155: Numerical Models of Groundwater Flow and Transport

EKKEHARD HOLZBECHER[1] AND SHAUL SOREK[2]

[1] *Humboldt Universität, Inst. of Freshwater Ecology (IGB), Berlin, Germany*
[2] *Ben Gurion University of the Negev, J. Blaustein Institutes for Desert Research, Sede Boker, Israel*

The article gives an introduction to numerical modeling of flow and transport problems and to software tools that are currently in use for modeling such phenomena. Details are explained on numerical approximations leading to different numerical models. Extensions for reactive transport are mentioned. Basic guidelines and criteria are given that should be taken into account by the modeler in order to improve the accuracy of results. Inverse modeling is presented as an advanced feature. Some examples of pre- and postprocessing, as implemented in several codes, are given, in addition to fundamental properties of solution methods. Finally, most common codes are listed with basic features and web-sites.

INTRODUCTION

The forecast of state variables that describe flow and solute transport in a given aquifer can be obtained by solving the *mathematical model* that describes these phenomena. Such a model is based on a *conceptual* model that includes a set of verbal statements introducing a simplified version of the various physical, chemical, and biological aspects of the flow domain and the phenomena of transport that take place in it. Because, in most cases of practical interest, analytical solutions of the mathematical models are not possible, the mathematical models are transformed into *numerical models*, which, in turn, are solved by specially designed computer programs (= codes).

The codes account for physical aspects (e.g. multiphase/multicomponent, density-driven, chemical reactions, inertial/drag dominant, energy considerations, small/large matrix deformation, Newtonian/nonNewtonian fluids), modeling aspects (e.g. domain dimensionality, type of boundary conditions, Eulerian/Lagrangian formulation, deterministic/stochastic representations, lumped parameter/continuum/sharp interface approaches, phreatic/confined, unsaturated/saturated), and optimal management (i.e. mathematical procedures deriving the optimal extremum trajectories under different constraints and objectives). These are addressed by sensitivity and inverse methods, analytical and/or numerical (e.g. differences, virtual work/variational approaches) approximations, grid methods (e.g. finite differences, finite elements, volume elements, boundary elements, spectral elements), numerical algorithms (Eulerian/Lagrangian coordinates, particle tracking, explicit/implicit approximations, linear/nonlinear iteration procedures) and their associated reliability/efficiency measures (e.g. stability/monotonicity of the numerical scheme, Peclet/Courant grid based numbers). Such codes, which are the focus of this chapter, are applied by the modeler to set up models (in the literature sometimes the term model is used also for a code, which is not correct). Mostly the codes are executable software files, mostly equipped with user-friendly graphical user interfaces (GUIs). Sometimes source-codes are distributed, which have to be altered, compiled, and linked by the user.

A modeling task can be subdivided into several steps:

- Preprocessing (transformation of data into a format appropriate for the numerical algorithm, including grid generation)
- Numerical calculation (direct modeling)
- Calibration (inverse modeling)
- Postprocessing

Today's software packages or codes assist in all of these modeling steps.

Encyclopedia of Hydrological Sciences. Edited by M G Anderson.

Calibration as a task cannot be separated from the other tasks. Inverse modeling includes direct model runs, performed in order to determine one or more parameter values, which lead to an optimal approximation of measured results. Some pre- and postprocessing may be necessary during the inverse modeling procedure, when its not the standard in- and output variables, on which the calibration is based.

Many different software tools are today available to help users to set up their models. The aim of the models is to assist in the solution of practical problems, simulating processes in subsurface fluids and porous media. In the majority of cases, modeling serves to improve understanding of hydrogeological systems. Forecasting and thus studying the response due to different scenarios is the most ambitious goal of modeling efforts. In scientific literature, this is discussed under the term *validation*

- Validation is a process carried out by comparison of model predictions with independent field/experimental observations. A model cannot be considered validated until sufficient testing has been performed to ensure an acceptable level of predictive accuracy (IAEA International Atomic Energy Agency, 1988)
- Validation: the process of obtaining assurance that a model as embodied in a computer program is a correct representation of the process or system for which it is intended. (United States NRC – Nuclear Regulatory Commission, see Silling, 1983)

The task of code verification is a step towards validation in which numerical results are compared with analytical solutions or with well accepted published results.

Software tools can be subdivided into different classes, for which codes that perform numerical calculations are considered as the core software program. Around these packages have been developed for several pre- and postprocessing tasks. GMS, Visual MODFLOW, and PMWIN are examples, which are build around the MODFLOW code in the core – in most recent versions accompanied by other numerical codes. Other packages, like FEFLOW, embed all tasks in one package.

In judging a numerical code for simulating flow and transport scenarios imposed to a specific aquifer site, one should first verify what aspects are being addressed by the code. This in general accounts for:

- the theoretical and mathematical assumptions and other considerations;
- the numerical method, algorithm, and grid configuration;
- verifications of the code against analytical and numerical solutions as well as laboratory and field observations;
- performance under a variety of time and space increments;
- platforms on which the code can be run.

