{\bar{b}}xe^{-\frac{2m}{\bar{b}}x}}{4m} = \lim_{m\to 0}\frac{-\frac{2}{\bar{b}^2}x^2e^{-\frac{2m}{\bar{b}}x}}{2} = -\frac{1}{\bar{b}^2}x^2$$

$$(6.19b)\quad \lim_{m\to 0}\frac{e^{-\frac{2m}{\bar{b}}x} - 1}{m} = \lim_{m\to 0}\frac{-\frac{2}{\bar{b}}xe^{-\frac{2m}{\bar{b}}x}}{1} = \lim_{m\to 0}\left(-\frac{2}{\bar{b}}xe^{-\frac{2m}{\bar{b}}x}\right) = -\frac{2}{\bar{b}}x$$

Substituting the results presented by Equations 6.19 into Equation 6.18, we obtain

$$(6.20)\quad b^2 \approx -\frac{N}{K}x^2 - 2\left(\frac{h_0 - h_L}{L}\bar{b} - \frac{N}{K}\frac{L}{2}\right)x + b_0^2$$

Let we put $y(x) = 0$, meaning that $h(x) = h = b$. Then Equation 6.20 becomes

$$(6.21)\quad h^2 = h_0^2 - \frac{h_0^2 - h_L^2}{L}x + \frac{N}{K}(L - x)x$$

This result is the exact analytical solution for the steady-state one-dimensional flow in the unconfined aquifer on the horizontal base (Bear, 1972, p.380).

Returning to the water table elevation with a sloppy base of the aquifer in Equation 6.18, we finally have

$$(6.22)\quad h(x) \approx \sqrt{b_0^2 + \frac{N}{K}\bar{b}^2\frac{1 - \frac{2m}{\bar{b}}x - e^{-\frac{2m}{\bar{b}}x}}{2m^2} + \bar{b}\left(\frac{h_0 - h_L}{L}\bar{b} - \frac{N}{K}\frac{L}{2}\right)\frac{e^{-\frac{2m}{\bar{b}}x} - 1}{m}} + y(x)$$

6.3. *Case 1.2: One Head and One Flux Boundary*

The boundary conditions are assigned as the water table elevation and the flux at the left end of interval [***0, L***]:

(B.C. 1.2) $\quad h\ (0) = h_0 \quad$ and $\quad Q\ (0) = Q_0$

In this case we can use as the solution presented by of Equation 6.15 rewriting it as

(6.23)
$$h(x) \approx \sqrt{b_0^2 - \left(\frac{N}{K}\bar{b}^2 \frac{\frac{2m}{\bar{b}}x - 1 + e^{-\frac{2m}{\bar{b}}x}}{2m^2} + \frac{Q_0}{K}\bar{b}\frac{1 - e^{-\frac{2m}{\bar{b}}x}}{m} \right)} + y(x)$$

7. Heterogeneous Aquifer on an Uneven Base: Numerical Solution

Analytical solutions provide a rapid means for evaluating solutions for ground water flow. Analytical solutions are limited in application to homogeneous aquifers. The segmented aquifer solution, originally developed by Weaver (2004) allows piecewise variation in parameters and overcomes some of the limitation of the analytical solution. Weaver's solution, however, does not allow for an aquifer with a varying base, a situation commonly encountered. In principle, however, analytical solutions for sloping bases could be pieced together. In practice, however, these solutions require linearization and the solution of a non-linear system of algebraic equations. These features negate the benefits of the analytical solutions. In this section a numerical solution to the one-dimensional ground water flow equation is presented. This solution allows for varying aquifer properties (hydraulic conductivity and recharge) as well as a variable aquifer base.

The algorithm and the program are universal in the sense that they work for the steady-state one-dimensional flow in heterogeneous or homogeneous unconfined aquifer on a horizontal base as well, that is for all cases described above. The MatLab program 'D1_Flow' implementing the algorithm is presented in Appendix 4.

7.1. *Homogeneous Aquifer*

Let the aquifer be homogeneous. That is, its hydraulic conductivity, K, and the recharge, N, are constant. Its base is described by function $Y = Y\ (x)$ (Figure 7.1). The following differential equation, based on the Dupuit-Forchheimer assumptions describes steady-state, one-dimensional, ground-water flow in this case:

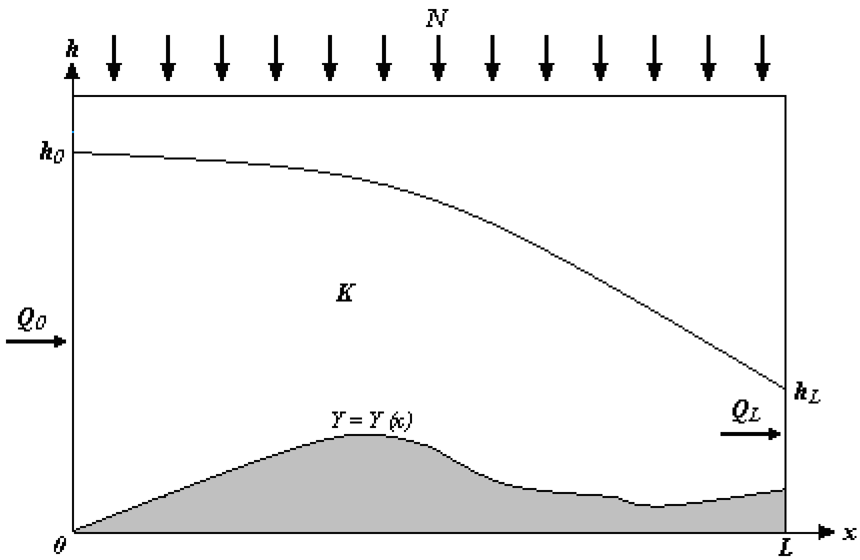


Figure 7.1: Conceptual rendering of flow system for model development.

(7.1) $$\frac{d}{dx}\left(K(h(x)-Y(x))\frac{dh(x)}{dx}\right)=-N$$

Integrating Equation 1 yields

(7.2) $$K(h(x)-Y(x))\frac{dh(x)}{dx}=-Nx+C$$

where ***C*** is an arbitrary constant.

Let us consider different boundary conditions for solving Equation 1.
Case 1: The boundary conditions are assigned as the water table elevation and the flux at the origin:

$\boldsymbol{h(0) = h_0}$ and $\boldsymbol{Q(0) = Q_0}$

Substituting $\boldsymbol{x=0}$ into Equation 2 gives, according to Darcy's law,

$\boldsymbol{C = -Q_0}$

Thus, Equation 7.2 becomes

(7.3) $$\frac{dh}{dx}=\frac{-Nx-Q_0}{K(h(x)-Y(x))}$$

Equation 7.3 can be approximated with a finite difference scheme and solved by a proper method under the boundary condition $\boldsymbol{h(0) = h_0}$.

Case 2: The boundary conditions are assigned at the right end of the object as

$\boldsymbol{h\ (L) = h_L}$ and $\boldsymbol{Q\ (L) = Q_L}$

In this case, $\boldsymbol{C}$ in Equation 2 becomes

$\boldsymbol{C = -(Q_L - NL) = -Q_0}$

Thus Equation 2 becomes

$$\textbf{(7.4)} \qquad \frac{dh}{dx} = \frac{(L-x)N - Q_L}{K(h(x) - Y(x))}$$

Equation 7.4 can be approximated with a finite difference scheme and solved by proper method under the boundary condition $\boldsymbol{h\ (L) = h_L}$.

Case 3: The above boundary conditions are assigned as

$\boldsymbol{h\ (x_1) = h_1}$ and $\boldsymbol{Q\ (x_2) = Q_2}$

Locations $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are within the object, that is, [$\boldsymbol{0 \le x_1 \le L}$] and [$\boldsymbol{0 \le x_2 \le L}$]. $\boldsymbol{x_1}$ may equal to or differ from $\boldsymbol{x_2}$. This case is a combination of Cases 1 and 2. For example, if $x_2 > x_1$ and $\boldsymbol{Q\ (x_2)}$ is known, we can calculate flux $\boldsymbol{Q\ (x_1)}$ and then solve an appropriate finite difference problem backward to the origin and forward to the right end of the object.

Case 4: The boundary conditions are given as the water table elevations in two locations:

$\boldsymbol{h\ (x_1) = h_1}$ and $\boldsymbol{h\ (x_2) = h_2}$

To reduce this problem to the previous ones, we need to find constant $\boldsymbol{C}$ in Equation 2. This goal can be achieved by employing the shooting method(Mathews and Fink, 1999; Boyce and DiPrima, 2000). First, we find interval [$\boldsymbol{Q_{min}, Q_{max}}$] such that it contains the unknown value $\boldsymbol{C}$. Then, **s**tarting with $\boldsymbol{Q = (Q_{min} + Q_{max})/2,}$ we solve Equation 2 numerically with the boundary conditions $\boldsymbol{h\ (x_2) = h_2}$ and $\boldsymbol{Q(x_1) = Q.}$ This $\boldsymbol{Q}$ becomes one of the boundaries of a new interval [$\boldsymbol{Q_{min}, Q_{max}}$]. The iterative process narrowing interval [$\boldsymbol{Q_{min}, Q_{max}}$] is continued until value $\boldsymbol{h_1}$ is reproduced with required accuracy. The value of $\boldsymbol{Q}$ obtained in this way than is used with one of values $\boldsymbol{h_1}$ or $\boldsymbol{h_2}$ in the appropriate Case 1 to 3.

7.2. Stream Functions

In the case of one–dimensional horizontal ground water flow in a vertically homogeneous aquifer, according to the Dupuit-Forchheimer assumptions the specific recharge is constant at any vertical section of the aquifer.

Let a stream function qF originate at location x_{qF} on the water table of the aquifer (Figure 7.2). Since according to the Dupuit assumption the specific flux does not depend on depth, we can write for any location x:

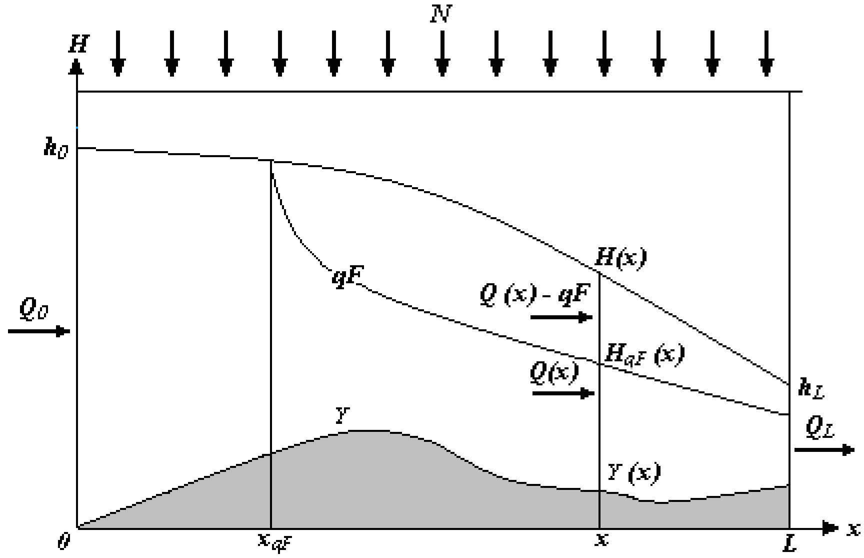


Figure 7.2 Definition of streamlines for calculation of plume diving.

(7.5) $$\frac{Q(x)-qF}{H(x)-H_{qF}(x)}=\frac{Q(x)}{H(x)-Y(x)}$$

where $H(x)$, $H_{qF}(x)$, $Y(x)$, are the elevations of the water table, the stream function qF, and the base at location x, respectively, and $Q(x)$ is the flux at the same location.
It follow from Equation 7.5 that the elevation (trajectory) of the stream function qF is

(7.6) $$H_{qF}(x)=H(x)-\left(1-\frac{qF}{Q(x)}\right)\big(H(x)-Y(x)\big)$$

As follows from the Dupuit-Forchheimer assumptions, the incoming recharge pushes down the already existing stream functions Strack (1989). It is interesting to note that in this model, the stream function $qF = 0$ is vertical at the location of a ground water divide. It extends to the base of the aquifer.

7.3. Plume Diving Trajectory

Plume delineation is one of the most important tasks of the early stages of site investigation. According to Weaver and Wilson (2000), in case of the advective contaminant transport, the contaminant plume created by the landfill located in interval [x_{lft}, x_{rght}] is limited by the stream tube defined by the stream functions corresponding x_{lft} and x_{rght}, that is, corresponding to $qF\ x_{rght}$ as the upper boundary in case of the flux in the positive direction and the lower boundary in case of negative flux, as it relates to Figure 7.2, and qFx_{lft} giving the second boundary. Thus, the space restricted by the trajectories of the stream functions qFx_{lft} and $qF\ x_{rght}$ define boundaries for the plume.

Since contaminant plumes develop in time, thee may not reach the entire extent defined by the object boundaries. Also transverse vertical dispersion may spread the contaminants beyond the delineated boundaries.

7.4. Travel Time

The following algorithm permits evaluating the travel time for the contaminant to reach any location. The algorithm is based on the Dupuit-Forchheimer assumptions about the underground flow which is steady-state and that the contaminant transport is only advective, meaning the absence of sorption and dispersivity. The absence of sorption excludes the retardation factor and decreases the real travel time.
The time for a contaminant particle to travel from location x_{st} to x along the streamline coinciding with stream function $qF(x_{st})$ is

$$\textbf{(7.7)} \quad t(x) = \int_{s_{st}}^{s_x} \frac{Rds}{v(s)} = Rn_{eff} \int_{x_{st}}^{x} \frac{\sqrt{1+\left(\frac{d\,_qH(z)}{dz}\right)^2}}{q(z)} dz = $$

$$= \frac{Rn_{eff}}{qF(x_{st})} \int_{x_{st}}^{x} \left(_qH(z)) - Y(z)\right)\sqrt{1+\left(\frac{d\,_qH(z)}{dz}\right)^2} dz$$

where s denotes the point on the trajectory traveled by the particle, $v(s)$ is the velocity of the participle at point s, n_{eff} and R are the effective porosity and retardation factor, $q(z)$, $H(z,)$, $_qF(x_{st})$, $_qH(z)$, and $Y(z)$, are the Darcy velocity, water table elevation, stream function starting at x_{st}, stream function elevation , and aquifer base elevation at location z representing a current coordinate x in integration.

The following finite difference procedure approximates Equation 7.7in the case of the positive direction of the flow moving the contaminant in direction of increasing x:

$$\textbf{(7.8)} \quad t_{j+1} \approx t_j + Rn_{eff} \frac{_qH_j + {_qH_{j+1}} - Y_i - Y_{i+1}}{2qF_{x_{st}}} \sqrt{1+\left(\frac{_qH_{j+1} - {_qH_j}}{\Delta x}\right)^2} \Delta x$$

In Equation 8, the initial i is defined by the index i_{st} corresponding to x_{st} ($x_i = x_{st}$) and

$j = i - i_{st} + 1$

For contaminant movement in the negative direction (decreasing x) or in both negative and positive directions (from a water divide) requires slight changes in the procedure described by Equation 8. If a water divide is included in the simulation, we cannot calculate the travel times along the no flow stream function. However we can come as close to this flow line as we wish for an approximate calculation.

7.5. Horizontally Heterogeneous Aquifer with an Uneven Base

For the heterogeneous aquifer Equation 7.1 1 - 2 take form

(7.9) $$\frac{d}{dx}\left(K(x)(h(x)-Y(x))\frac{dh(x)}{dx}\right)=-N(x)$$

(6,\.10) $$K(x)(h(x)-Y(x))\frac{dh(x)}{dx}=-\int_0^x N(z)dz-Q(0)$$

Let us consider a piecewise heterogeneous aquifer consisting of ***n*** homogeneous segments with boundaries at locations $L_0, L_1, \ldots, L_{j-1}, L_j, L_{j+1}, \ldots, \ldots, L_n$ (Figure 3). That is, the hydraulic conductivity K_j and recharge N_j are constant within any interval [L_j, L_{j+1}]. (Note that in the case of the travel time calculations, the piecewise homogeneity assumes that the retardation factor and the effective porosity are constant within any interval [L_j, L_{j+1}].)

Within a homogeneous interval [L_j, L_{j+1}], Equation 1 takes form

(7.11) $$\frac{d\left(K_j(h(x)-Y(x))\frac{dh(x)}{dx}\right)}{dx}=-N_j$$

The first integral of Equation 11 is

(7.12) $$K_j(h(x)-Y(x))\frac{dh(x)}{dx}=-N_j(x-L_j)+C_j$$

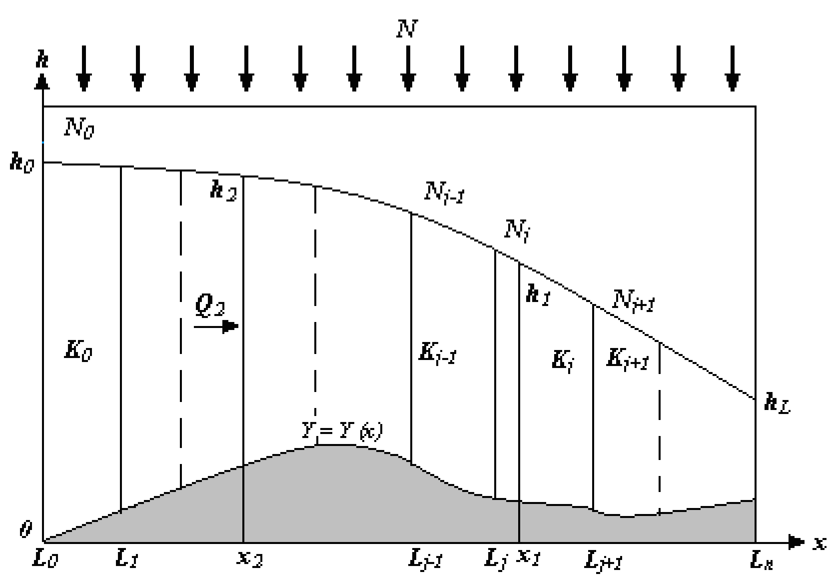


Figure 7.3 Piecewise heterogeneous aquifer.

It follows from Equation 7.12, according to the Darcy's law that $C_j = -Q_j$, that is, to the influx through the left boundary of interval $[L_j, L_{j+1}]$. And finally, we have the following equation analogous to Equation 7.3:

$$\textbf{(7.13)} \quad \frac{dh}{dx} = \frac{-N_j(x - L_j) - Q_j}{K_j(h(x) - Y(x))}$$

Thus the filtration within each homogeneous segment is described by a specific Equation 7.13. Those equations are connected with the inner boundary conditions on the continuity of the water table elevations and fluxes:

$$\textbf{(7.14)} \quad h(x_-) = h(x_+) \text{ and } K(x_-)(h(x_-) - Y(x_-))\frac{dh(x_-)}{dx} = K(x_+)(h(x_+) - Y(x_+))\frac{dh(x_+)}{dx}$$

Conditions 7.14 are satisfied automatically in the final difference procedures used in D1_Flow program.
There may be two kinds of boundary conditions for solving Equations 11. First, the boundary conditions are assigned as the water table elevation at location x_1 and the flux is assigned at location x_2, that is,

$h(x_1) = h_1$ and $Q(x_2) = Q_2$

The boundary conditions are assigned also as the water table elevations at two locations:

$h(x_1) = h_1$ and $h(x_2) = h_2$

If both x_1 and x_2 belong to the same interval, than we work within a homogeneous segment and can obtain water table elevations within it as described in Section 2.1. Obtaining the water table elevations and fluxes at the end of the segment, we can proceed moving in both directions, that is, to $x = x_0$, the left edge of the object, and to $x = x_L$, the right end of the object.

The procedure is analogous to those described in Section 7.1 can be applied, if x_1 and x_2 belong to different segments. In this case the procedures take in consideration the heterogeneity of the object. All other procedures require only minimal changes. Thus the travel time procedure, described by Equation 8 must be rewritten as

$$t_{j+1} \approx t_j + R_j n_{eff\,j} \frac{{}_qH_j + {}_qH_{j+1} - Y_i - Y_{i+1}}{2qF_{x_{st}}} \sqrt{1 + \left(\frac{{}_qH_{j+1} - {}_qH_j}{\Delta x}\right)^2} \Delta x$$

7.6. Program D1_Flow

Program D1_ Flow implementing all above algorithms is written in Matlab (Mathews and Fink, 1999). The program permits practically an unlimited number of piecewise segments for assignment of the recharge, hydraulic conductivity, retardation factor and effective porosity if calculating the travel time is desired. The base of the aquifer is assigned as a set of points (defined by x coordinate and elevation) with linear interpolation between them. Any piecewise representation or smooth functional representation of the aquifer base is acceptable. Defining the elevations of the aquifer base at the endpoints of the object is mandatory.

To solve the finite differences equations, the forward and backward second and the fourth orders Runge-Kutta and the backward Euler methods are used. By default, increment Δx is equal to 0.01, meaning that if the object dimension is given in meters, all calculations are being done with increment 1 cm, if the object dimension is given in kilometers the calculations are being done with increment 10 m, and so on.

Other default characteristics are the retardation factor equal to ***1***, and the effective porosity equal to ***0.4***. All default characteristics can be easily changed if necessary. The outer boundary conditions: the water table elevations at two different locations: $h_1 = h(x_1)$ and $h_2 = h(x_2)$ or the water table elevation $h_1 = h(x_1)$ and the flux $Q_2 = Q(x_2)$ ($x_1 = x_2$ is permitted in this case) can be assigned anywhere within the object. The program consists of several subroutines. All characteristics of the object are to be input in subroutine “object”. Calculating of the water table elevations and the fluxes at all locations of the object go automatically. All other calculations, the plume diving, stream function and travel time can be done by request of a user.

The program executes rapidly. However it works faster when boundary conditions are assigned as water table elevation and flux at the same or different location. In this case there are no iterations that are necessary in the case assigned as water table elevations at two locations.

The program is stable. However, it may happen that the assigned boundary conditions and the aquifer characteristics are contradictory. The contradiction can lead to yielding the water table elevations that exceed the object surface or that are below the base of the aquifer. In such cases the program continues calculating the water table elevations, plots them, and prompts the consumer about the occurred contradiction.

7.7. Verification of the Code D1_Flow

Verification of program D1_Flow is a complicated problem since there are very few analytical solutions to the one-dimensional flow in unconfined aquifers with not horizontal base. All existing analytical solutions assume the base being a sloped plane and apply other simplifying assumption such as homogeneity of the aquifer, absence of recharge, and some others. Some of them are complicated and require considerable efforts to realize them (Polubarinova-Kochina, 1962; Bear 1972; Hantush and Marino, 2001). For these reasons, we limited ourselves with verifying program D1_Flow using cases of homogeneous and heterogeneous aquifers on horizontal bases. This permits us also verifying plume diving and streamlines based on the solutions presented by Strack, 1989, and Weaver and Wilson, 2000.

Two examples were compared against the segmented, horizontal-base, solution developed by Weaver (2004)[2]. In the first problem there was a drop in head of 10 meters over a 1070 meter distance. The aquifer, in this case was assumed to be homogeneous with hydraulic conductivity of 60.96 m/d and recharge rate of 558.8 mm/y (Table 7.1). The maximum error between the analytical and numerical models was 8.6×10^{-5} for the water table and 9.2×10^{-5} for the upper stream line bounding the diving plume. Figures 7.4 and 7.5 show the errors in water table and bounding streamline, respectively.

The second case allowed for variation in recharge rate in one of seven aquifer segments and hydraulic conductivity in three of the seven. The maximum error between the analytical and numerical models was 1.8×10^{-4} for the water table and 8.1×10^{-4} for the upper stream line bounding the diving plume. Figure 7.6 shows the errors in water table and bounding streamline.

Table 7.1 Parameters for comparison of D1_Flow against an analytical solution.

Segment	Length (m)	Boundary Head (m)	Case 1		Case 2	
			K (m/d)	N (mm/y)	K (m/d)	N (mm/y)
1	152	16.76	60.96	558.8	60.96	558.8
2	152	n/a	60.96	558.8	60.96	558.8
3	152	n/a	60.96	558.8	60.96	1117.6
4	152	n/a	60.96	558.8	60.96	558.8
5	152	n/a	60.96	558.8	6.1	558.8
6	152	n/a	60.96	558.8	6.1	558.8
7	152	6.76	60.96	558.8	6.1	558.8

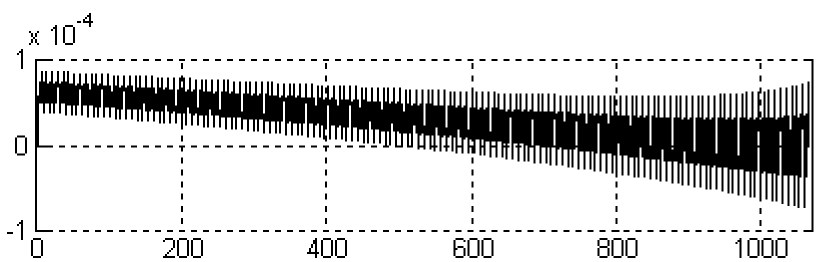


Figure 7.4: Errors in the water table elevations simulated by program D1_Flow against the results obtained analytically

[2] See http://www.epa.gov/athens/onsite

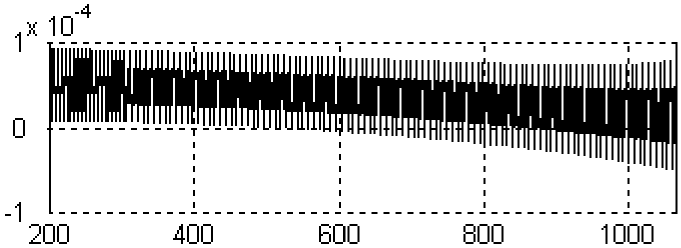


Figure 7.5: Errors in the plume top elevations simulated by program D1_Flow against the results obtained analytically

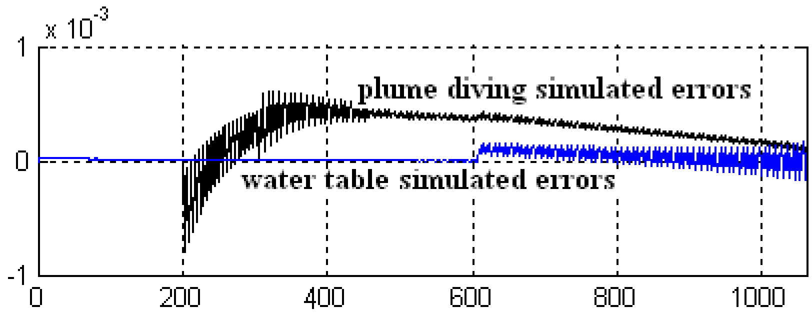


Figure 7.6: Errors of the results simulated by program D1_Flow with respect to the result obtained analytically.

The solution of a problem with a sloping aquifer base was compared against a solution of Polubarinova –Kochina (1962, p.415) which does not include recharge. The solution, rewritten in notation of this text, is

(7.15) $$x = Ae^{\frac{Km}{Q}h} + \frac{h}{m} + \frac{Q}{Km^2}$$

where $\boldsymbol{h}$ is the water table elevation, $\boldsymbol{x(h)}$ is the distance from the origin, $\boldsymbol{m}$ is the slope of the aquitard, $\boldsymbol{K}$ is the hydraulic conductivity, $\boldsymbol{Q}$ is the flux, and $\boldsymbol{A}$ is the arbitrary constant ($\boldsymbol{m}$, $\boldsymbol{K}$, and $\boldsymbol{Q}$ are constants). For the boundary condition assigned as $\boldsymbol{x(h_0) = 0}$ where is the water table elevation on the left end of the interval of interest [$\boldsymbol{0, L}$], Equation 7.15 takes form

(7.16) $$x = \frac{1}{m}\left(h + \frac{Q}{mK} - \left(h_0 + \frac{Q}{mK}\right)e^{\frac{mK}{Q}(h-h_0)}\right)$$

Table 7.2 Summary of cases for comparison of D1_Flow and Polubarinova–Kochina sloping aquifer solution.

Case	Boundary Conditions		Aquifer Base Elevation (m)		Infinite error norm (m)
	Upgradient	**Downgradient**	**b(0)**	**b(L)**	
1	h(0) = 31.27 m	Q(0) = 2.68976 m^2/day	0.00	22.34	9.05×10^{-5}
2	h(0) = 31.27 m	Q(0) = 4.05062 m^2/day	0.00	11.17	2.24×10^{-5}
3	h(0) = 31.27 m	h(L) = 28.27 m	0.00	11.17	6.05×10^{-5}
4	h(0) = 31.27 m	h(L) = 28.27 m	0.00	22.34	9.97×10^{-5}

Four cases were used to compare D1_Flow and the sloping aquifer solution of Polubarinova-Kochina (Table 7.2). In each case the solutions were essentially indistinguishable from each other and the maximum errors were less than 1 x 10^{-4} m. The effect of increasing the base slope was to increase the maximum error for both of the two sets of boundary conditions.

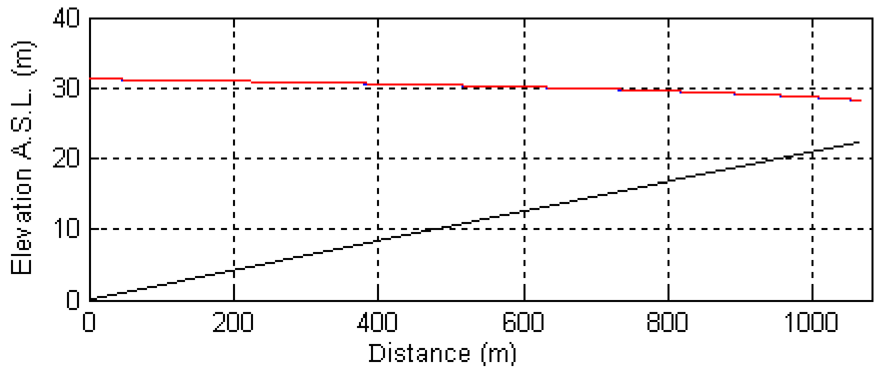


Figure 7.7: Graphical comparison of the results simulated by program D1_Flow and the analytical solution of Polubarinova -Kochina (They are not distinguishable at this scale.)

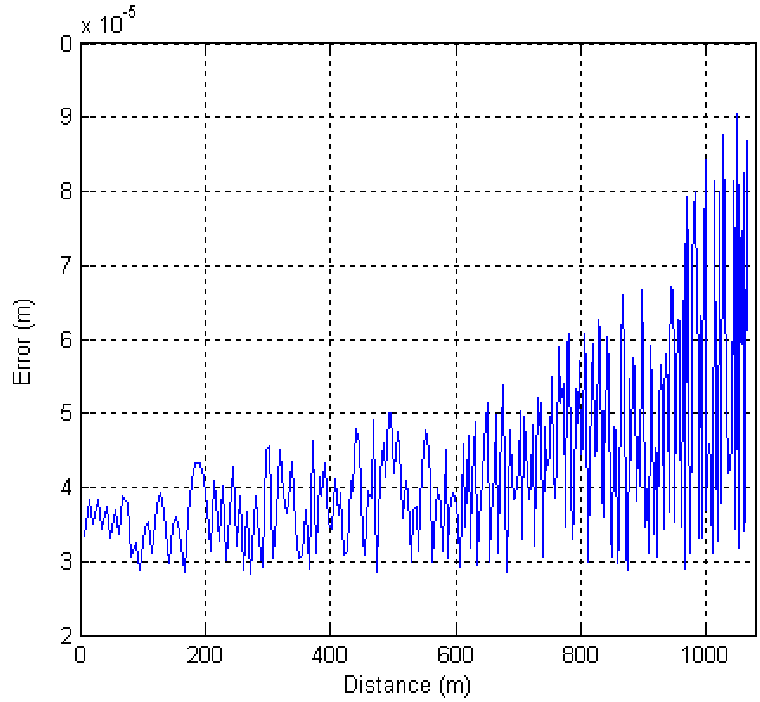


Figure 7.8 Case 1: Errors of the results simulated by program D1_Flow against the analytical solution of Polubarinova -Kochina

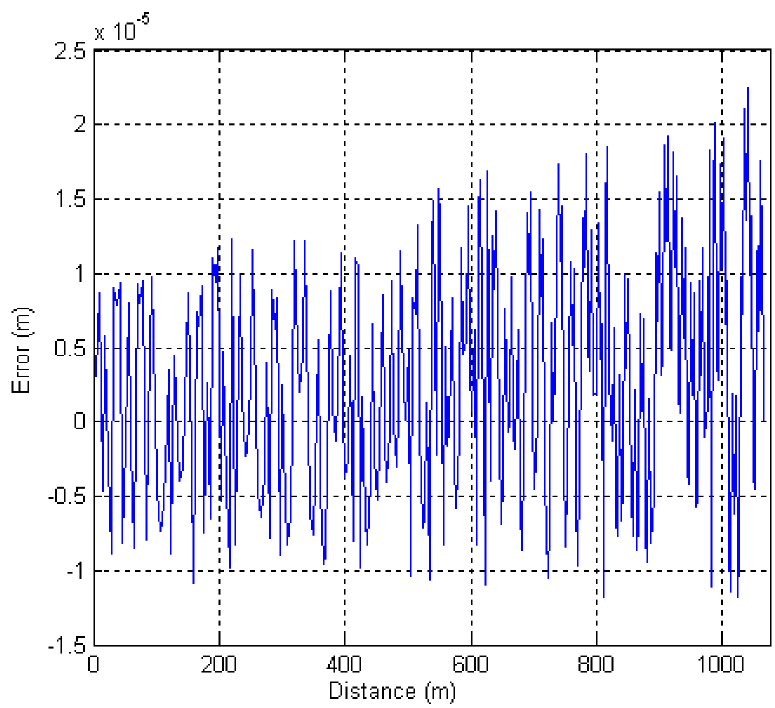


Figure 7.9 Case 2: Errors of the results simulated by program D1_Flow against the analytical solution of Polubarinova -Kochina