Input data that is required for code simulations can be classified into:

1. Geometry and topography issues
   (a) Site boundaries and dimensions
   (b) Surface topography (e.g. to detect zones with surface infiltration)
   (c) Location of streams, divides, ponds and so on
   (d) Land use (landfills, dikes, well locations, irrigation systems. . . .)
2. Geology and hydrology issues
   (a) Aquifers (stratification, depth, lithologic parameters, hydraulic conductivities, longitudinal and transversal dispersivities, storativities (i.e. matrix and water compressibilities), porosities)
   (b) Porous medium density
   (c) Water levels at surface reservoirs (rivers, ponds, etc.) compressing shallow aquifers
   (d) Pumping/recharge point sources (well depth, intensity, periodicity, and time of application)
   (e) Distributed sources of inflow, for example, rainfall and irrigation rates
   (f) Distributed sources of outflow, for example, evapotranspiration
   (g) Time dependent data at spatial points
3. Water and porous medium chemical properties
   (a) Sorption (adsorption and desorption) factors.
   (b) Electrical conductivities.
   (c) Temporal and spatial concentration of solutes in the water and the solid phases of the porous medium
   (d) Solutes associated with sources of recharge fluxes (rainfall, irrigation, etc.).
   (e) Concentration of stable isotopes and microelements.
4. Boundary and initial conditions
   (a) Initial field distribution of piezometric head and components concentration.
   (b) Pervious/impervious boundary segments with the ascribed flux conditions.
   (c) Piezometric heads and concentrations along boundaries.

The user has to take into account that codes often differ concerning the input parameters, an aspect that may make some codes more appropriate for a given task than others. As an example many codes allow conductivity anisotropy only in direction the principal axes of the coordinate system. When anisotropies in changing directions have to be taken into account, the modeler has to choose a code that is capable of handling such a situation.

MATHEMATICAL MODELS

Groundwater Flow

Groundwater flow models are based on the differential equations for groundwater flow. Such differential equations, as described in **Chapter 149, Hydrodynamics of Groundwater, Volume 4**, are usually based on Darcy's Law as the linear macroscopic fluid momentum balance equation, considering the drag terms of the Navier Stokes equation as dominant, and on the principle of the fluid mass conservation.

Depending on the special features of the situation to be modeled, different circumstances have to be taken into account. A model for a confined aquifer is different from that for an unconfined (phreatic) aquifer. The spatial dimensionality (1D or 2D or 3D) depends on the physical situation and the aim of modeling. Depending on the very same aspects, a decision about steady state versus unsteady simulations has to be taken, just to name the most basic properties of a model.

There are different formulations of the differential equations. Equation (1) states the mass balance in 3D:

$$\frac{\partial}{\partial t}(\phi\rho_f) = -\nabla\cdot(\rho_f \mathbf{v}) + Q \tag{1}$$

where ϕ denotes porosity, ρ_f fluid density, $\mathbf{v}$ the three-dimensional vector of Darcy velocity, that is specific discharge, and Q represents mass sources or sinks of whatever type. Most models work with a simplified version of equation (1), which is valid for constant density ρ_f. With the help of Darcy's Law, the equation can be reformulated in terms of hydraulic head h. Simplified 2D versions of equation (1) are used quite frequently, which are different for confined or unconfined aquifers. In the confined situation:

$$S\frac{\partial h}{\partial t} = -\nabla\cdot T\nabla h + P - Q \tag{2}$$

in which P and Q represent pumping and recharge rates, respectively, where S denotes the storativity and T the transmissivity. Usually the hydraulic head, h is the dependent prime variable, for which the differential flow equation is formulated and which is calculated by the model. In 2D problems the stream function can be used as an alternative (Holzbecher, 1996). For applications involving variable density, a generalized hydraulic head or pressure p have to be introduced (Holzbecher, 1998) as dependent prime variable.

The codes allow the specification of different boundary conditions, which are relevant for groundwater flow (*see* **Chapter 149, Hydrodynamics of Groundwater, Volume 4**). When the first-type (or Dirichlet) condition is used, h is specified at that location. For the second-type (or Neumann) condition, the normal velocity has to be specified. The often-used no-flow condition is a special case (zero velocity). Third type (or Cauchy) boundary conditions can be used, when a relation between flux and head has to be considered, for example, when an aquifer is connected to a surface water body.

Hydraulic conductivity is an input parameter for most models. In the case of the 2D flow in a confined aquifer, transmissivity (integration of the conductivity over the third spatial direction) is required instead. Unsteady models require specific storativity for unconfined aquifers, and storativity as product of specific storativity and layer thickness for confined aquifers (*see* **Chapter 149, Hydrodynamics of Groundwater, Volume 4**). Porosity is needed when real interstitial velocities have to be determined, for example, when a transport model is to be set up in addition or when travel times along flowpaths are required.

Transport (Mass and Heat)

Transport models are derived from the transport equation, which is the mass balance equation in terms of the concentration of a certain substance in case of solute transport (*see* **Chapter 152, Modeling Solute Transport Phenomena, Volume 4**) through the aquifer system. In case of heat transport, it is a differential equation for temperature as the dependent prime variable (Holzbecher, 1998).

The generic form of the solute transport equation in porous media reads

$$\frac{\partial}{\partial t}(\phi c) = -\nabla(\mathbf{v}c - \phi\mathbf{D}\nabla c) + q \tag{3}$$

with porosity ϕ, combined coefficient of diffusion and dispersion tensor $\mathbf{D}$, specific discharge $\mathbf{v}$ and source/sink-term q. Equation (3) is a balance equation for component mass, which is valid for constant density fluids (*see* **Chapter 152, Modeling Solute Transport Phenomena, Volume 4**). The dependent prime variable is the concentration c. The term on the left side of the equation represents storage in general (gains or losses). The term $\mathbf{v}c$ on the right side represents advection and $\phi\mathbf{D}\nabla c$ is the sum of diffusion and dispersion. The last term is for sources and sinks of all types. Within the heat transport equation, the same terms can be found with different parameters:

$$\frac{\partial}{\partial t}T = \nabla D_{\text{therm}}\nabla T - \kappa\mathbf{v}\nabla T \tag{4}$$

with thermal diffusivity D_{therm} the ratio κ between heat capacity of the fluid and the heat capacity of fluid and porous medium. Internal sources and sinks are neglected in equation (4), which results from the energy balance

equation by dividing through the heat capacity of the porous medium. The dependent prime variable is the temperature T (Holzbecher, 1998).

The types of boundary conditions in general are the same as in flow models. In the first type, a concentration or a temperature needs to be specified. The second type accounts in general for advective and dispersive fluxes. Frequently used is the no dispersive flux condition, where it is required that the derivative of the dependent variable normal to the boundary is zero. The latter condition is not only used at impermeable boundaries. For lack of alternative, it is most often also used at outflow edges. Obviously the condition is not correct for fronts crossing the boundary, or vice versa: it is only applicable when during the simulation period the concentration or temperature gradients are small.

Reactive Transport

Taking into account that only few biogeochemical species in the subsurface are independent from their biogeochemical surrounding, models couple physical transport processes and biogeochemical reactions. The basic features concerning the coupling of transport and reactions are outlined in **Chapter 152, Modeling Solute Transport Phenomena, Volume 4**.

When several species, which are linked to each other by reactions, are modeled, the transport equation has to be solved for each of these species. Two types of reactions have to be distinguished: equilibrium or kinetics. If for an application the timescale of interest exceeds the typical reaction time, the reaction is fast and it can be assumed that the equilibrium is reached, when the reaction is reversible. Equilibria are usually described by the *Law of Mass Action*. For the reaction $A + B \leftrightarrow C$ between species A, B, and C, the equilibrium is given by:

$$\frac{a_C}{a_A a_B} = K \tag{5}$$

where a denotes the activity of the species, and K is the reaction specific equilibrium constant. The activity of a species is the product of the concentration and an activity factor γ, which depends on the charge of the species and the ionic strength of the solution (Krauskopf and Bird, 1995).

When for an application the reaction time is smaller than the time of interest, an explicit formula for the development of the reaction rate in time has to be used. Such a reaction is termed *kinetic*. The difference concerning the coupling with transport.

In general for a set of species, the coupled problem for reactive transport is given by:

$$\phi \frac{\partial}{\partial t}\mathbf{c} = \left(\frac{\partial}{\partial z} D \frac{\partial}{\partial z} - v\frac{\partial}{\partial z}\right)\mathbf{c} + \mathbf{S}_{\text{eq}}{}^{T}\mathbf{r}_{\text{eq}} + \mathbf{S}_{\text{kin}}{}^{T}\mathbf{r}_{\text{kin}} \tag{6}$$

(Saaltink *et al.*, 1998), where the vector $\mathbf{c}$ contains all species concentrations. The vectors $\mathbf{r}_{\text{eq}}$ and $\mathbf{r}_{\text{kin}}$ denote the reaction rates of equilibrium and kinetic reactions, and the matrices $\mathbf{S}_{\text{eq}}$ and $\mathbf{S}_{\text{kin}}$ relate reactions (in rows) and species (in columns) for equilibrium and kinetic reactions.

The problem with (6) is that the rates of the equilibrium reactions are not known beforehand. Thus the entire set of equations is manipulated by linear transformations in order to make the term corresponding with equilibrium reactions vanish. Such a transformation is always possible, but not unique. It can be described by the multiplication of the system (6) with another matrix $\mathbf{U}$ from the left (Saaltink *et al.*, 1998), which is equivalent to the transition from species concentrations to total concentrations, also called *components*.

The system for the total concentrations $\mathbf{u} = \mathbf{U}\cdot\mathbf{c}$ is given by

$$\varphi \frac{\partial}{\partial t}\mathbf{u} = \left(\frac{\partial}{\partial z} D \frac{\partial}{\partial z} - v\frac{\partial}{\partial z}\right)\mathbf{u} + \mathbf{U}\mathbf{S}_{\text{kin}}{}^{T}\mathbf{r}_{\text{kin}} \tag{7}$$

In addition the reaction equilibria have to be taken into account, which can be noted as:

$$\mathbf{S}_{\text{eq}}\cdot\mathbf{log(a)} = \mathbf{log(K)} \tag{8}$$

where the vector $\mathbf{a}$ denotes all activities, and the vector $\mathbf{K}$ all equilibrium constants. Altogether a mathematical system results in which transport differential equations (7) are combined with a set of algebraic equations (8), so-called algebraic differential equations.