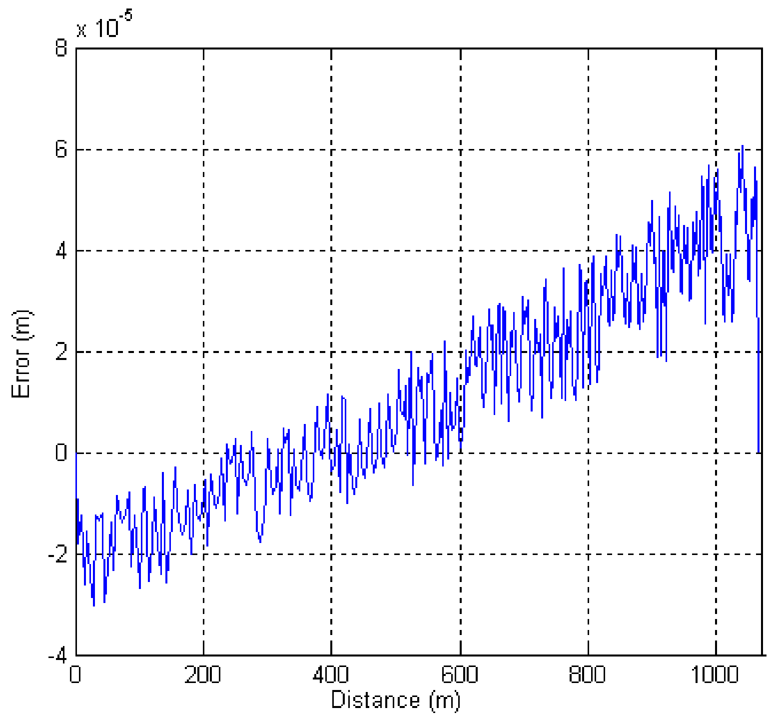


Figure 7.10 Case 3: Errors of the results simulated by program D1_Flow against the analytical solution of Polubarinova –Kochina, Systematic nature of the error is caused small difference between Q0 and q that are found by iteration.

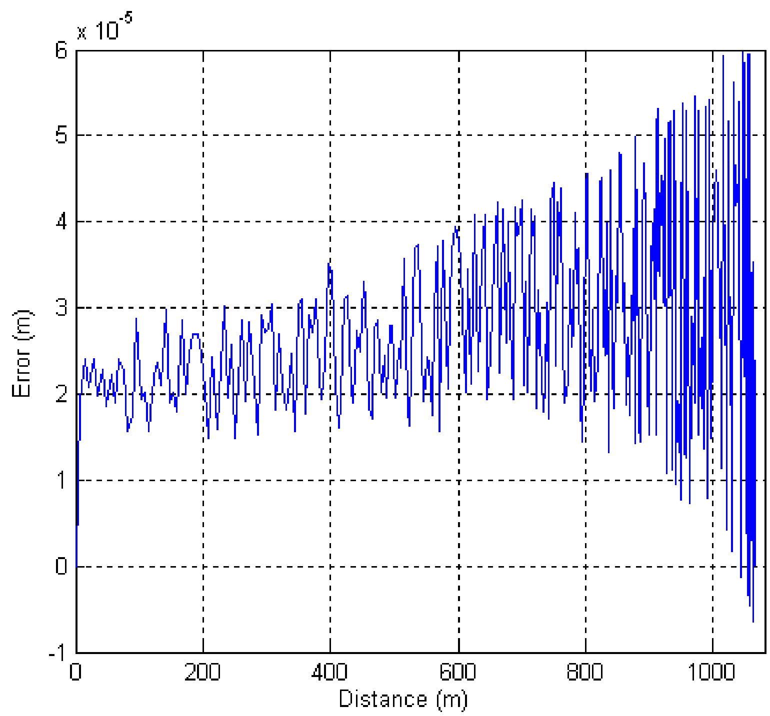


Figure 7.11: Case 4.:Errors of the results simulated by program D1_Flow against the analytical solution of Polubarinova –Kochina. Systematic nature of the error is caused small difference between Q0 and q that are found by iteration.

7.8. Comparison of One- and Two-Dimensional Modeling Results for the Borden Landfill

7.8.1. Borden Landfill Site

Data and simulation results from the Borden Landfill were used for testing the model. The Borden Landfill plume was the object of three-dimensional characterization reported by MacFarlane et al. (1983). The reported investigation at the site lasted from 1974 to 1980. The Borden Landfill is situated between two northward-flowing streams that feed the Georgian Bay of Lake Superior. Glacial deposits form the dominant surface material in this region (Eyles et al., 1992) The unconfined surfacial aquifer consists of beds and lenses of fine-, medium- and coarse-grained sand overlying an extensive deposit of clayey and sandy silt.

Following the presentation by MacFarlane et al. (1983) longitudinal cross sections are used to represent the contaminant plume. Chloride was chosen as a tracer to represent the contamination. MacFarlane et al. (1983) used the 10 ppm contour as the boundary between the plume and uncontaminated water. These show evidence of recharge-driven plume diving as the plume generally becomes deeper in the aquifer with distance from the landfill, and there is a localized deepening of the contaminant plume beneath the sand quarry (7.7). The plume exhibits other interactions with the hydrologic system as it drops to the base of the aquifer beneath the ephemeral ground water divide located near the southern edge of the landfill. The purpose of comparing results with the Borden Landfill was to determine the ability of the new plume diving model D1_Flow to reproduce these qualitative features and to assess its ability to reproduce them quantitatively, using a full calibration data set. Since the intended purpose of the model is to aid in site characterization, the model was also applied to subsets of the data. This application was intended to show how the model results can be incorporated into an iterative site characterization approach.

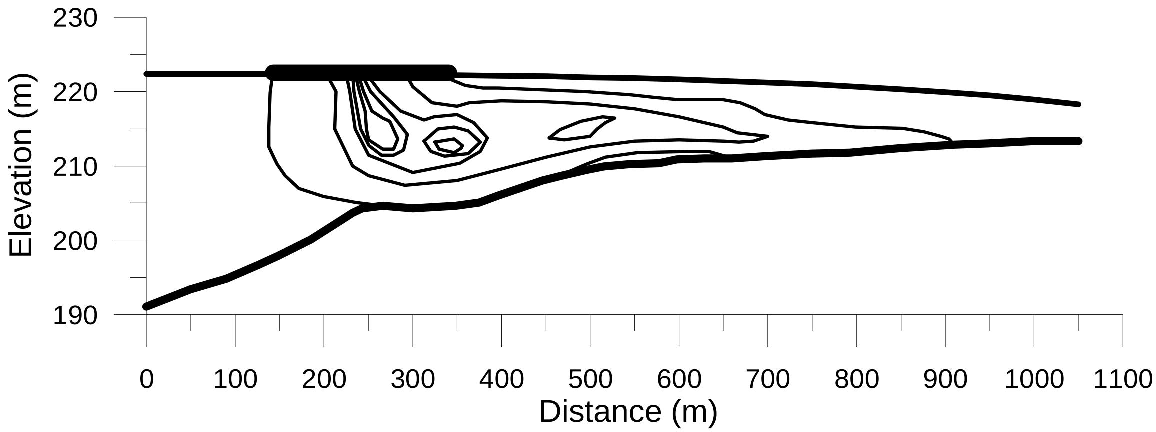


Figure 7.7 Cross section showing the distribution of chloride from the Borden Landfill (MacFarlane et al., 1983). The outermost contour represents 10 ppm chloride, while the inner contour interval is 100 ppm, starting at 100 ppm.

The quantitative task is split into two parts. First is the matching of model results to the calibrated model of Frind and Hokkanen (1987), based on the earlier model developed by Frind and Matanga (1985) to establish the capacity of this simplified model. The Frind and Matagna (1985) model is well suited for our comparison, because it is based on full simulation of the vertical and horizontal flows, whilst our model assumes, according to Dupuit, that the aquifer can be represented by horizontal flows only. The second test is to use the model in a simulated site assessment to illustrate its potential use in characterizing an aquifer of this type.

7.8.2. Reproducing Two-Dimensional Modeling Results

In this exercise, we reproduced the flow system presented graphically by Frind and Hokkanen (1987). The recharge pattern was simplified is shown in 7.3 as simulation 1. The hydraulic conductivity, 1.16×10^{-2}cm/s, was the same as used by Frind and Hokkanen (1987). The boundary conditions were assigned at $x = 0$ as the elevation of the water table, 222.36 m, and the flux of 70 cm/yr. The results gave a ground water divide at 136.83 m with elevation of 222.37 m. At the downgradient end of the flow domain there was a water table elevation of 218.49m and flux of 0.63 m^2/day.

Table 7.3 Simulation parameters for three test problems, including the discretization intervals, recharge rates and hydraulic conductivities.

Simulation	Interval (m)	0 – 140	140-300	300-600	600-800	800-1050
1	Recharge (cm/y)	15	55	15	45	15
	Hydraulic Conductivity (cm/s)	$1.16*10^{-2}$	$1.16*10^{-2}$	$1.16*10^{-2}$	$1.16*10^{-2}$	$1.16*10^{-2}$
2	Recharge (cm/y)	5	5	0	0	0
	Hydraulic Conductivity (cm/s)	$1.16*10^{-2}$	$1.16*10^{-2}$	$1.16*10^{-2}$	$1.16*10^{-2}$	$1.16*10^{-2}$
3	Recharge (cm/y)	15	55	15	45	15
	Hydraulic Conductivity (cm/s)	$1.16*10^{-2}$	$1.16*10^{-2}$	$0.812*10^{-2}$	$0.986*10^{-2}$	$2.32*10^{-2}$

A graphical comparison of results is presented in Figure 7.8. Close agreement of the modeled water table was obtained between the results of Frind and Hokkenan (1987) and the current model. The maximum deviation from the water table presented by Frind and Hokkanen (1987) is 0.10 m. It can be seen, however, that both model results are biased with respect to the actual observations, especially where they over predict the water table elevation over the region between 300 m to 900 m from the origin.

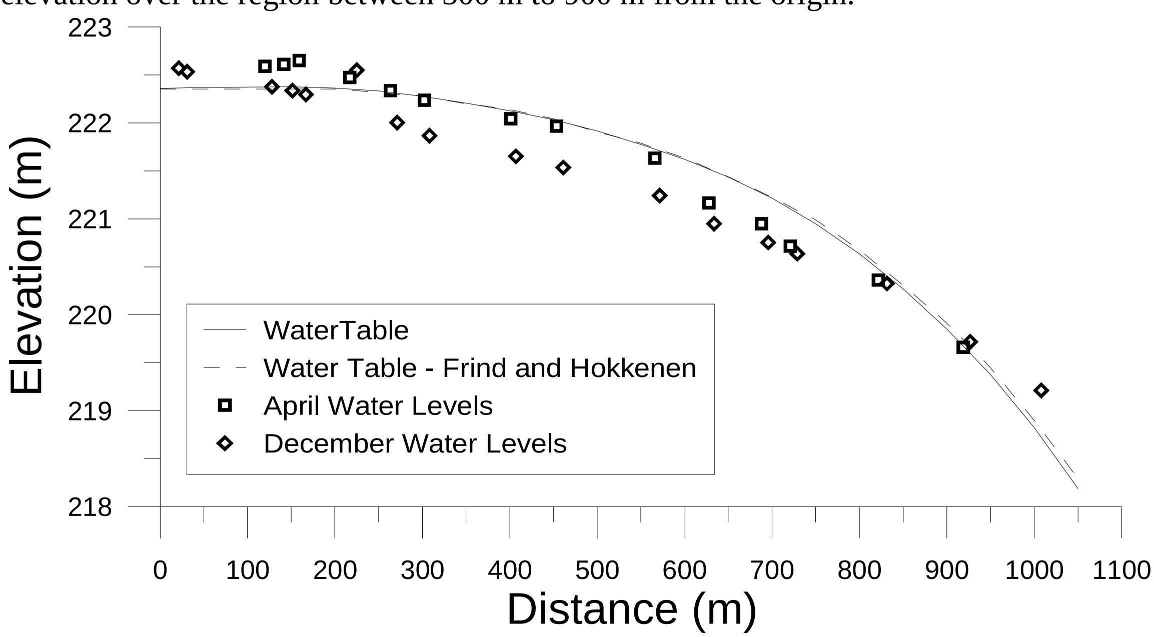


Figure 7.8 Comparison between measured water levels in two seasons (April and December) with simulations performed by Frind and Hokkenan (1987) and the current model.

These water level data were collected in April and in December. The April data show evidence of the ephemeral ground water mound which was said to dissipate in the late summer and fall. By the December sampling the mound was gone and water levels were generally lower throughout most of the domain. Because of the seasonally transient nature of the site, application of a steady state model is a compromise. To further evaluate the fit to the water table, we used a linear regression to estimate the least-squares representation of the water table. The result was

$$E = -2.576\ x\ 10^{-6}\ x^2 - 7.753\ x\ 10^{-4}\ x + 222.63$$

where E is the water table elevation (m) and x is the distance from the origin (m). Figure 7.9 shows two additional simulation results. First there is a very close agreement between the regression equation and the present model calibrated by changing the recharge pattern and keeping the conductivity fixed to the value used by Frind and Hokkanen (Table 7.3, simulation 2). Insofar as this recharge pattern is plausible, this model could represent the flow system. The recharge in this system, however, was low with 5 cm/y in the landfill and 0 cm/y downgradient. The second result uses the recharge pattern of Frind and Hokkenan and allows the hydraulic conductivity to vary. Here the conductivities varied by a factor of three (Table 7.3, simulation 3). The lowered water tables achieved in the first of these simulations may represent wintertime conditions with reduced recharge due to frozen conditions, while conductivities used in the second simulation are within the range measured for the site. Differing parameter sets are in part a consequence of differing purposes of simulation. Matching water tables, transport times, or streamline positions lead to different parameter sets (Gorokhovski 1977, 1996).

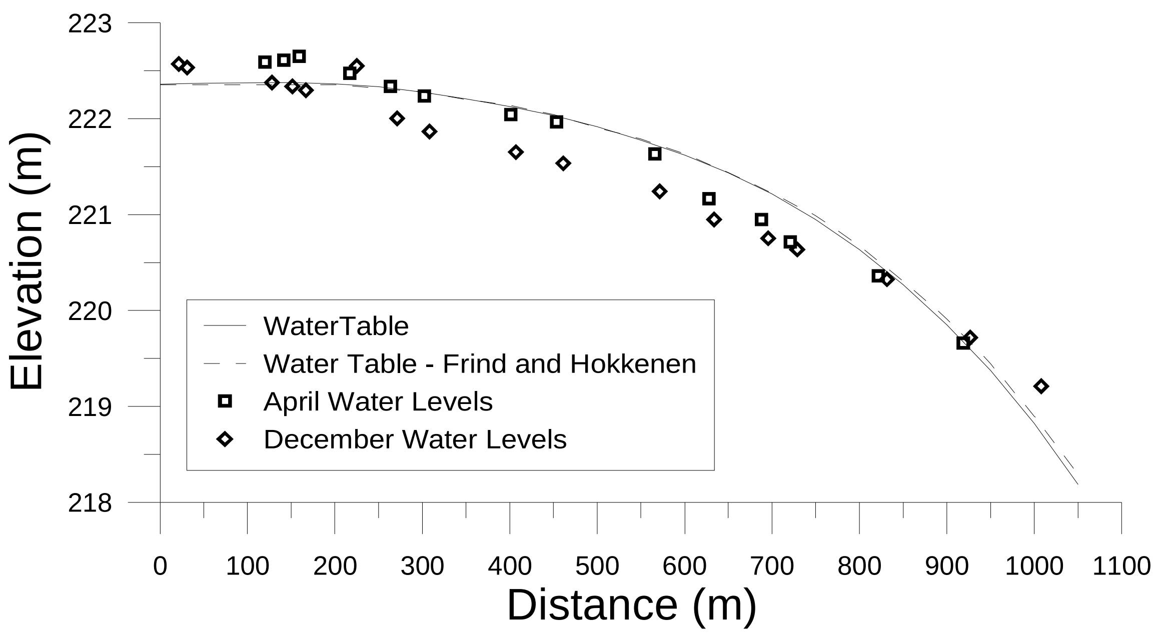


Figure 7.9 Comparison between measured water levels in two seasons (April and December) with simulations performed by Frind and Hokkenan (1987) and the current model.