Sorption

Sorption denotes a variety of phenomena and processes, which concern the interaction between fluid and solid phase. In the most general approach, sorption reactions can be treated within an extended concept of reactive transport as surface reactions (Parkhurst, 1995). More common are simplified approaches. Kinetic laws can be used to describe slow (nonequilibrium) sorption. The simplest approach is to take first-order kinetics, but this may not suffice for reactive transport, where more complex approaches (for example: Monod) may become necessary. The most common situation of fast (equilibrium) sorption is modeled by combining the transport differential equation with a sorption isotherm (linear sorption, Freundlich-, Langmuir) that describes the equilibrium between solid and fluid phase, leading to the concept of retardation. The details are outlined in **Chapter 152, Modeling Solute Transport Phenomena, Volume 4**.

Different codes handle sorption differently. Some codes allow the direct input of retardation factors (FAST), while others require the specification of the isotherm and the corresponding parameters. Some codes require one isotherm

for the entire domain, others allow the sorption characteristic to change with layers, others by cell (MT3D96).

NUMERICAL MODELS

There are different numerical techniques by which computer algorithms are derived from equations that govern the model. In order to obtain a numerical model, the mathematical (e.g. differential) formulation for continuous variables has to be transformed into discrete form. The discrete variables (e.g. hydraulic head in flow models, concentration, or temperature in transport models) of the model are determined at nodes in the model domain, determined by a *grid*.

Finite Differences

The method of Finite Differences (FD) is derived as approximation of the differential equation. Derivatives (differential quotients) are replaced by difference quotients. For first and second order derivatives, the simplest central stencils (CIS = central in space) are given by:

$$\frac{\partial f}{\partial x} \approx \frac{f_{i+1} - f_{i-1}}{2\Delta x} \qquad \frac{\partial^2 f}{\partial x^2} \approx \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} \tag{9}$$

where the f-values denote function values at the grid nodes, that is, f_i is the approximate value of the function at node i, f_{i-1} at the previous node, and f_{i+1} at the following node (see Figure 1). This leads to a system of equations for the unknown values ($f_i, i = 1..N$), where N denotes the total number of nodes. For transport problems, the upwind scheme (BIS = backward in space) is important:

$$\frac{\partial f}{\partial x} \approx \frac{f_i - f_{i-1}}{\Delta x} \tag{10}$$

In two space dimensions (2D), the five-point stencils, describing finite differences, can be visualized as shown in Figure 1. For example, the Laplace-operator $(\partial^2/\partial x^2 + \partial^2/\partial y^2) f$ is represented by a stencil with factor -4 at the center and 1 in the other four nodes.

FD-grids are usually rectangular, and may be irregular, that is, each column, row, or layer may possess individual grid spacing. The values of the dependent variable are calculated at the nodes, while parameters are specified for the spacing between the nodes (node centered grid).

Finite Volumes

The method of Finite Volumes (FV) is derived from a mass or volume balance for all blocks of the model region. As visualized in Figure 2, the load (e.g. volume or mass) balance in block ij is obtained by:

$$\frac{\partial V}{\partial t} = Q_{i-} + Q_{i+} + Q_{j-} + Q_{j+} + Q \tag{11}$$

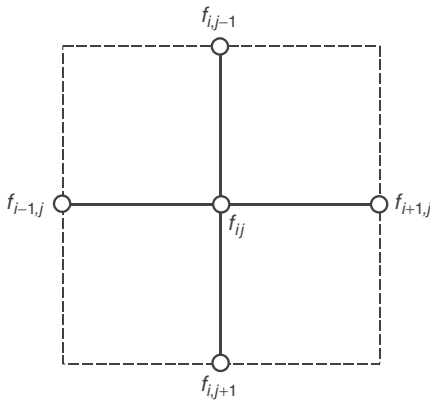

Figure 1 Finite difference stencil for function f in 2D

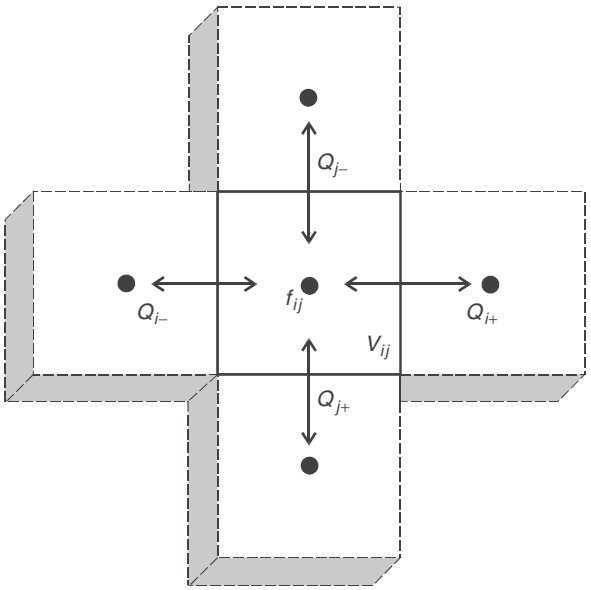

Figure 2 Quadratic finite volume in 2D

where V denotes the volume or mass (see Figure 2) in the block, Q_{i-}, Q_{i+}, Q_{j-}, Q_{j+} the fluxes across the block edges, and Q other sources or sinks for volume or mass.