Streamlines from simulation 1 and the actual boundary of the 10 ppm chloride contour are shown in Figure 7.10. These streamlines are similar to those produced by Frind and Hokkenan, differing mainly due to differences in recharge assumptions. We used an infiltration rate of 55 cm/y from the ground water flow divide at the southern edge of the land fill, throughout its 160 m length. Consequently 0.24 m^2/d of water flows between the divide and the streamline on the downgradient edge of the landfill. The ability of a

Dupuit-Forchheimer model to transfer recharge water to depth within an aquifer was explained by Kirkham's (1967) slot-slab interpretation. Water is conducted downward in slots that are adjacent to slabs of porous media. Thus it is expected that the D1_Flow model can represent downward flow at the ground water divide and the sand quarry. Thus the simplified model has the ability to reproduce qualitatively and quantitatively the plume behavior at this site.

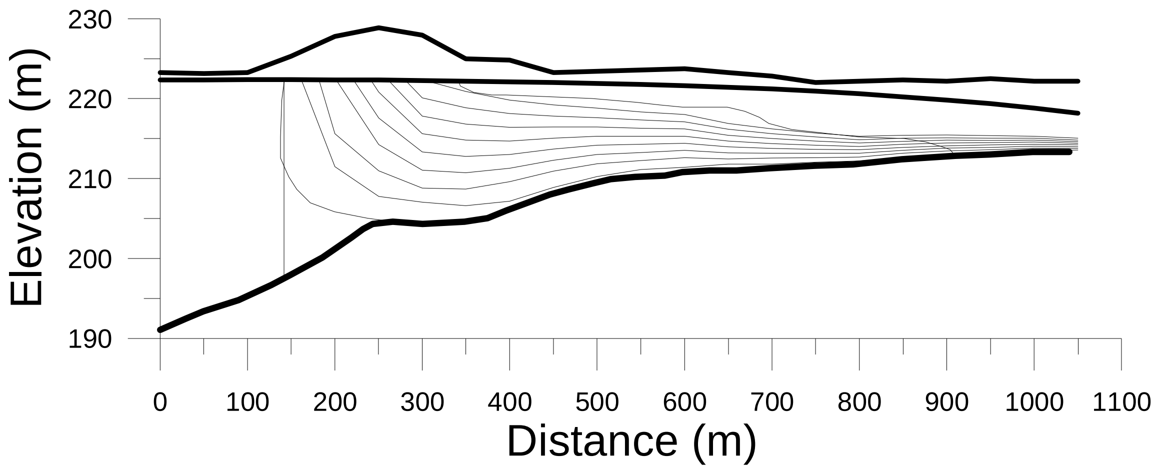


Figure 7.10 Streamlines originating at assumed boundaries of the land fill. 10ppm contours are plotted showing that the streamlines lie within the envelope created by these data.

8. Conclusions

A series of solutions for one-dimensional ground water flow have been developed. These begin with an analytical solution for a homogeneous aquifer with an horizontal base, and progress to a numerical solution for a piecewise heterogeneous aquifer with a sloping base, where the "pieces" can be arbitrarily small. The latter solution was shown to reproduce results from 1) an analytical solution for an aquifer with an horizontal base, 2) an analytical solution for a piecewise heterogeneous aquifer and 3) field data from the Borden Landfill. Both the water table and upper bound streamline were reproduced accurately numerically. These comparisons demonstrate that under suitable site conditions the numerical model can be used for plume diving calculations.

9. References

Bear J., Dynamics of Fluids in Porous Media, 1972.

Boyce W.E. and DiPrima R. C. (2001), Elementary Differential Equation, ed. 7,592 pp.

Brutsaert, W., 1994, The unit response of groundwater outflow from a hillslope, Water Resources Research, 30(10), 2759-2763.

Childs, E.C., 1971 Drainage of groundwater resting on a sloping bed, Water Resources Research, 7(5), 1256-1263.

Dernbach, L.S., 2000. "The Complicated Challenge of MTBE Cleanups," Environmental Science & Technology 34 no. 23: 516A-521A.

Eyles N., J.I. Boyce, and J.W. Hibbert, 1992, The geology of garbage in southern Ontario, Geoscience Canada, 19(2), 50-62.

Frind E.O. and Hokkanen G.E., Simulation of the Borden plume using the alternating direction Galerkin technique, 1987, WWR, 23, (5), 918-930.

Frind, E.O., G.B. Matanga, 1985, The dual formulation of flow for contaminant transport modeling: 1. Review of theory and accuracy aspects, Water Resources Research, 21(2), 159-169.

Fan, Y, and R. Bras, 1998, Analytical solutions to hillslope subsurface storm flow and saturation overland flow, Water Resources Research, 34(4), 931-927.

Gorokhovski V.M., 1977, Matematicheskie metody i dostovernost idrogeologicheskikh i ingenerno-geologicheskikh prognozov (Mathematical methods and reliability of hydrogeological and engineering geological predictions), Moscow, Nedra, 77 p., (in Russian)

Gorokhovski V.M., 1996, Problem-dependence of ground-water model identifications: Significance, extent, treatment, Ground Water, 34(3), pp.461-469

Gorokhovski V.M. and J.W. Weaver, 2007, Numerical Modeling of One-Dimensional Steady-State Flow and Contaminant Transport in a Horizontally Heterogeneous Unconfined Aquifer with an Uneven Base, submitted to Water Resources Research.

Hantush M.M. and Marino M.A, Analytical modeling of the influence of denitrifying sediments on nitrate transport in aquifer with sloping beds, WWR, 37(12), 3177-3192.

Hilberts, A.G.J., E.E. van Loon, P.A. Troch, and C. Paniconi, 2004, The hillslope-storage Boussinesq model for non-constant bedrock slope, Journal of Hydrology, 291, 160-173.

ITRC, 2003, Technical and Regulatory Guidance for the Triad Approach: A New Paradigm for Environmental Project Management, Interstate Technology and Regulatory Council, Sampling, Characterization and Monitoring Team, www.itrcweb.org.

Kirkham, D., 1967, Explanation of paradoxes in Dupuit-Forchheimer seepage theory, Water Resources Research, 2(3), 609-622.

Landmeyer J.E., F.H. Chapelle, P.M. Bradley, J.F. Panlow, C.D. Church, P.G. Tratnyek, 1998, Fate of MTBE relative to benzene in a gasoline-contaminated aquifer, Ground Water Monitoring and Remediation, 18(4), 93-102.

LeBlanc, D.R., S.P. Garabedian, K. M. Hess, L.W. Gelhar, R.D. Quadri, K.G. Stollenwerk, and W.W. Wood, Large-scale natural gradient tracer test in sand and gravel, Cap Cod, Massachusetts, 1, Experimental design and observed tracer movement., Water Resources Research, 27(5), 895-910.

MacFarlane D. S., Cherry J. A., Gillham R. W., and Sudicky, 1983, Migration of contaminants in groundwater at a landfill: A case study, 1.Groundwater plume delineation, J. Hydrol., 63(1/2), 1 -30

Mathews, J.H., and Fink K.D. (1999), Numerical Methods using Matlab

Nichols, E.M. and T. L. Roth, 2006, Downward Solute Plume Migration: Assessment, Significance, and Implications for Characterization and Monitoring of "Diving Plumes," American Petroleum Institute Soil and Groundwater Technical Task Force, Bulletin 24, April.

Polubarinova- Kochina P. Ya., Theory of Ground Water Movement, 1962, 633p.

Steward, D.R., 2007, Groundwater response to change water-use practices in sloping aquifers, Water Resources Research, 43, W05408, doi:10.1029/2005WR004837.

Strack O. D., Three-Dimensional Streamlines in Dupuit - Forchheimer Models, Water Resources Research, 1984, 20, (7), 812-822.

Strack O. D., Groundwater Mechanics, 1989, Prentice – Hall Inc., 732 pp.

Telyakovskiy, A.S. and M.B. Allen, 2006, Polynomial approximate solutions to the Boussinesq equation, Advances in Water Resources, 29, 1767-1779.

Troch, P., E. van Loon, A. Hilberts, 2002, Analytical solutions to a hillslope-storage kinematic wave equation for subsurface flow, Advances in Water Resources, 25, 637-649.

United States Environmental Protection Agency, 1997, Expedited Site Assessment Tools for Underground Storage Tank Sites: A Guide for Regulations, U.S. EPA, Office of Underground Storage Tanks, Washington, DC, 20460, EPA/510/B-97/001. http://www.epa.gov/oust/pubs/sam.htm.

United States Environmental Protection Agency, 2003, Using the Triad Approach to Streamline Brownfields Site Assessment and Cleanup – Brownfields Technology Primer Series, U.S. EPA, Office of Solid Waste and Emergency Response, Washington, DC, 20460, EPA 542-B-03-002.

United States Environmental Protection Agency, 2004, Improving Sampling, Analysis, and Data Management for Site Investigation and Cleanup, Office of Solid Waste and Emergency Response, Washington, DC., EPA-542-F-04-001a.

United States Environmental Protection Agency, 2005, Triad saves $109k on three petroleum sites, Cleanup News II, Office of Site Remediation, Washington, DC, Issue 21S2, September, page 5.

United States Environmental Protection Agency, 2006, Triad speeds cleanup of lead-contaminated firing range soil, Cleanup News, Office of Site Remediation Envorcement, Washington, DC, Issue 25, March, page 4.

Verhoest, N.E.C. and P. A. Troch, 2000, Some analytical solutions of the linearized Boussinesq equation with recharge for a sloping aquifer, Water Resources Research, 26(3), 793-800.

Weaver, J.W, J.T. Haas, and J.T. Wilson, 1996, Analysis of the gasoline spill at East Patchogue, New York, Proceedings of Non-Aqueous Phase Liquids (NAPLs) in Subsurface Environment: Assessment and Remediation, ed. L. Reddi, American Society of Civil Engineers, Washington, D.C., November 12-14, pp. 707-718.

Weaver, J.W. and J.T. Wilson, 2000, Diving Plumes, L.U.S.T.Line, New England Interstate Water Pollution Control Commission, 36(12-15).

Weaver, J.W., S.C. McCutcheon, S.L. Lewis, and N.L Wolfe, 2002, Hydrologic Feasibility Assessment and Design of Phytoremediation Systems that Rely on Evapotranspiration, Handbook of Phytoremediation.

Weaver, J. W., 2004, On-line Tools for Assessing Petroleum Releases, United States Environmental Protection Agency, Washington, D.C., EPA 600/R-04/101.

Wiedemeier, T.H., M.A. Swanson, D.E. Moutoux, E.K. Gordon, J.T. Wilson, B.H. Wilson, D. H. Kampbell, P.E. Haas, R. N. Miller, J.E. Hansen, F. H Chapelle, 1998, Technical Protocol for Evaluating Natural Attenuation of Chlorinated Solvents in Ground Water, United States Environmental Protection Agency, Office of Research and Development, Washington, DC, 20460, EPA/600/R-98/128.

Wilson, J.T., J.S. Cho and B. H. Wilson, 2000, Natural Attenuation of MTBE in the Subsurface under Methanogenic Conditions, United States Environmental Protection Agency, Cincinnati, Ohio, EPA/600/R-00/006.

Wilson, J.T., R. Ross, and S. Acree, 2005, Using Direct-Push Tools to Map Hydrostratigraphy and Predict MTBE Plume Diving, Ground Water Monitoring and Remediation, 25(3), 93–102.

Appendix 1: Script D1_methodology and Examples of its Applications

```
%D1_methodology

%All solutions are based on equation h(x)^2=c^2+bx-(N/K)x^2
%In cases 1 to 2, coefficients 'c' and 'b' are calculated.
%Detailed Algorithm is presented in D1 Flow: Complete Methodology:
%3. Solutions for uniform horizontal aquitards

R1_='SIMULATED AQUIFER THICKNESS is KEPT in ARRAY H   ';
R2_='To ACCESS H at LOCATION x, TYPE H(100*x+1)       ';
R3_='Thus,H at x=25 CAN be OBTAINED by CALLING H(2501)';

%clearing previous graphs
%Input Data on Object
%--------------------
K=60.96;
N=558.8/1000/365*30;
L=1066.8;
%-------------------

%Preparing Object Relating Data for Calculations
%----------------------------------------------
W=N/K;
dX=0.01;
X=0:dX:L;
n=length(X);
H2=zeros(1,n);
%----------------------------------------------
 %Prompts Calling for Different Cases;
C='Input the number of appropriate boundary conditions                 ';
C0='from the following list:                                           ';
C1='1. Thickness are known at two points x1 and x2;                   1';
C2='2. Thickness h is known at point x1 and Flux Q at point x2:        2';
S_='                                                                   ';
txt=[C;C0;S_;C1;S_;S_;C2;S_];
disp(txt);
%-----------------------------------
k=input ('Input appropriate number from the above list: k=');
disp(S_);
%Prompts for Boundary Conditions Inputting for Called Cases
%-------------------------------------------------------

x1_='Input coordinate x1 where thickness h1 is known: x1=';       %C1&C2
h1_='Input thickness at x1: h1=h(x1)=';                           %C1&C2
x2_='Input coordinate x2 where thickness h2 is known: x2=';       %C1
h2_='Input thickness at x2: h2=h(x2)=';                           %C1
x2hQ_='Input coordinate x2 where Flux Q is known: x2=';           %C2
Q_='Input Flux Q at x2: Q=Q(x2)=';                                %C2
%-------------------------------------------------------
```

```
 switch k;
   case 1
       x1=input(x1_);
       h1=input(h1_);
       x2=input(x2_);
       h2=input(h2_);
       Q1=(h1^2-h2^2)*K/2/(x2-x1)-N*(x2-x1)/2;
       Q0=Q1-N*x1;
       b=-2*Q0/K;
       c=h1^2-b*x1+W*x1^2;
   case 2
       x1=input(x1_);
       h1=input(h1_);
       x2=input(x2hQ_);
       Q=input(Q_);
       Q0=Q-N*x2;
       b=-2*Q0/K;
       c=h1^2-b*x1+W*x1^2;
end
H=c+b*X-W*X.*X;
H=sqrt(H);

hold off;
plot(X,H);
grid
hold on
[x,h]=ginput();

results=[x';h']
plot(x,h,'mo')

if Q0*(Q0+N*L)<0
   waterDivide= L/2-(HB(1)^2-H(n)^2)/2/L/W
   wd_Index=round(100*waterDivide)+1;
   H_at_waterDivide=H(wd_Index)
end
disp([R1_;R2_;R3_])
%end of D1_Methodology
%-------------------------------------------/
```

Examples

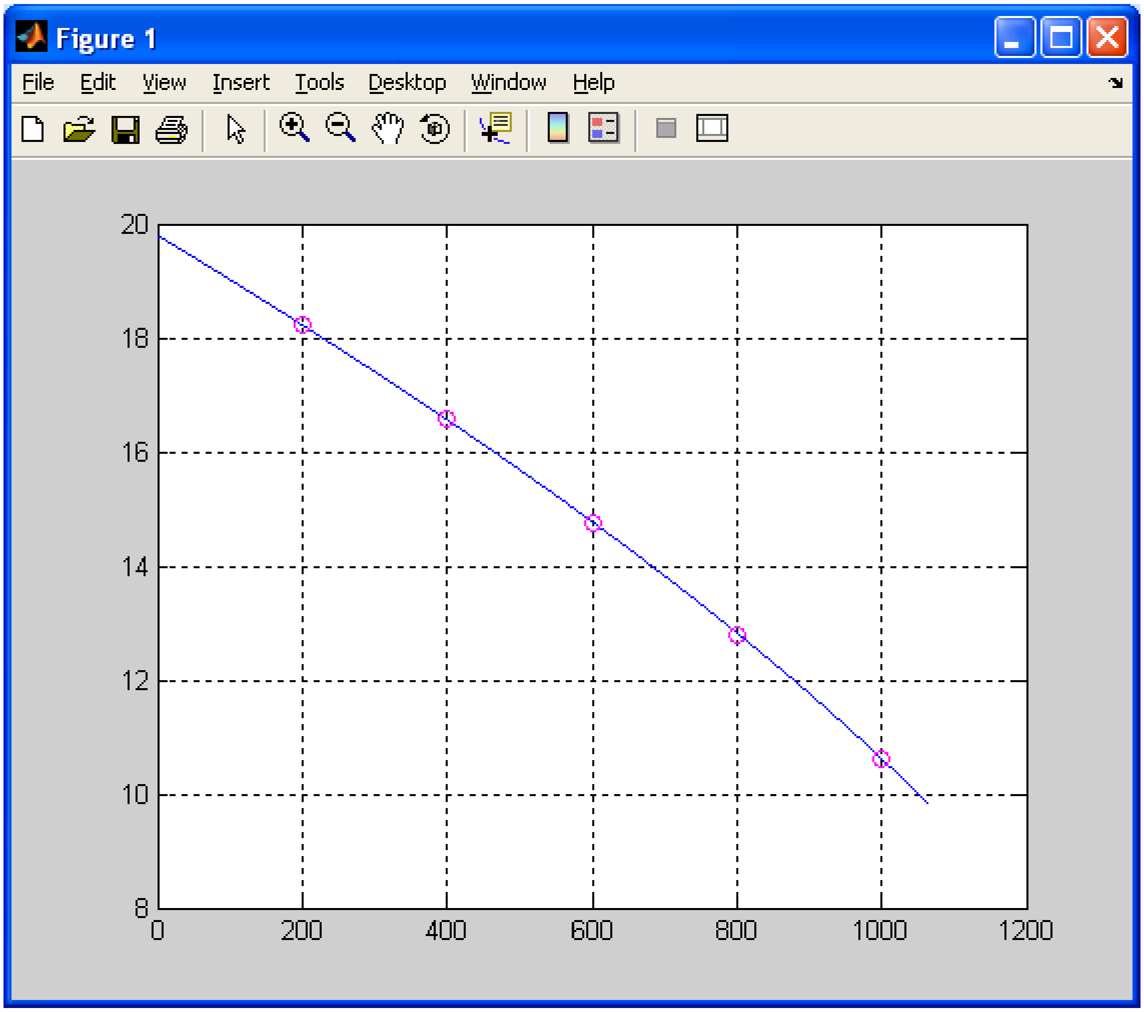
There are three groups of examples below. In first one, the water is monotonic. In the second 2 there exist water dividers inside of the interval of interest. In group 3, evaporation takes place
Data inputting from key board is in ***bold italic***.