A set of equations for the values of the unknown variables in the block centers is obtained by expressing the fluxes in terms of that variable. With the help of Darcy's Law, all volume balance equations come into a form that expresses relations between hydraulic heads $f_{ij} = h_{ij}$.

Similarly, Finite Volume grids may be of general form, as they are defined by the budget of fluxes across the boundaries of a block. The dependent variable is calculated at the center of the block (block centered grid). Parameters are specified for blocks, that is, around the block centers. The different form of grids used in FD and FV makes it difficult to compare results obtained by the different methods without interpolation.

Finite Elements

Finite elements grids can be of different shape, but very often they can be recognized by the simple triangular form of the single elements. The triangular shape is convenient to approximate arbitrary shaped regions with small deviations, where rectangular grids often show stairway structures at least at parts of the boundaries.

Using Finite Elements, the solution of the differential equation is found as a combination of shape (or Lagrangian)-functions. These functions are different for different elements. For the most common triangular elements, the prescribed Lagrangian functions are linear within each element (see Figure 3). Thus f is approximated, for example, in Cartesian coordinates, by

$$f(x, y) = a_{\alpha 0} + a_{\alpha 1}x + a_{\alpha 2}y \qquad \text{within element } \alpha \tag{12}$$

All coefficients $a_{\alpha j}$ for all elements are computed as solution of a linear or nonlinear system, which is derived from the so-called weak form of the differential equation (Huyakorn and Pinder, 1983).

Method of Characteristics / Lagrangian Methods

One approach that has been gaining popularity is the mixed Eulerian–Lagrangian method that combines the simplicity of the fixed Eulerian grid with the Lagrangian approach being especially effective in advective dominant regions. Following Neuman and Sorek (1982) for transport problems (e.g. see also Neuman, 1984; Sorek, 1988a,b), for flow problems (Sorek, 1985; Sorek and Braester, 1988) and a modified Eulerian–Lagrangian method for coupled flow and transport model (Bear *et al.*, 1997; Sorek *et al.*, 2000), a technique consisting of the following two steps is used:

1. Formal decomposition of the dependent variable into two parts, one controlled by pure Lagrangian advection and a residual governed by a combination of Euler–Lagrange approaches.

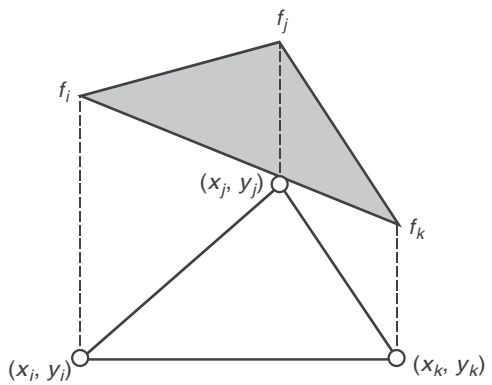

Figure 3 Finite element in 2D

2. Solution of the resulting advection problem by the method of characteristics for forward particle tracking. The residual problem is solved by, for example, an implicit finite element scheme on a fixed grid.

Information is projected back-and-forth between the Eulerian–Lagrangian and the Lagrangian schemes. The major problem of the decomposition strategy is the interpolation (coupling) between the residual and translation solutions as these are obtained at different spatial locations and should be performed in a manner that is mass conservative.

To understand the concept of the decomposition, consider a governing partial differential equation

$$\mathbb{L} f = Q \tag{13}$$

in which $\mathbb{L}$ denotes the differential operator, f denotes the dependent prime variable, and Q denotes the source term. We now decompose (13) into translation, $()_{\mathbb{T}}$, and residual, $()_{\mathbb{R}}$, terms to read

$$(\mathbb{L}_{\mathbb{T}} + \mathbb{L}_{\mathbb{R}})(f_{\mathbb{T}} + f_{\mathbb{R}}) = Q \tag{14}$$

One way to perform the decomposition is to allow

$$\mathbb{L}_{\mathbb{T}} f_{\mathbb{T}} = 0 \tag{15}$$

for which $f_{\mathbb{T}}$ is solved along a characteristic pathline defined by

$$\mathbb{L}_{\mathbb{T}} \mathbf{X} = \mathbf{V} \tag{16}$$

where $\mathbf{X}$ denotes the position spatial vector known only if $\mathbf{V}$ the velocity vector tangent to the characteristic pathline is provided, otherwise iterations are required. By virtue of (14) and (15), the next step is to solve

$$\mathbb{L} f_{\mathbb{R}} = Q - \mathbb{L}_{\mathbb{R}} f_{\mathbb{T}} \tag{17}$$

However, as we may expect the condition

$$\mathbb{L} f_{\mathbb{R}} \gg \mathbb{L}_{\mathbb{R}} f_{\mathbb{T}} \tag{18}$$

and in view of (17) and (18), we obtain

$$\mathbb{L} f_{\mathbb{R}} \cong Q \tag{19}$$

which is similar to a Poisson equation and is not expected to suffer from stability difficulties typical to advection dominant flow regimes. In cases with dominant source terms, we allow $f_{\mathbb{T}}$ to obey

$$\mathbb{L}_{\mathbb{T}} f_{\mathbb{T}} = Q \tag{20}$$

for which by virtue of (17) and (18), we solve $f_{\mathbb{R}}$ by a Laplacian equation in the form

$$\mathbb{L} f_{\mathbb{R}} \cong 0 \tag{21}$$

which is more stable than (19).