Example 1**:** Characteristics of the Object: The length of the segment **L= 1066.8**
Recharge **N=558.8/1000/365**
Coefficient of filtration **K=60.96**

A1.
\>\> D1_methodology
Input the number of appropriate boundary conditions
from the following list:
1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**

Input appropriate number from the above list: **k=*1***

Input coordinate x1 where thickness **h1** is known: **x1=*0***
Input thickness at **x1: h1=h(x1)=*19.81***
Input coordinate **x2** where thickness **h2** is known: **x2=*L***
Input thickness at x2**: h2=h(x2)=*9.81***



Example A1: The thickness of the aquifer

results = ***1.0e+003 * 0.2005 0.3995 0.6014 0. 8005 0.9995*** *x coordinate*
18.2281 16.5789 14.7544 12.7895 10.6140 *thickness*
(The above is results of graphical evaluating of thickness at points x)

SIMULATED AQUIFER THICKNESS is KEPT in ARRAY **H**
To ACCESS **H** at LOCATION **x**, TYPE *H(100*x+1)*
Thus,H at **x=25** CAN be OBTAINED by CALLING ***H(2501)***
***H(200.46*100+1)*= 18.2323**

A2.
Input the number of appropriate boundary conditions
from the following list:
1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**

Input appropriate number from the above list: **k=*1***

Input coordinate x1 where thickness **h1** is known: **x1=*1000***
Input thickness at **x1: h1=h(x1)=*H1(1000*100+1)***
Input coordinate **x2** where thickness **h2** is known: **x2=*100***
Input thickness at x2**: h2=h(x2)=*H1(100*100+1)***

SIMULATED AQUIFER THICKNESS is KEPT in ARRAY **H**
To ACCESS **H** at LOCATION **x**, TYPE *H(100*x+1*)
Thus,H at **x=25** CAN be OBTAINED by CALLING ***H(2501)***

[H(1.4*100+1)	***H(399.5*100+1)***	***H(601.5*100+1)***	***H(800*100+1)***	***H(L*100+1)]***
19.8011	**16.9767**	**15.2465**	**13.2542**	**9.8100**

A3.
Input the number of appropriate boundary conditions
from the following list:
1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**
Input appropriate number from the above list: **k=*2***
Input coordinate x1 where thickness **h1** is known: **x1=*0***
Input thickness at **x1: h1=h(x1)=** ***19.81***
Input coordinate **x2** where Flux **Q** is known: **x2=*0***
Input Flux **Q** at x2**: Q=Q(x2) =*Q00***

SIMULATED AQUIFER THICKNESS is KEPT in ARRAY **H**
To ACCESS **H** at LOCATION **x**, TYPE *H(100*x+1*)
Thus,H at **x=25** CAN be OBTAINED by CALLING ***H(2501)***

[H(1.4*100+1)	***H(399.5*100+1)***	***H(601.5*100+1)***	***H(800*100+1)***	***H(L*100+1)]***
19.8011	**16.9767**	**15.2465**	**13.2542**	**9.8100**

A4.
Input the number of appropriate boundary conditions
from the following list:
1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**
Input appropriate number from the above list: **k=*2***
Input coordinate x1 where thickness **h1** is known: **x1=*L***
Input thickness at **x1: h1=h(x1)=** ***9.81***
Input coordinate **x2** where Flux **Q** is known: **x2=*500***
Input Flux **Q** at x2**: Q=Q(x2) =*Q0+N*500***

SIMULATED AQUIFER THICKNESS is KEPT in ARRAY **H**
To ACCESS **H** at LOCATION **x**, TYPE *H(100*x+1*)
Thus,H at **x=25** CAN be OBTAINED by CALLING ***H(2501)***

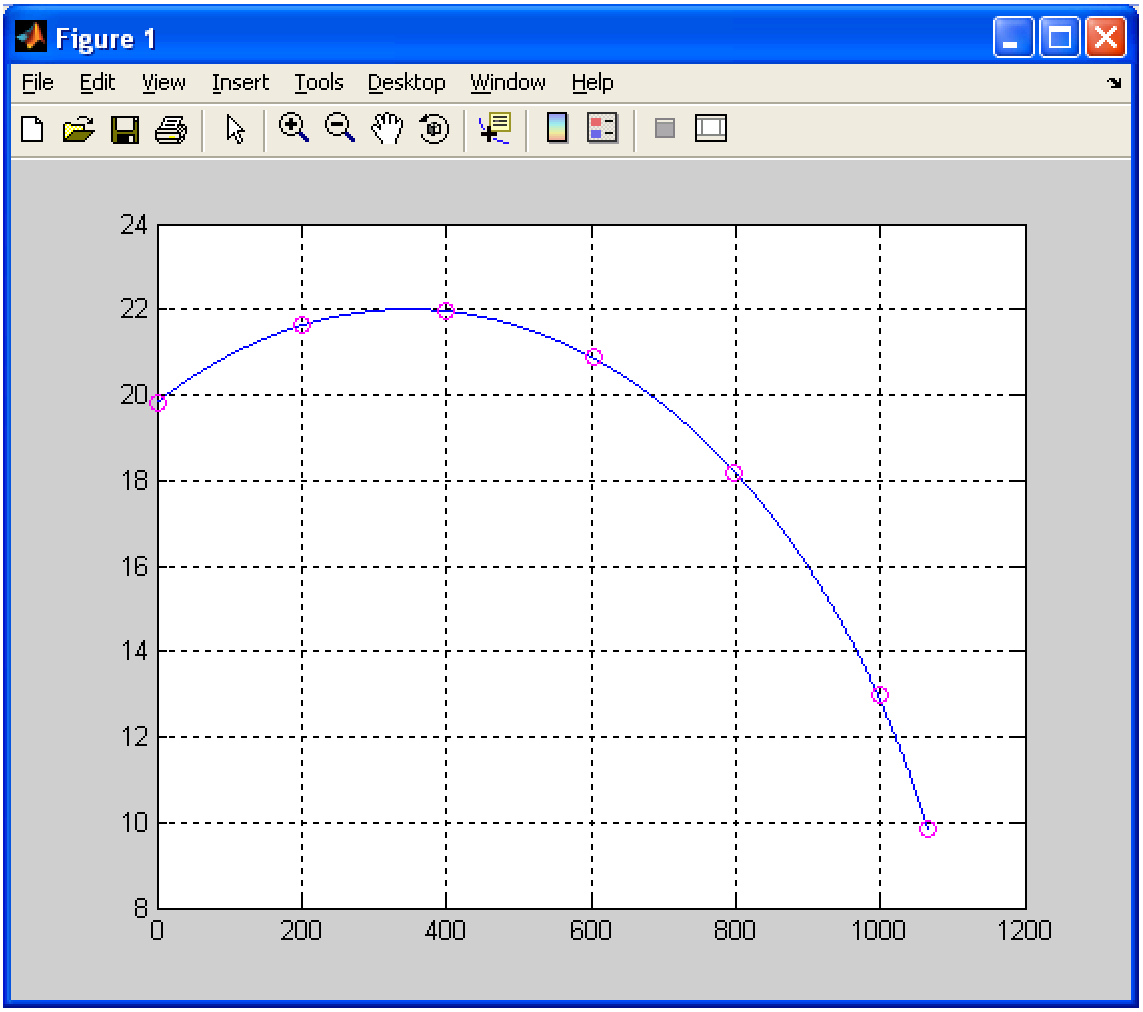
[H(1.4*100+1)	***H(399.5*100+1)***	***H(601.5*100+1)***	***H(800*100+1)***	***H(L*100+1)]***
19.8011	**16.9767**	**15.2465**	**13.2542**	**9.8100**

***Group B*:** Characteristics of the Object: The length of the segment **L= 1066.8**
Recharge **N=558.8/1000/365*30←**
Coefficient of filtration **K=60.96**

B1.
>> D1_methodology
Input the number of appropriate boundary conditions
from the following list:

1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**

Input appropriate number from the above list: **k=*1***
Input coordinate **x1** where thickness **h1** is known: **x1=*0***
Input thickness at **x1: h1=h(x1) =*19.81***
Input coordinate **x2** where thickness **h2** is known: **x2=*L***
Input thickness at **x2: h2=h(x2)=*9.81***



Example B1: The thickness of the aquifer

Results: **x=** 1.0e+003 *	***0.0014***	***0.2005***	***0.3995***	***0.6041***	***0.7977***	***0.9995***	***1.0659***
H=	**0.0198**	**0.0216**	**0.0219**	**0.0209**	**0.0182**	**0.0130**	**0.0098**
	H(141)	***H(20051)***	***H(39951)***	***H(60411)***	***H(79771)***	***H(99951)***	***H(106591])***
	19.8285	**21.6248**	**21.9628**	**20.8639**	**18.2396**	**12.8687**	**9.8595**

HB=H; Q00=Q0 = **-16.0355**

B2.
>> D1_methodology
Input the number of appropriate boundary conditions
from the following list:

1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**

Input appropriate number from the above list: **k=*1***
Input coordinate **x1** where thickness **h1** is known: **x1=*951***
Input thickness at **x1**: **h1=h(x1)**=***HB(951*100+1)***
Input coordinate **x2** where thickness h2 is known: **x2=732**
Input thickness at **x2**: **h2=h(x2)=*HB(732*100+1)***

H*(141)*	***H(20051)***	***H(39951)***	***H(60411)***	***H(79771)***	***H(99951)***	***H(106591***
19.8285	**21.6248**	**21.9628**	**20.8639**	**18.2396**	**12.8687**	**9.8595**

waterDivide **=** **349.1395**
H_at_waterDivide **= 22.0063**

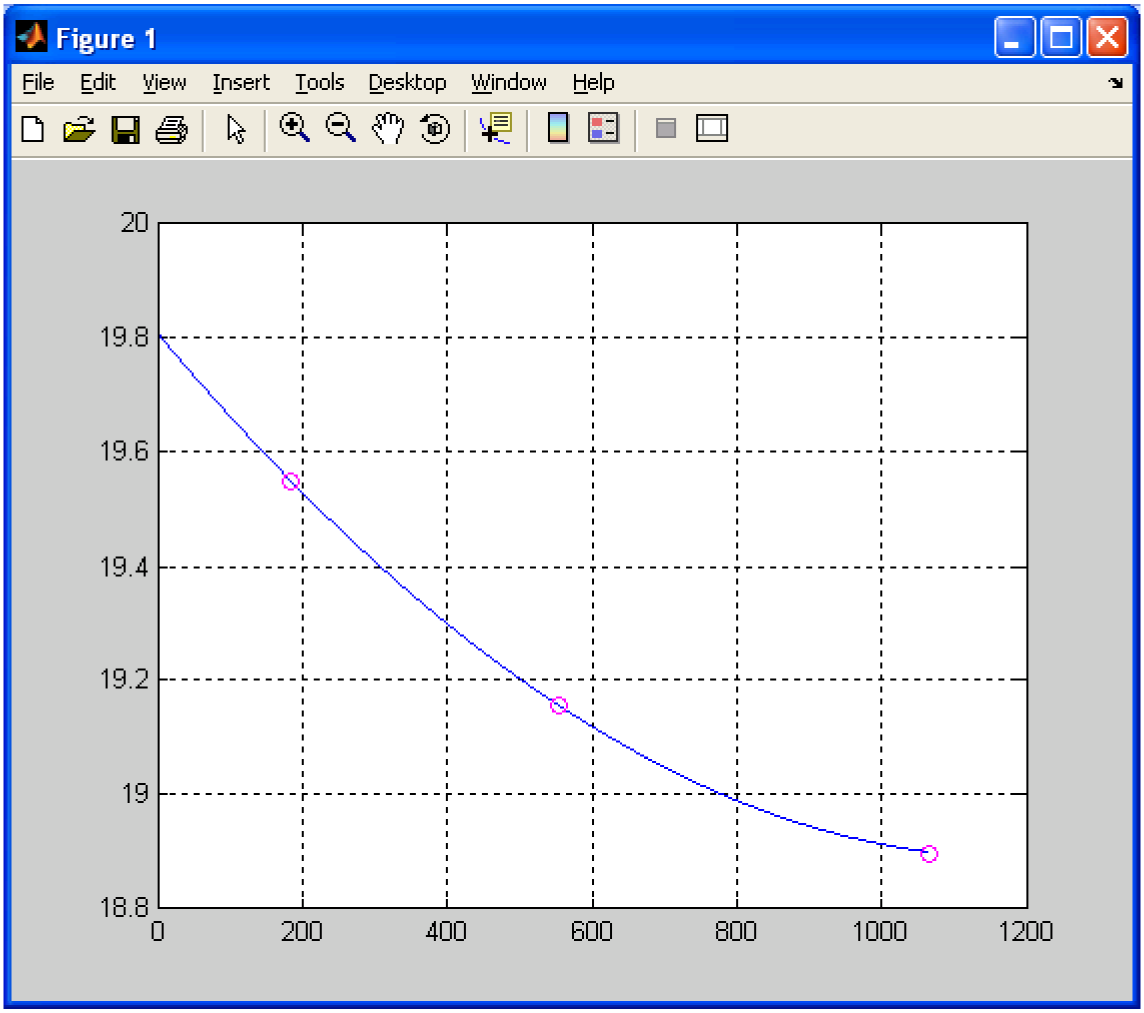
B3.
>> D1_methodology
Input the number of appropriate boundary conditions
from the following list:
1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness h is known at point **x1** and Flux **Q** at point **x2**: **2**
Input appropriate number from the above list: **k=*2***
Input coordinate **x1** where thickness **h1** is known**: x1=*300***
Input thickness at **x1: h1=h(x1) =*HB(100*x1+1)***
Input coordinate **x2** where Flux **Q** is known**: x2=*500***
Input Flux **Q** at **x2: Q=Q(x2) =*Q00+500*N***

H*(141)*	***H(20051)***	***H(39951)***	***H(60411)***	***H(79771)***	***H(99951)***	***H(106591)***
19.8285	**21.6248**	**21.9628**	**20.8639**	**18.2396**	**12.8687**	**9.8595**

waterDivide **=** **349.1395**
H_at_waterDivide **= 22.0063**

Group C: Characteristics of the Object: The length of the segment **L= 1066.8**
Recharge **N=— 558.8/1000/365** ←***Evaporation***
Coefficient of filtration **K=60.96**

>> D1_methodology
Input the number of appropriate boundary conditions
from the following list:
1. Thickness are known at two points **x1** and **x2**; **1**
2. Thickness **h** is known at point **x1** and Flux **Q** at point **x2**: **2**
Input appropriate number from the above list: **k**=*1*
Input coordinate **x1** where thickness **h1** is known: **x1=*0***
Input thickness at **x1: h1=h(x1)=*19.81***
Input coordinate **x2** where thickness **h2** is known: **x2=L**
Input thickness at **x2**: **h2=h(x2)=*18.9***
results = 1***.0e+003 **** ***0.1839*** ***0.5544*** ***1.0659***
0.0195 **0.0192** **0.0189**



Example C: The thickness of the aquifer

SIMULATED AQUIFER THICKNESS is KEPT in ARRAY **H**
To ACCESS **H** at LOCATION **x**, TYPE *H(100*x+1)*
Thus,H at **x=25** CAN be OBTAINED by CALLING ***H(2501)***