Variations of the method of characteristics are implemented in some of the most used transport codes, as MT3D. The user can choose between MOC (method of characteristics), Modified MOC (MMOC), and Hybrid MOC (Zheng and Bennett, 1995). While MOC uses forward tracing along the flowpaths (characteristics), in the MMOC the characteristics are traced backward from the nodes. The hybrid scheme combines both approaches (Neuman, 1984).

Time Stepping

The discretization of time is as a rule implemented as time stepping. The computed values at the previous time level t are used as initial condition for the calculation at the new time level $t + \Delta t$. The time-step Δt may vary. The entire algorithm starts in using the initial condition for the dependent variable of the problem setup, which has to be specified by the user in addition to the differential equation with its boundary conditions.

Simple time-stepping algorithms can be described by the following formula:

$$f(t + \Delta t) = f(t) + \Delta t[\kappa \cdot C(f(t + \Delta t) + (1 - \kappa)C(f(t))] \tag{22}$$

In the notation, the dependency of f on space variables is neglected. The operator C denotes the spatial discretization, which can be achieved using the finite differences (FD), finite volumes (FV), or finite elements (FE) method. κ is a parameter, which either has a fixed value for the implemented method or can be chosen by the modeler. For $\kappa = 1/2$, when both old and new time levels are equally weighted, one obtains the classical *Crank–Nicolson* procedure. For $\kappa = 1$, the totally implicit method results, where the spatial discretization is taken only at the new time level. For $\kappa = 0$, an explicit algorithm results, which is cheaper to solve (as the solution of a linear system is not required). But often the accuracy of the explicit algorithm is poor, or it even does not converge.

Advanced time-stepping algorithms are given by the so-called *Runge-Kutta* methods (Holzbecher, 1996), which can, for example, be found in the FEFLOW code. In mathematical toolboxes, which can also be applied for the solution of differential equations, often an automatic time stepping is implemented, where the accuracy is checked during the simulation and the time step is reduced, if necessary.

Mixing Cells

In contrast to the usual aforementioned procedure, in the mixing cell approach, grid spacing and time stepping are combined. For given velocity v, time-step Δt and grid spacing Δx are related by the formula

$$\Delta t = \frac{\Delta x}{v} \tag{23}$$

The approach is usually applied for constant velocity (1D). The technique can also be used for 1D flow fields with variable velocity, when Δx is varied along the flow paths. But the method is not applicable for general higher dimensional flow fields. Nevertheless, when transverse gradients are small, the mixing cell approach can be used for simulation of 1D transport along a flowpath.

It is obvious from formula (23) that advection with velocity v is modeled without any discretization error. It turns out to be advantageous to combine the mixing cell approach for advection with the conventional FD or FV approach for dispersion. Such operator splitting is implemented in the PHREEQC code (Appelo and Postma, 1993). The PHREEQC code was not originally intended to be combined with a transport model as velocity is not a prescribed input. Instead *lengths* (Δx), *time-step* (Δt), *cells* (number of blocks), and *shifts* (number of time steps) have to be specified as parameters (Parkhurst, 1995).

As for general time stepping, the discretization error, connected with the advection term, turns out to be the most severe in many applications, and the mixing cell approach is very competitive, concerning accuracy with other approaches, when it can be applied.

Reactive Transport

The simultaneous solution of general 3D transport and geochemical speciation is still a challenge for modelers nowadays, as there is a high demand on computer resources' time and space. In the simplest case, the equations for transport can be solved in a first step, delivering total concentrations. The second step is the speciation calculation based on the equilibrium equations (8) depending on the total concentrations available. Speciation calculations require the solution of a highly nonlinear problem, for which the Newton–Raphson algorithm is usually applied (Parkhurst, 1995).

Unfortunately with respect to the outlined solution strategy, the reaction terms for the kinetic reactions mostly depend on species concentrations. Thus the first step cannot be performed without the results of the second step. The problem is handled using three different strategies (Steefel and MacQuarrie, 1996):

- the sequential two step is performed (sequential noniterative approach – SNIA)

- the two-step method is iterated within each time step (sequential iterative approach – SIA)
- discretized differential and algebraic equations are gathered in one system and solved (direct solution approach – DSA)

Surely the SNIA is the cheapest of these methods with respect to the consumption of computer resources, but it delivers incorrect results when there is a strong coupling through the kinetics. Another reason for the popularity of the SNIA is that different available programs for transport and programs for speciation calculations can be loosely coupled, while both other approaches request an internal coupling during each time step.

The coupling of the popular MT3D-MS code for multi-species transport and PHREEQC2 is currently in development (Prommer *et al.*, 2003), for which the SNIA approach is applied.