Appendix 2: Script 'hrzAQT'

```
%hrzAQT
```

%The script solve analytically filtration in unconfined heterogeneous aquifer on horizontal aquitard

```
clear

%Geometry of Object------------------------------------------\
n=6;
segEnds=[0 50 135 200 500 900 1000];%load homogeneous interval
                                    %boundaries
D=segEnds(2:n+1)-segEnds(1:n);      %lengths of intervals
L=segEnds(n+1);                      %left end of object
%End of Geometry---------------------------------------------/

%Properties of Object-------------------------\
K=[30 30 60 40 20 10];%m/day
N=10^(-4)*[15 50 15 10 20 10];%m/day

%Case of a Homogeneous Object----------\
if n==1
    n=2;
    K=[K K];
    N=[N,N];
    D=[L L]/2;
    xB=segEnds(1);
    segEnds=[xB,xB+D(1),L];
end %----------------------------------/

W=N./K;
aqBtm = 3.048;               %elevation of the aquifer bottom
h_0=11;%19.81;                   %elevation of the up gradient water
table
h_L=10;%9.81;                    %elevation of the down gradient water
table
%End of Properties of Object-------------------/

%Developing Boundary Conditions at the Ends of Object----------\
h0=h_0-aqBtm;              %left boundary condition
h02=h0^2;
hL=h_L-aqBtm;               %right boundary condition
hL2=hL^2;
%End of Developing Boundary Conditions at the Ends of Object---/

%Developping matrix for evaluation of model parameters-----------\
sizeC=2*n-1;                %size of the matrix of coefficients.
C=sparse(sizeC);            %declaring sparse matrix C
b=zeros(sizeC,1);           %declaring sparse right hand vector

%assigning coefficients for first equation: j=1------\
C(1,1)=-D(1);
C(1,2)=1;
b(1)=h02-W(1)*D(1)^2;%------------------------------/
```

```
%assigning coefficients for n_th equation: j= n------\
C(n,2*n-2)=1;
C(n,2*n-1)=D(n);
b(n)=W(n)*D(n)^2+hL2;--------------------------------/

%assigning coefficients for quations: j=2 to n-1-----\
for j=2:n-1                                                %|
C(j,2*j-2)=1;
C(j,2*j-1)=D(j);
C(j,2*j)=-1;
b(j)=W(j)*D(j)^2;
end %------------------------------------------------/

%assigning coefficients for quations: j=2 to n-1-----\
m=0;               %counter of homogeneous segments
for j=n+1:2*n-1
   m=m+1;
   i=2*m-1;
   C(j,i)=K(m);
   C(j,i+2)=-K(m+1);
   b(j)=2*N(m)*D(m);
end %------------------------------------------------/
%End of Developing matrix for evaluation of model parameters-----/

sol=C\b; %solving the system

%Converting Vector Solution in Matrix: -----------------\
%Three Coefficients for each homogeneous segment
A=zeros(n,3);
A(:,3)=-W;
A(1,1)=h02;
A(:,2)=sol(1:2:2*n-1);
A(2:n,1)=sol(2:2:2*(n-1));
%-----------------------------------------------------/

%Calculating Water Table:(h(x))^2=A(j,1)+A(j,2)*(x-L(j))+W((x-L(j))^2-\
%Preparing Arrays for calculating ------------\
step=.01;                    %increment of calculation (deltaX)
X=segEnds(1):step:L;         %creating X array
xL=length(X);
H2=zeros(1,xL);              %creating arrey for keeping squared
thickness
jH=round((segEnds-segEnds(1))/step)+1;%finding indexes for segment ends
%--------------------------------------------/
%Calculating Water Table------------------\
for j=1:n                   %loop calculating squared thickness H2
    xJ=X(jH(j));
    for i=jH(j):jH(j+1)
        H2(i)=[1 X(i)-xJ (X(i)-xJ)^2]*A(j,:)';
    end
end
H=sqrt(H2)+ aqBtm;              %H2-->elevation H
%--------------------------------------/

%Calculating flux at boundaries of homogeneous segments---\
Q0=-K(1)*A(1,2)/2;      %incoming flux
Q(1)=Q0;
```

```
for i=1:n
    Q(i+1)=Q(i)+N(i)*D(i);
end
%End of calculating array of flux ---------------------/

%Plotting Elevations-------\
plot(X,H)
grid on
hold on
%-------------------------/

%Graphical Evaluatiing of Elevations H ----------------\
T1='Do you want to evaluate of water table graphically?';
T2='If YES, type Y. If NO,type N. Strike ENTER.        ';
T3='Having finished grahical evaluation, strike ENTER. ';
disp([T1;T2;T3]); %displaying above prompt
reply =input('Thus: Y or N?  ','s');
if reply =='Y'
   [x,y]=ginput;
   disp('Results of Graphical Evaluations:')
   disp([x';y'])
end
% -----------------------------------------------------/

%Analytical(Exact)Evaluatiing of H at assigned locations x---\
T4='Do you want to evaluate of water table analytically    ';
T5='at locations assigned by you?                          ';
T6='If YES, type Y. If NO,type N. Strike ENTER.            ';
T7='Type x coordinate of desired location and strike Enter:';
T8='Having finished evaluation, type desired coordinate x  ';
T9='that is out of the object and strike ENTER.            ';
disp([T4;T5;T6]); %displaying above prompt
reply =input('Thus: Y or N?  ','s');
if reply =='Y' %evaluation=search for proper element in H array---\
    i=0;    %three lines preparing search
    L0=X(1);
    x=L/2;
    while (x>=L0|x<=L)
        disp([T8;T9])
        x=input(T7);
        if (x<L0|x>L) %exit from searc
            break;
        end
        i=i+1;
        XH(i)=x;
        evX=round(x/step)+1;
        evH(i)=H(evX);
        disp([XH(i),evH(i)]);
    end
    if i>0
    disp('Results of Analytical Evaluations:')
    disp([XH;evH])
    end
end %-----------------------------------------------------------/
disp('The program is terminated')
%end of hrzAQT
%--------------------------------------------------------------/
```

Appendix 3: Script 'hrzD1_Unvrsl'

```
%hrzD1_Unvrsl

%Script for calculating elevations of water table in heterogeneous object
%with horizontal aquitard that has elevation hAQT

object;
prompts;
disp(txt);
k=input ('Input appropriate number from the above list: k=');    %inputting k
if k==1
   CASE_1;
elseif k==2
   CASE_2;
else                              % k is not equal to1 or 2
   disp('Wrong k. Program is terminated')
   return
end
flux_Q;                           %array of fluxes at segment boundaries
h0_squared;                       %calculating squared thickness of aquifer
                                  %at most left boundary of object
waterTable;                       %calculating elevations of water table
display_;                         %displays elevations and flux at ends of object
waterDivide;                      %coordinate and elevation of Water Divide if it exists
graph_H;                          %graphing elevations if desired
graphINPUT;                       %obtaining data from graph of elevations, if desired
anltc_H;                           %obtaining elevations analytically, if desired
disp(txtR)                        % explaining how to get elevation for a given
                                  % coordinate when the code is terminated
hold off
%End of hrzD1_Unvrsl
%------------------------------------------------------------/

%object

%Script Introducing Object

name='Object 2'; %name of object must be typed here in single quotes
%Geometry-------------------------------\
n=6;       %number of homogeneous segments
segEnds=[0 50 135 200 500 900 1000]; %ends of homogeneous segments
%End of Geometry------------------------/
%Properties-------------------------------------\
K=[30 30 60 40 20 10];  %coefficients of filtration of segments
N=10^(-4)*[15 50 15 10 20 10];  %recharge within homogeneous segments
hAQT = 3.048;   %elevation of the aquifer bottom
%End of Properties--------------------------------/
%End of Object 2-------------------------------------/
disp=(name)                       %displays object's name

%Preparing Working Arrays: --------------------------\
D=segEnds(2:n+1)-segEnds(1:n);      %lengths of intervals
L=segEnds(n+1)- segEnds(1);        %length of object
W=N./K;
```

```
Q=zeros(1,n+1);       %preparing array of flux
step=.01;             %default increment of calculation (deltaX)
                 %for another increment, change step
X=segEnds(1):step:L;  %array x coordinates where value of H are
                             %calculated
xL=length(X);         % number of elements in array X
jH=round(segEnds/step)+1;    %array of indexes for segment ends
H=zeros(1,xL);              %array for calculated elevations
%Preparing Working Arrays: --------------------------/
%End of object
%-------------------------------------------------------------/

%prompts

%Set Including Most of Prompts

C_='Input the number of appropriate boundary conditions          ';
C0_='from the following list:                                   ';
C1_='1.Elevations h1 and h2 are known at two points x1 and x2: k=1';
C2_='2.Elevation is known at point x1 and Flux Q at point x2:  k=2';
S_='                                                             ';
txt=[C_;C0_;S_;C1_;C2_;S_]; %in text array txt all elements have the same length:61
                                        % although it may not be seen here
x1_='Enter coordinate x1 where elevation h1 is known: x1=';    %cases 1&2
h1_='Enter elevation at x1: h1=h(x1)=';                        %cases 1&2
x2_='Enter coordinate x2 where elevation h2 is known: x2=';    %case 1
h2_='Enter elevation at x2: h2=h(x2)=';                        %case 1

xQ_='Enter coordinate x2 where Flux Q is known: x=';           %case 2
Q_='Enter Flux Q at x2: Q=Q(x2)=';                             %case 2

xQ_='Enter coordinate x2 where Flux Q is known: xQ=';          %case 2
Q_='Enter Flux Q at x2: Q=Q(x2)=';                             %case 2

T1_='Do you want to evaluate of water table graphically?    ';
T2_='If YES, enter Y. If NO, enter N.                       ';
T3_='Having finished graphical evaluation, strike ENTER.    ';
txtT=[T1_;T2_;T3_];

T4_='Do you want to evaluate of water table analytically    ';
T5_='at locations assigned by you?                          ';
txtA1=[T4_;T5_;T2_];

T6_='Enter x coordinate of desired location                 ';
T7_='Having finished evaluation, enter such coordinate x    ';
T8_='that is out of the object and strike ENTER.            ';
txtA2=[T6_;T7_;T8_];

G_='Do you want graph? If Yes, enter Y. If No,enter N.';

R1_='SIMULATED ELEVATIONS are SAVED as ARRAY H         ';
R2_='To ACCESS H at LOCATION x, TYPE H(100*x+1)        ';
R3_='Thus, H at x=25 CAN be OBTAINED by CALLING H(2501)';
txtR=[R1_;R2_;R3_];
%End of prompts
 %------------------------------------------------------/
```

```
%CASE_1

%boundary conditions: elevations at two locations: k=1

input_1                    %input x1 and h1 and x2 and h2
reordering;                %if x1< x2 than x1 and h1 become x1 and h2and x1 and h2 become x1 and h2
intervals1;                %evaluates intervals to which x1and x2 belong
fluxQx1_Case1;  %calculates flux at x1 when x1 and x2 belong to the same homogeneous %segment
auxMaking                  %creates auxiliary object in interval [x1, x2] to evaluate flux at x1
systemMatrix;     %developing matrix based on auxiliary object
systemSolving;    %finding coefficient permitting to evaluate flux at x1
fluxQx1_Case1a;            % calculates flux at x1 when x1 and x2 belong to different homogeneous
                           %segments
%-End of CASE_1
%----------------------------------------------------------/

%input_1

% CASE_1: k=1: input (x1, h1) and (x2, h2)-----\

x1=input(x1_);
h1=input(h1_)-hAQT; %hAQT=elevation of aquitard
x2=input(x2_);
h2=input(h2_)-hAQT;
%-End of input_1
%---------------------------------------------

%reordering

%CASE_1 assumes that x1<x2
%If it is not, this script reorders x1 and x2

if x1>x2  %reordering x1 and x2 in case of jx1>jx2---\
         t=x2; x2=x1; x1=t;
         t=h2; h2=h1; h1=t;
end
%End of reordering
----------------------------------------------------------------------------/

intervals1

%Script for finding intervals for x1 and x2 for CASE_1

jx1=1;   %Finding segment to which x1 belongs------\
while x1>=segEnds(jx1+1) jx1=jx1+1; end
%x1 belongs to segment #jx1-----------------------/
if x2<=segEnds(jx1+1)
    jx2=jx1;
else
   jx2=jx1;   %Finding segment to which x2 belongs------\
   while x2>segEnds(jx2+1) jx2=jx2+1;end
   if x2==segEnds(jx2) jx2=jx2-1;end
   %x2 belongs to segment #jx2-----------------------/
end
d=x1-segEnds(jx1); %distance from the left segment boundary and x1
```

```
% end of intervals1
%------------------------------------------------------------------------------------------/




%fluxQx1_Case1;           %calculates flux at x1 when x1 and x2 belong to the same homogeneous segment

%Finding flux at x1 for Case 1
%x1 and x2 belong to same segment--------------------------\

Qx1 =(h1^2-h2^2)*K(jx1)/(x2-x1)/2-N(jx1)*(x2-x1)/2;
%End of case of x1 and x2 belong to same segment--------/
%End fluxQx1_Case1
%------------------------------------------------------------------------------------------/

%auxMaking

%Making up Auxiliary Object, Based on Segments of Initial Object
%within Interval [x1,x2]----------------------------------------\

nA=jx2-jx1+1;              %number of segments in Auxiliary Object
aEnds=zeros(1,nA+1);       %preparing array for description of Auxiliary Object
aN=zeros(1,nA);            %----------------------------------------------------------
aK=zeros(1,nA);            %----------------------------------------------------------
aW=zeros(1,nA);            %----------------------------------------------------------
%Making First and Last aSegments-----------------------------------\
aEnds(1)=x1;          %left end of the first segment
aEnds(nA+1)=x2;     %right end of the last segment
j=1;
m=jx1;
if x1<segEnds(jx1+1)
   aK(j)=K(jx1);                  %K of the first segment
   aN(j)=N(jx1);                  %N of the first segment
   aD(j)=segEnds(jx1+1);       %length of the first segment
   j=j+1;
   m=m+1;
end
if x2>segEnds(jx2)
   aK(nA)=K(jx2);                 %K of the last segment
   aN(nA)=N(jx2);;                %N of the last segment
   aD(nA)=x2-segEnds(jx2);     %length of the last segment
end
% First and last segment of aObject are done-----------------------/
 %Making up Internal Segments of aObject---------------------------------------------\
for i=m:jx2
   aEnds(j)=segEnds(i);    %end of aObject segment i
   aK(j)=K(i);
   aN(j)=N(i);
   j=j+1;
end                          %End of making up internal segments of aObject-----/
 aD=aEnds(2:nA+1)-aEnds(1:nA);    %length of aSegments
aW=aN./aK;                             %W of aSegments
%End of  auxMaking
%------------------------------------------------------------------------------------------/
```

```
%systemMatrix

%Developing Matrix for Auxiliari Object to Finding Qjx1--------------\

sizeC=2*nA-1;              %size of the matrix of coefficients.
C=sparse(sizeC);           %declaring sparse matrix C
b=zeros(sizeC,1);          %declaring  right hand vector

%Assigning coefficients for first equation: j=1 based on known h1-----------\
C(1,1)=-aD(1);
C(1,2)=1;
b(1)=h1^2-aW(1)*aD(1)^2;
%End of assigning coefficients for first equation: j=1----------------------------/
%Assigning coefficients for nA_th equation: j= nAbased on known h2----\
C(nA,2*nA-2)=1;
C(nA,2*nA-1)=aD(nA);
b(nA)=aW(nA)*aD(nA)^2+h2^2;
%End of assigning coefficients for nA_th equation: j= nA--------------------/
%Assigning coefficients for equations: j=2 to nA-1based continuty of elevations at boundaries---------\
   for j=2:nA-1
      C(j,2*j-2)=1;
      C(j,2*j-1)=aD(j);
      C(j,2*j)=-1;
      b(j)=aW(j)*aD(j)^2;
   end %End assigning coefficients for equation: j=2 to nA-1-----------------------------------------------------/
%Assigning coefficients Equations j=nA+1 to 2*nA-1expressing continuity of flux: ------------------------\
m=0;                       %counter of aSegments
for j=nA+1:2*nA-1
   m=m+1;
   i=2*m-1;
   C(j,i)=aK(m);
   C(j,i+2)=-aK(m+1);
   b(j)=2*aN(m)*aD(m);
end
%End of assigning coefficients for equations: j=nA+1 to 2*nA-1-----------------------------------------------/
%End of  systemMatrix
%------------------------------------------------------------------------------------------/

%systemSolving

 sol=C\b;
%End of %systemSolving
%------------------------------------------------------------------------------------------/


%fluxQx1_Case1a
%Finding flux at x1 for Case 1a: h1 and h2 are known in different segments

Qx1=-sol(1)/2*aK(1);
%End of fluxQx1_Case1a
%------------------------------------------------------------------------------------------/



%CASE_2                    %boundary conditions: elevation at one location
```

```
                                        %and flux in one location: k=2
    %input_2
    %intervals2
    %fluxQx1_Case2
%End of CASE_2
%-------------------------------------------------------------------------------------------/


%input_2

%Input for CASE_2:   (x1,h1) and (x2, Qx2) are given-------------\

x1=input(x1_);                  %location with known h1
h1=input(h1_)-hAQT;             % h1 - elevation of aquitard
x2=input(xQ_);                  %location with known flux Qx2
Qx2=input(Q_);
%End of input_2
%--------------------------------------------------------------------------/


%intervals2
%Script for finding intervals for x1 and x2 for CASE_2

jx1=1;   %Finding segment to which x1 belongs----\
while x1>segEnds(jx1+1) jx1=jx1+1; end
%x1 belongs to segment #jx1---------------------------/

jx2=1;               %Finding segment to which x2 belongs------\
while x2>segEnds(jx2+1) jx2=jx2+1;end
%x2 belongs to segment #jx2----------------------------------------/
 d=x1-segEnds(jx1);          %distance from the left segment boundary and x1
%End of ‘intervals2
%-------------------------------------------------------------------------------------------/

%fluxQx1_Case2

%Script for calculating flux Qx1 at x1 in CASE_2--------------------\

if jx1==jx2 Qx1=Qx2+(x1-x2)*N(jx1);  %Done for case with x1 and x2
                                  %belonging to the same segment
elseif jx1>jx2  %x1 belongs to a segment to right from the segment of x2
    Qx1=Qx2+N(jx2)*(segEnds(jx2+1)-x2)+N(jx1)*d;
    for j=jx2+1:jx1-1
        Qx1=Qx1+N(j)*D(j);
    end
    %Done for case jx1>jx2
else           %x1 belongs to a segment to left from the segment of x2
    Qx1=Qx2-N(jx2)*(x2-segEnds(jx2))+N(jx1)*d;
    for j=jx2-1:-1:jx1
        Qx1=Qx1-N(j)*D(j);
    end
end
%Done for case jx1<jx2
%End of fluxQx1_Case2
%-------------------------------------------------------------------------------------------/
% Note: goal of CASE_1 and CASE_2 is to find flux at x1, Qx1.
```

```
% Knowledge of Qx1 permits evaluating flux Q at all segment boundaries
% Knowledge of Q permits calculation thickness at all segment boundaries
% Knowledge of Q(1) and  h0_squared at the most left ends of object (at segEnds(1))
% permits on calculation elevations of water table with increment 'step'



%flux_Q

%Calculating Array of Flux at Boundaries of Object;

Q(jx1)=Qx1-N(jx1)*d;
if jx1>1
   for j=jx1:-1:2
      Q(j-1)=Q(j)-N(j-1)*D(j-1);
   end
end
for j=1:n
      Q(j+1)=Q(j)+N(j)*D(j);
end
%Ends of flux_Q;
%-------------------------------------------------------------------------------------------/


%h0_squared

%Calculating Squared Thickness of Aquifer at Most Left Boundary of Object

h02=h1^2+2*Q(jx1)/K(jx1)*d+W(jx1)*d^2;
if jx1>1
   for j=jx1-1:-1:1
      h02=h02+2*Q(j)/K(j)*D(j)+W(j)*D(j)^2;
   end
end
%End of h0_squared
%-------------------------------------------------------------------------------------------/


%waterTable

%Calculating Water Table:(hj(x))^2=hj^2+bj*(x-L(j))-Wj((x-Lj)^2--------\
%with bj=-2*Qj/Kj

jH=round(segEnds/step)+1;   %finding indexes for segment ends
H2(1)=h02;                   %squared thickness at left end
for j=1:n                    %loop calculating squared thickness H2
   p=-2*Q(j)/K(j);
   w=W(j);
   h2=H2(jH(j));
   x=segEnds(j);
   for i=jH(j):jH(j+1)
      H2(i)=h2+p*(X(i)-x)-w*(X(i)-x)^2;
   end
end
H=sqrt(H2)+ hAQT;            %H2-->elevation H
%End of waterTable
```

```
%--------------------------------------------------------------------------------------------/


%display_
 disp('Elevations at Ends of Object, H0 and HL:')
disp(['      H0        HL'])
disp([H(1) H(L*100+1)])
disp('Flux at Ends of Object, Q0 and QL:')
disp(['      Q0        QL'])
disp([Q(1) Q(n+1)])
%End of display_
%--------------------------------------------------------------------------------------------/


%waterDivide

%Calculating Coordinate and Elevation at Water Divide if it exists

if Q(1)*Q(n+1)<=0
   [M,I]=max(H);
   xWD=(I-1)/100;
   disp('Water Divide at x=')
   disp(xWD)
   disp('Elevation at Water Divide=')
   disp(M)
end
%End of waterDivide
%--------------------------------------------------------------------------------------------/


%graph_H
disp(G_)              %prompts asking to whether the option is desired
graph_=input('Y/N?','s');
if graph_=='Y'
   plot(X,H,'b')
   grid on
   hold on
end
%End of graph_H
%--------------------------------------------------------------------------------------------/


%graphINPUT

disp([T1_;T2_;T3_]);        %prompts asking to whether the option is desired
reply =input('Thus: Y or N?  ','s');
if reply =='Y'
   if graph_~='Y'
      plot(X,H,'b')
      grid on
      hold on
   end
   [x,y]=ginput;
   plot(x,y,'mo')
   disp('Results of Graphical Evaluations:')
   disp([x';y'])      %initial x and y are vertical vectors, but they displayed as horizontal ones
```

```
end
%End of graphINPUT
%-----------------------------------------------------------------------------/


%anltc_H

%Analytical Calculating Elevations if Desired

disp(txtA1)                    %prompts asking to whether the option is desired
reply =input('Thus: Y or N?  ','s');
if reply =='Y'
   i=0;
   L0=X(1);
   x=L/2;
   while (x>=L0|x<=L)
      disp([T8_;T2_])
      x=input(T7_);
      if (x<L0|x>L)
         break;
      end
      i=i+1;
      XH(i)=x;
      evX=round(x/step)+1;
      evH(i)=H(evX);
      disp([XH(i),evH(i)]);
   end
   if i>0
      disp('Results of Analytical Evaluations:')
      disp([XH;evH])
   end
end
%End of anltc_H
%-------------------------------------------------------------------------------------------/
```

Appendix 4: Script 'D_1 Flow'

Program '*D1_Flow*' solves numerically one – dimensional flow problem in homogeneous and heterogeneous unconfined aquifers with horizontal and non-horizontal bottoms assuming that the flow is horizontal (the Dupuit assumptions). It calculates water table elevations, fluxes at any point of the object, stream functions, plume counters, travel time for contaminant particles. It is universal in the sense that it solves all the problems that the all scripts described in this report do and even more.

The program consists of set of scripts. All its variables are global.

Listing of '*D1_Flow*'
In this listing the factual information is that fron the Borden Landfill calculations.

```
%D1_Flow

clear all
subplot(2,1,1,'replace')
subplot(2,1,2,'replace')
prompts
object
caseChoosing
switch k
    case 1
        CASE_1
    case 2
        CASE_2
    otherwise   %k is not equal to 1 or 2
        disp('Wrong k. k must be 1 or 2. Program is terminated')
        return
end
if flag==1
    disp('Failure: cannot find interval containg Q at x1')
    disp('Program is terminated')
    return
end
if flag==2
    disp('Failure: cannot find Q1 at x1')
    disp('Program is terminated')
    return
end
waterTable
throwingErrors
if flag ==3
    disp('The aquifer cannot LET the Flow TROUGH')
    disp('under assigned Boundary Conditions or Recharge')
    disp('see the plot')
    disp('Program is terminated')
    return
end
plumeD1
disp(' ')
disp('If you wish to calculate and plot stream functions,')
disp('Enter command:  streamFunction   and follow instructions')
disp(' ')
disp('If you wish to calculate and plot travel time,')
```

```
disp('Enter command:  travelTime  and follow instructions')
%end of D1_Flow
%-----------------------------------------------------------------------------------------/


%prompts    Subscript of D1_Flow

%Set of Prompts for CASE_1 and CASE_2

C00='Input the number corresponding to assigned boundary conditions ';
C0_='from the following list:                                        ';
C1_='1.Elevations h1 and h2 are known at two points x1 and x2: k=1  ';
C2_='2.Elevation is known at point x1 and Flux Q at point x2:  k=2  ';
S0_='                                                                ';
txt=[C00;C0_;S0_;C1_;C2_;S0_];

x1_='Enter coordinate x1 where elevation h1 is known: x1=';     %cases
1&2
h1_='Enter elevation at x1: h1=h(x1)=';                         %cases
1&2
x2_='Enter coordinate x2 where elevation h2 is known: x2=';     %case 1
h2_='Enter elevation at x2: h2=h(x2)=';                         %case 1

xQ2_='Enter coordinate x2 where Flux Q2 is known: x2=';         %case 2
Q2_='Enter Flux Q2 at x2: Q2=Q(x2)=';                           %case 2
%End of prompts
%----------------------------------------------------------/

%object    Subscript of D1_Flow.

name='Borden Site';
disp(name)
%Geometry------------------------------\
segEnds=[0 140 300 600  800 1050];% m
n=length(segEnds)-1;        %number of homogeneous segments  *ones(1,n)
X0=segEnds(1);              %left end of the object
XL=segEnds(n+1);            %right end of the object
%End of Geometry------------------------/
%Properties-------------------------------------\
N=[7,34,10,20,10]/365/100;%plnmN=[32.7862,9.5269,-1.3694,2.7794, -
9.7853]
K=11.7*10^(-3)*60*60*24/100*[1 1 1 1 1];%

nEff=0.38*ones(1,n);  %
%Aquitard coordinates:-----------------------------------------\
%aqtX=[X0,XL];
%aqtH=[204.6,204.6];
%aqtX=[X0,100,350,600,800,XL];
%aqtH=[195.3,195.3,204.6,210.8,211.78,211.78];
aqtX=[X0 29.41 49.54 89.78 126.93 148.61 185.76 218.27 232.2 243.03
266.25 300.31 348.3 374.61 396.28 445.82 467.49 495.36 515.48 543.34
577.4 597.52 628.48 659.44 696.59 749.23 792.57 848.3 910.22 948.92
998.45 XL];
aqtH=[191.09 192.49 193.42 194.82 196.69 197.93 200.11 202.6 203.69
204.31 204.62 204.31 204.62 205.09 206.02 208.04 208.67 209.44 209.91
210.22 210.38 210.84 211 211 211.31 211.62 211.78 212.4 212.87 213.02
213.33 213.33];
```

```
%--------------------------------------------------------------/
%End of Properties--------------------------------/

%Preparing Working Arrays:
%descritizing X and assigning properties for each element of X--------\
step=0.01;                  %increment of calculations
                            %CHANGE HERE IF YOU WANT A DIFFERENT ONE
X=X0:step:XL;               %array for descritised X coordinates
xL=length(X);
H=zeros(1,xL);              %array for water table elevations
Q=H;                        %array of flux
arrK=H;                     %array of coefficients of filtration
arrN=H;                     %array of recharges
arr_nEff=H;
jEnds=zeros(1,n+1);         %array of indexies of segment ends
jEnds(1)=1;
jEnds=round((segEnds-X0)/step+1); %calculating the above indexes
for j=1:n       %filling descritized arrays arrK and arrN--------\
    arrK(jEnds(j):jEnds(j+1))=K(j);     %loop to fill properties
    arrN(jEnds(j):jEnds(j+1))=N(j);     %within segment j
    arr_nEff(jEnds(j):jEnds(j+1))=nEff(j);
end %end of filling arrays arrK, arrN and n_Eff------------------/
aquitardElevation           %interpolates aquitard elevations on every
                            %point of array X linearly
surface=bordenSurface;
              %End of preparing Working Arrays:--------------------/
eps=10^(-3);                %default acceptable error of calculations
                            %CHANGE HERE IF YOU WANT A DIFFERENT ONE
ITR=50;                     %default acceptable number of iterations
                            %CHANGE HERE IF YOU WANT A DIFFERENT ONE
itrtn='50 iterations are performed';
itr=0;                      %counter of iterations in CASE_1: assigned
                            %here to enable proceeding in CASE_2
flag=0;
%end of object
%-------------------------------------------/


%caseChoosing     Subscript of D1_Flow

%Two kinds of boundary conditions are optional:
%CASE_1:    k=1->h(x1)=h1; h(x2)=h2
%CASE_2:    k=2->h(x1)=h1; Q(x2)=Q2

disp(txt);  %string ‘txt’ from ‘prompts’ explains choice of k
k=input ('k? = ');
%End of caseChoosing
%-------------------------------------------/






%CASE_1     Subscript of D1_Flow

%calculates array Q of fluxes at points of array X
```

```
%when boundary conditions are assigned as: h(x1)=h1 and h(x2)=h2

input_1;                %inputs x1,h1,x2,h2
reordering              %if x1>x2, x2 becomes x1, x1 becomes x2,
                        %h1 becomes h2, and h2 becomes h1

Q1_intervalSearch       %looks for interval [minQ, maxQ] containing Q1
if itr>=ITR
  flag=1;
   return;
end                     %fails to find [minQ, maxQ]:terminates CASE_1
Q1_iteration            %iterations to find Q1=Q(x1)in [minQ, maxQ]
if itr>=ITR
    flag=2;
    return;
end                 %fails to find Q1:terminates CASE_1
Q1_Q                %calculating array Q at all points within Object;
%End of CASE_1
%---------------------------------------------/

%input_1        Subscript of D1_Flow

% CASE_1: k=1: input (x1,h1) and (x2, h2)-----\

disp('CASE 1')
x1=input(x1_);  %x coordinate of water table elevation h1
h1=input(h1_);   %water table elevation h1 at x1
x2=input(x2_);  %x coordinate of water table elevation h2
h2=input(h2_);     %water table elevation h1 at x2
%End of input_1
%---------------------------------------------/

%reordering     Subscript of D1_Flow

%CASE_1 assumes that x1<x2
%If it is not, script reorders x1 and x2

if x1>x2  %reordering x1 and x2 in case of jx1>jx2---\
    t=x2; x2=x1; x1=t;
    t=h2; h2=h1; h1=t;
end %reordering is finished-------------------------/
x1L=round((x1-X0)/step+1);   %index of x1
x2L=round((x2-X0)/step+1);   %index of x2
%End of reordering
%------------------------------------------/

%Q1_intervalSearch      Subscript of D1_Flow

%Search for interval such that QL and QR have different signs

minQ=-2.03;            %minQ=default minimal influx at x1
maxQ=8.2;             %maxQ=default maximal influx at x1
erMin=1;            %initial error for h1 evaluated with minQ
erMax=erMin;        %initial error for h1 evaluated with maxQ
itr=0;              %counter of iterations
z=0;
while erMin==erMax  %searching loop using backward Euler------------\
```

```
    itr=itr+1;
    minH=h2;
    maxH=h2;
    sm=sum(arrN(x1L:x2L-1))*step;%loop to calculate recharges at x2
    for i=x2L:-1:x1L+1   %loop calculating minH and maxH at x1----\
        minH=minH+(minQ+sm)*step/arrK(i)/(minH-hAQT(i));
        maxH=maxH+(maxQ+sm)*step/arrK(i)/(maxH-hAQT(i));
        sm=sm-arrN(i-1)*step;
    end      %end of loop calculating minH and maxH at x1---------/
    erMin=sign(minH-h1);    %calculating signs of error with minQ
    erMax=sign(maxH-h1);    %calculating signs of error with maxQ
    if erMin~=erMax         %the interval is found
        return
    end
    if erMin>0              %need to decrease minQ
        minQ=2*minQ;
    end
    if erMax<0              %need to increase maxQ
        maxQ=2*maxQ;
    end
end        %end of searching loop using backward Euler--------------/
if erMin==erMax
    z=1;
    disp(itrtn)
    return
end
%End of 'Q1_intervalSearch'
%-----------------------------------------/

%Q1_iteration           Subscript of D1_Flow

Q1=(minQ+maxQ)/2;   %minQ and maxQ come from Q1_intervalSearch
H(x2L)=h2;
stpK=step./arrK(1:xL);  %loop  to fast main loop below
sm=sum(arrN(x1L:x2L-1))*step; %summing recharge to find QR=Q1+Q(x2L)
for itr=1:ITR   %iterations to find Q1=Q(x1L)------------------------\
    QR=Q1+sm;;              %Q(x2L) for currently tested Q1
    for i=x2L:-1:x1L+1  %backward Euler to evaluate H(x1L)----\
       H(i-1)=H(i)+QR*stpK(i)/(H(i)-hAQT(i));
       QR=QR-arrN(i)*step;  %Q(i-1) for currently tested Q1
    end    %------------------------------------------------/
    er1=H(x1L)-h1;
    if abs(er1)<eps      % calculations are over
        H(x1L)=h1;  %substituting approximation H(x1L)by exact value h1
        break %return
    end
    if er1<0    %narrowing interval of search [minQ, maxQ]--\
       minQ=Q1;
    else
       maxQ=Q1;
    end        %-----------------------------------------/
    Q1=(minQ+maxQ)/2;
end
 %----------------------------------------------------------------/
if itr>=ITR
    z=2;
    disp(itrtn)
```

```
    return
end
%End for_Q1_iteration
%----------------------------------------/


%Q1_Q             Subscript of D1_Flow

%Fills array of flux Q based on Q1

Q(x1L)=Q1;
for i=x1L-1:-1:1    %going from x1 to the left end of object
    Q(i)=Q(i+1)-arrN(i)*step;
end
for i=x1L:xL-1   %going from x1 to the right end of object
    Q(i+1)=Q(i)+arrN(i)*step;
end
%End of Q1_Q
%----------------------------------------/


%CASE_2           Subscript of D1_Flow

%calculates array Q of fluxes at points of array X
%when boundari conditions are assigned as: h(x1)=h1 and Q(x2)=Q2
input_2
Q2_Q      %calculating array Q at all points within Object;
%End of Case 2
%----------------------------------------/


%input_2          Subscript of D1_Flow
%Input for CASE_2:   (x1,h1) and (x2, Q2) are given----------\
x1=input(x1_);  %location with known elevation of water table,h1
h1=input(h1_);  %h1 - elevation of water table at x1
x2=input(xQ2_); %location with known flux Q2
Q2=input(Q2_);  %flux at x2 (x2=x1 is permitted)
x1L=round(x1/step+1);   %calculating index for x1
x2L=round(x2/step+1);   %calculating index for x2
%End of input_2
%------------------------------------------------------------/

%Q2_Q             Subscript of D1_Flow

%Fills array of flux Q based on assigned Q2
Q(x2L)=Q2;
for i=x2L-1:-1:1
    Q(i)=Q(i+1)-arrN(i)*step;
end
for i=x2L:xL-1
    Q(i+1)=Q(i)+arrN(i)*step;
end
%End Q2_Q
%---------------------------------/


%waterTable       Subscript of D1_Flow
```

```
%calculates and plots array of water table elevations
%evaluates them graphically and analytically elevations
Hx1_H0          %calculates H in [X0, x1)in Cases 1 and 2
Hx21_HL         %calculates H in(x2,XL]in Case 1 and (x1,XL]in Case 2
waterTablePlotting      %plotting results
flag=0;
for i=1:xL              %checking whether water table elevation is
lesser
    if H(i)<hAQT(i)     %than aquitard elevation
        flag = 3;       %if Yes the script stops and throws the error
        break;
    end
end

waterDivide     %calculates coordinate and water table elevation at
                %water divide if it exists
onEndValues     %displays elevations and fluxes at ends of Object
%End of waterTable
%---------------------------------------------------------/


%Hx1_H0            Subscript of D1_Flow

%calculates array H in from x1 to the left boundary of Object'
%using backward Runge-Kutte method of 4th order
if x1L==1 return; end
dX=step/2;
stp=step/6;
H(x1L)=h1;
for i=x1L:-1:2
   Y1=hAQT(i);
   Y2=hAQT(i-1);
   Y=(Y1+Y2)/2;
   h=H(i);
   q1=Q(i);
   q2=Q(i-1);
   q=(q1+q2)/2;
   C1=arrK(i);
   C2=arrK(i-1);
   C=(C1+C2)/2;
   k1=q1/C1/(h-Y1);
   k2=q/C/(h+dX*k1-Y);
   k3=q/C/(h+dX*k2-Y);
   k4=q2/C2/(h+step*k3-Y2);
   H(i-1)=h+(k1+2*k2+2*k3+k4)*stp;
end
%End of Hx1_H0
%--------------------------------------------/


%Hx21_HL    %RK_4       Subscript of D1_Flow

%in CASE_1 (k=1):
%calculates array H from x2 to the right boundary of Object
%in CASE_2 (k=2)
%calculates array H from x1 to the right boundary of Object
```

```
%uses the 4th order forward Runge-Kutte method

if k==1         %CASE_1
    x21=x2;
    x21L=x2L;
    H(x21L)=h2;
else            %CASE_2
    x21=x1;
    x21L=x1L;
    H(x21L)=h1;
end
if x21==segEnds(n+1)    %water table is calculated by Hx2_Hx1 (CASE_1)
    return              %or by Hx1_H0 (CASE_2)
end
dX=step/2;
stp=step/6;
for i=x21L:xL-1
    h=H(i);
    Y1=hAQT(i);
    Y2=hAQT(i+1);
    Y=(Y1+Y2)/2;
    q1=Q(i);
    q2=Q(i+1);
    q=(q1+q2)/2;
    C1=arrK(i);
    C2=arrK(i+1);
    C=(C1+C2)/2;
    k1=-q1/C1/(h-Y1);
    k2=-q/C/(h+dX*k1-Y);
    k3=-q/C/(h+dX*k2-Y);
    k4=-q2/C2/(h+step*k3-Y2);
    H(i+1)=h+(k1+2*k2+2*k3+k4)*stp;
end
%End of Hx21_HL
%-----------------------------------------/

%waterTablePlotting           Subscript of D1_Flow
subplot(2,1,1)
plot(X,H,'b',X,hAQT,'k')
hold on
plot(surface(1,:),surface(2,:),'k')
grid on
xlabel('Distance (m)')
ylabel('Elevation A.S.L. (m)')
%end of script waterTablePlotting
%-----------------------------------------/

%throwingErrors               Subscript of D1_Flow

%throwing out errors if boundary conditions or recharge
%do not let flow through aquifer

for i=1:xL
    if H(i)<hAQT(i)
        flag=3;
        break;
    end
```

```
end

for i=1:length(surface(1,:))
    srfX=(surface(1,i)-X0)/step+1;
    if surface(2,i)<H(srfX)
        flag=3;
        break;
    end
end
%End of script throwingErrors
%-----------------------------------------/


%waterDivide            Subscript of D1_Flow

%Calculating Coordinate and Elevation at Water Divide
%if it exists

if sign(Q(1))~=sign(Q(xL))
    [M,I]=max(H);   %matlab function to find maximum H and its index
    wtDivIndx=I;
    xWD=(I-1)/100;  %recalculates index in x coordinate
    disp('Water Divide at x=')
    disp(xWD)   %displays coordinate of water divide
    disp('Elevation at Water Divide=')
    disp(M)   %displays elevation of wtar table at water divide
end
%End of script waterDivide
%-----------------------------------------/

%onEndValues      Subscript of D1_Flow

%displays water table elevations and fluxes at Object ends

H0=H(1);
HL=H(xL);
Q0=Q(1);
QL=Q(xL);

disp(' ')
disp('On the Object Ends:')
disp('   H0          Q0            HL          QL')
disp(num2str([H0 Q0 HL QL]))
%End of script onEndsValues
%-----------------------------------------/




%plumeD1          Subscript of D1_Flow

%Evaluates Elevations of Plume Top and Bottom
%Based on the Dupuit Assumption

disp('Do you want to trace POLLUTION PLUME?')
disp('If YES, enter Y and strike "ENTER".')
```

```
disp('If NO, strike,"ENTER":')
qw_='Thus, Y or N?';
reply=input(qw_,'s');
if reply=='Y'
   plumeZone
   if out == 1 return; end
   plumeLeftBoundary
   plumeRghtBoundary
else
    return
end
   plot(plumeTopX,plumeTopH,'r',plumeBtmX,plumeBtmH,'r')
%End of script plumeD1
%-----------------------------------------/


%plumeZone        Subscript of D1_Flow

%Defining Plume Zone and corresponding stream functions

out=0;          %indicator of error: no error
p1=input('Input coordinate of left, smaller, boundary of polluting
zone=');
if p1<X0||p1>XL
    out =1;     %indicator of error: error
    disp('Left boundary is out of object. Try again')
    return
end

p2=input('Input coordinate of right, greater, boundary of polluting
zone=');
if p2<X0||p2>XL
    out =1;     %indicator of error: error
    disp('Right boundary is out of object. Try again')
    return
end
pL=round((p1-X0)/step+1);    %index of p1
pR=round((p2-X0)/step+1);    %index of p2

qFL=Q(pL);              %flux at p1 (stream function staring at p1)
qFR=Q(pR);              %flux at p2 (stream function staring at p2)
%End of plumeZone
%-----------------------------------------/





%plumeLeftBoundary            Subscript of D1_Flow

znk=sign(qFL);
if znk ==0      %qF=0:Stream function coincides with aquitard surface--
-\
    plumeBtmX=X; plumeBtmH=hAQT;
    plumeBtmH(wtDivIndx)=H(wtDivIndx);   %qElv at water divide is not
defined
```

```
                                    %and is drawn as vertical line
    plot(plumeBtmX,plumeBtmH,'r')
    return
end      %script is done for calculating and plotting for qF=0--------/

stPt=pL;
if znk==1
   qX=zeros(1,xL-stPt+1);
else
   qX=zeros(1,stPt);
end
qElv=qX;
if znk==1
    jSt=1;
    jFn=length(qX);
else
    jSt=stPt;
    jFn=1;
end
i=stPt;
for j=jSt:znk:jFn  %Calculating elevations of stream function
    qX(j)=X(i);
    qElv(j)=H(i)-(H(i)-hAQT(i))*(1-qFL/Q(i));
    i=i+znk;
end
plumeBtmX=qX;
plumeBtmH=qElv;
%End of script plumeLeftBoundary
%-----------------------------------------/

%plumeRghtBoundary            Subscript of D1_Flow

znk=sign(qFR);
if znk ==0      %qF=0:Stream function coincides with aquitard surface-\
    plumeTopX=X; plumeTopH=hAQT;
    plumeTopH(wtDivIndx)=H(wtDivIndx);
    plot(plumeTopX,plumeTopH,'r')
    strFnct=[plumeTopX;plumeTopH]; %Preparing array for saving the
                        %result if needed
                        %The result should be saved with different name
    return
end      %script is done for calculating and plotting for qF=0-------/

stPt=pR;
if znk==1
   qX=zeros(1,xL-stPt+1);
else
   qX=zeros(1,stPt);
end
qElv=qX;
if znk==1
    jSt=1;
    jFn=length(qX);
else
    jSt=stPt;
    jFn=1;
end
```

```
i=stPt;
for j=jSt:znk:jFn  %Calculating elevations of stream function
    qX(j)=X(i);
    qElv(j)=H(i)-(H(i)-hAQT(i))*(1-qFR/Q(i));
    i=i+znk;
end
plumeTopX=qX;
plumeTopH=qElv;
%End of script plumeRightBoundary
%-----------------------------------------/


%streamFunction         Subscript of D1_Flow

%Finding stream function of given value qF
%Inputting and analyzing qF---------------------------\

qF=input('Input the value of stream function of interest: qF = ');
if qF<=Q0&qF<=QL
    disp('Too small stream function value qF');
    disp('Try the code again with larger qF value');
    return;
end
if qF>=QL&qF>=Q0
    disp('Too large stream function value qF');
    disp('Try the code again with smaller qF value');
    return;
end %-----------------------------------------------/
streamFunctionCalculation
plot(qX,qElv,'b')
strFnct=[qX;qElv]; %Preparing array for saving the result if needed
                   %The result should be saved with different name
disp('Results are in D2 array strFnctn(qX,qElv)')
disp('Rename the array if you want save them')
disp(' ')
disp('If you wish to continue calculating and plotting stream
functions.')
disp('Enter command:  streamFunction')
%End of streamFunctions
%-------------------------------------------------------------/


%streamFunctionCalculation         Subscript of D1_Flow

%Finding the point on water table where stream function starts
%and calculating

znk=sign(qF);
if znk ==0      %qF=0: qF starts at water divide----------------------\
    qX=X; qElv=hAQT;    %arrays for x coordinate and elevations
    qElv(wtDivIndx)=H(wtDivIndx); %at water divide qF is vertical
    plot(qX,qElv,'k')
    strFnct=[qX;qElv]; %Preparing array for saving the result if needed
                       %Result should be saved with different name
    return
end      %script is done for calculating and plotting for qF=0--------/
for i=1:xL-1   %Finding index of qF starting point qF
```

```
    if (Q(i)<=qF)&(qF<=Q(i+1))
        stPt=i;
        break
    end
end
if znk==1   %preparing arrays for qF>0
   qX=zeros(1,xL-stPt+1);   %array of x coordinate
    jSt=1;
    jFn=length(qX);
else           %preparing arrays for qF<0
    qX=zeros(1,stPt);
    jSt=stPt;
    jFn=1;
end
qElv=qX;  %array for elevations qElv

i=stPt;
for j=jSt:znk:jFn  %Calculating elevations of stream function
    qX(j)=X(i);
    qElv(j)=H(i)-(H(i)-hAQT(i))*(1-qF/Q(i));
    i=i+znk;
end
%-------------------------------------------------------------/
%End of streamFunctionCalculation


%travelTime     Subscript of D1_Flow

%calculated based on Dupuit Assumption and assumption that
%the shortest travel time is for the upper boundary of plume
%The longest travel time is for the lower boundary of plume

disp('Input x coordinate (stX), starting point of the stream
Function');
disp('for which you wish to calculate TRAVEL TIME:');

stX =input('stX ? =');        %Starting point of travel

if stX<X0
    disp('Too small stX');
    disp('Try the code again with larger stX value');
    return;
end
if stX>XL
    disp('Too large stX');
    disp('Try the code again with smaller stX value');
    return;
end
stPt=round((stX-X0)/step+1);
qF=Q(stPt);
if qF==0
    disp('Coordinate stX coincides with water divide and qF=0.')
    disp('There is no advection along qF=0 infor this model. Thus,')
    disp('TRAVEL TIME TO ANY POINT along qF=0 is INFINITY')
    disp('You may start again with command  travelTime CHANGING stX
slightly')
    disp('Code is terminated')
```

```
    return
end
trajectoryCalculation
travelTimeCalculation
%End of travelTime
%---------------------------------------------------------------/


%trajectoryCalculation      Subscript of D1_Flow

%Finding the point on water table where stream function starts
%and calculating

znk=sign(qF);

if znk==1   %preparing arrays for qF>0
   qX=zeros(1,xL-stPt+1);   %array of x coordinate
    jSt=1;
    jFn=length(qX);
else           %preparing arrays for qF<0
    qX=zeros(1,stPt);
    jSt=stPt;
    jFn=1;
end
qElv=qX;  %array for elevations qElv

i=stPt;
for j=jSt:znk:jFn  %Calculating elevations of stream function
    qX(j)=X(i);
    qElv(j)=H(i)-(H(i)-hAQT(i))*(1-qF/Q(i));
    i=i+znk;
end
%End of trajectoryCalculation
%---------------------------------------------------------------/


%travelTimeCalculation        Subscript of D1_Flow

time=zeros(1,length(qX));
i=stPt;
for j=jSt:znk:jFn-znk  %Calculating elevations of stream function
    m1=(qElv(j)+qElv(j+znk)-hAQT(i)-hAQT(i+znk));
    ds=sqrt(step^2+(qElv(j+znk)-qElv(j))^2);
    time(j+znk)=time(j)+znk*arr_nEff(i)*m1*ds/2/qF;
    i=i+znk;
end
time=time/365;
trvlTm=[qX;time];
subplot(2,1,2)
plot(qX,time,'k');
grid on
xlabel('x coordinate (m)')
ylabel('time (years)')
hold on

%Calculating travel time to location fnlX
disp('Do you want to know TRAVEL TIME to some LOCATION?')
```

```
disp('If YES, enter Y and strike "ENTER"')
disp('If NO, strike "ENTER":')
qw_='Thus, Y or N?';
reply=input(qw_,'s');
while reply=='Y'
    disp('Enter coordinate X of the LOCATION of interest')
    fnlX=input ('X? = ');
    answer=['travel time = ' num2str(time((fnlX-stX)/step+1))];
    disp(answer);
    disp('If you want to continue, enter Y. Otherwise just strike ENTER
key');
    reply=input(qw_,'s');
end
%End of travelTimeCalculation
%-------------------------------------------------------------/
```

United States
Environmental Protection
Agency

Office of Research
and Development (8101R)
Washington, DC 20460

Official Business
Penalty for Private Use
$300

EPA 600/R-07/122
September 2007
www.epa.gov

Please make all necessary changes on the below label, detach or copy, and return to the address in the upper left-hand corner.

If you do not wish to receive these reports CHECK HERE

☐; detach, or copy this cover, and return to the address in the upper left-hand corner.

PRESORTED STANDARD
POSTAGE & FEES PAID
EPA
PERMIT No. G-35

Recycled/Recyclable
Printed with vegetable-based ink on paper that contains a minimum of 50% post-consumer fiber content processed chlorine free