Accuracy and Stability Criteria

The approximation of a differential equation by a numerical method is not exact, and yields discretization errors. When derivatives are replaced by finite differences, a truncation error results, for which formulae can be derived using the Taylor (or Lagrangian)-series representation. Local errors, due to truncation of derivatives or due to round-off of numbers, may be amplified with the further application of the algorithm. In such a case, the algorithm is called *unstable*.

Concerning accuracy and stability, three dimensionless numbers and three related criteria are relevant:

$$\text{Grid-Péclet number/criterion}: \quad Pe = \frac{v \cdot \Delta x}{D} \leq 2 \qquad (24)$$

$$\text{Courant number/criterion}: \quad Cou = \frac{v \cdot \Delta t}{\Delta x} \leq 1 \qquad (25)$$

$$\text{Neumann number/criterion}: \quad Neu = \frac{D \cdot \Delta t}{\Delta x^2} \leq \frac{1}{2} \qquad (26)$$

The grid-Péclet criterion is relevant for most numerical methods, although an explicit derivation is seldom found, except for finite differences. Figure 4 illustrates typical errors when the criterion is violated. Breakthrough curves from analytical and numerical solutions for 1D front propagation with constant parameters are depicted. The CIS method is burdened by overshooting. In contrast, the BIS method is burdened by enhanced dispersion, the so-called *numerical dispersion*. While the CIS algorithm is stable, when the grid-Péclet criterion is fulfilled the BIS still displays numerical dispersion for $Pe < 2$. An improvement of the BIS method can then be obtained, when the input value for dispersion is reduced by the numerical dispersion value, which according to the truncation error analysis is given by $D_{\text{num}} = v \cdot \Delta x/2$ (Lantz, 1971). The curve,

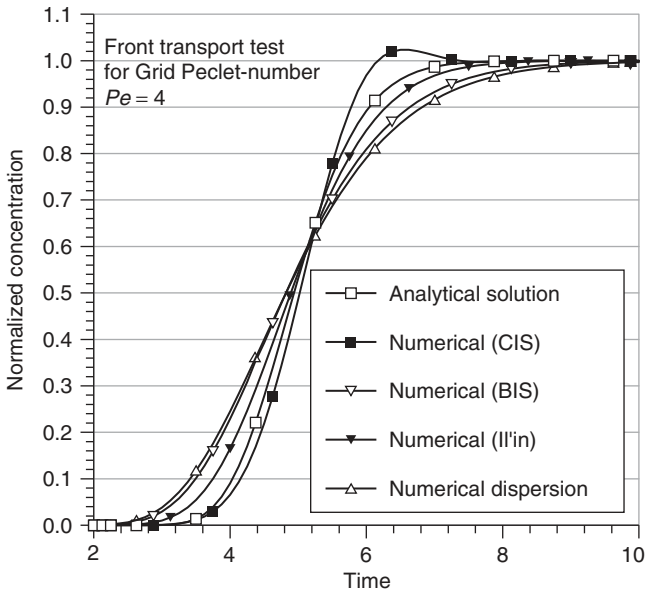

Figure 4 The performance of three numerical schemes for grid-Péclet number $Pe = 4$; all numerical results obtained using automatic time stepping with the Euler method (FIT)

referenced by 'numerical dispersion', depicts the analytical solution for a situation in which D_{num} increases D. The deviation of the BIS-result from that curve shows that the real error of BIS is slightly less than the one predicted by truncation error analysis.

The Il'in method (Il'in, 1969) can be described as a compromise between BIS and CIS. Tests show that it has advantages compared to the other methods only in the vicinity of the critical value for the grid-Péclet numbers, $Pe = 2$. For higher grid-Péclet numbers, the Il'in scheme suffers also from severe numerical dispersion.

The grid-Péclet criterion can be fulfilled when the grid is chosen fine enough, that is, for small Δx. For advection-dominated problems, this may lead to problems when the number of unknown becomes too high.

The Neumann criterion in the given form is valid for explicit algorithms and is less strict or can be completely neglected for implicit algorithms. Both Courant and Neumann criteria can be fulfilled when for a given grid the time step is chosen small enough. This strategy has its limits, as the execution time for a computer run increases with the number of time steps. Especially in 3D modeling, a reduction of the time step may lead to computation periods that are unacceptably high.

The mixing cell method has no discretization error for advection. In order to reduce the discretization error, an operator splitting approach can be applied. It is convenient to use a different time step for diffusion that is a fraction of the advection time step. The PHREEQC code uses a diffusion time-step Δt, which fulfills the condition:

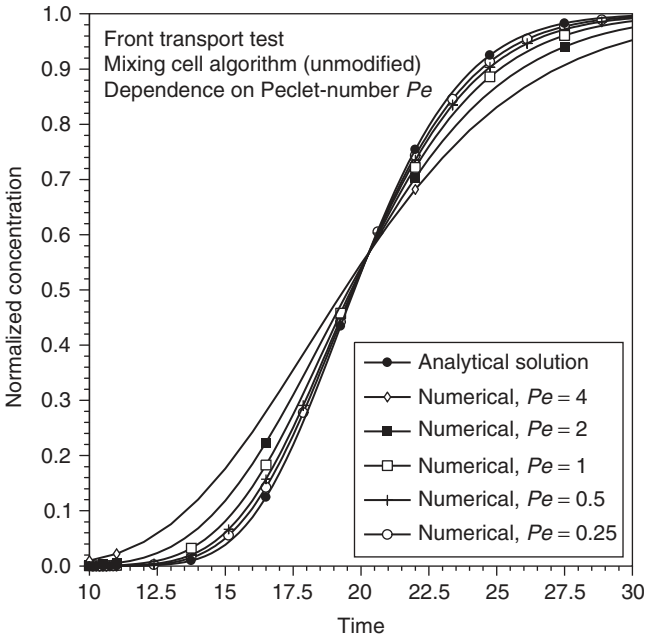

Figure 5 The performance of the mixing cell algorithm for different grid-Péclet numbers; all numerical results obtained using automatic time stepping with the Euler method (FIT)

$Neu \leq \frac{1}{3}$, and which reduces the numerical error very effectively (Appelo and Postma, 1993). Figure 5 shows the performance of a mixing cell method for the 1D front propagation testcase. For given velocity and diffusivity, the grid-spacing and corresponding time step have been reduced, leading to decreasing grid-Péclet numbers *Pe*. The breakthrough curves illustrate how the error is reduced upon reducing of the grid-Péclet number.

INVERSE MODELING

It is the usual task of modeling to determine the dependent variable while the user provides the parameters as input. In groundwater flow models, values for the piezometric head at grid nodes result from using input module for hydraulic conductivities and boundary conditions. But in practical problems, it is often the piezometric heads for which field measurements are available, while hydraulic conductivities are uncertain by at least 1 order of magnitude. In such a situation, the model is used in a different manner. Conductivities as input parameters are varied in order to obtain an optimal match between numerical and measured values of heads.

The illustrated strategy is termed *inverse modeling*, as the role of parameters and variables is exchanged. Inverse modeling is treated in more detail in **Chapter 156, Inverse Methods for Parameter Estimations, Volume 4**. The mathematical procedure is the same for calibration or parameter estimation.

The task of inverse modeling is mostly not implemented in a groundwater modeling code itself. For example, in the ProcessingMODFLOW or FEFLOW, the software package PEST is referred for parameter estimation. PEST or UCODE (alternative in ProcessingMODFLOW) perform all operations concerning the parameter estimation task, that is, new parameter datasets are calculated that are candidates for a model with a better match with the observed data. Then a direct model run is started and finally the results of that run are checked.

Modern GUIs allow the modeler to input measured datasets, to select parameters to be estimated, and to select options for the parameter estimation package. When there are also options for the comparison of measured and modeled data, there is no need to use any other tool for direct and inverse modeling.

PREPROCESSING

Preprocessing is the transformation of data into a form appropriate for the calculating code. Often geologic data are available in a database that cannot be accessed by the modeling program directly. Sometimes special transformation routines are needed to perform this task. Sometimes GIS software for the visualization of geo-data is used.

All operations that are concerned with the selection of the model region, which means especially the location of the boundaries, belong to preprocessing. The modeler can load a background map from some file and construct the model region on the screen using the mouse. Parts of the boundary can be selected for the definition of boundary conditions. Input boxes that require the relevant parameters to be specified open immediately.

Preprocessing includes the choice of model characteristics, such as: confined/unconfined/partially confined, 1D/2D/3D, flow/transport/flow and transport, transient/steady state, solute/heat/solute and heat transport. Nowadays using GUIs most of these options can be selected by mouse-click on a virtual button on the display.

Preprocessing includes the representation of data in a finite grid. Most codes allow external files to be linked with the model. An interpolation routine is then started internally, which delivers parameter values for all elements, blocks, or volumes (see FEFLOW for example). Some codes have options to define parameters as random variable with given statistical properties (see FAST for example).

Finite Element Codes are usually equipped with a *grid generator*. Based on some basic options concerning the approximate number of elements and the type of elements, the GUI usually allows an FE-grid to be created by the push of a button when the model region is defined.

When the modeler is not content with the grid, it can be refined entirely or in parts only. The part to be refined is selected by simple mouse operation. For FD or FV codes that use rectangular grids, such a gridding is usually not necessary.

SOLUTION

The solver is the code that really solves the set of equations for the discrete variable. From GUIs, the solver is started using a submenu entry (ProcessingMODFLOW, Visual MODFLOW, GMS) and for standard situations works with default values for solution parameters. In the case of problems with the solution, that is, no or poor convergence to the numerical solution, or when the results are not accurate enough, the solver or/and parameters should be changed.

Internally mostly a sparse linear or nonlinear system has to be solved. Often there are some options concerning the solution of such a system. Direct solvers can only be recommended for relatively small number of unknowns, that is, for coarse grids. An iterative solver usually is the default. Preconditioned conjugate gradient solvers are mostly used because they are relatively robust, that is, with their standard parameters they work well for a wide range of different problems. For details the reader should consult textbooks like Barrett *et al.* (1994). In some codes, multigrid solvers are included, which are expected to deliver faster convergence.

POSTPROCESSING

After the run of the solver, the discrete representation of the dependent variable is calculated. In order to get specific information from a huge array of numbers, the modeler can perform various postprocessing tasks.

A major postprocessing tool for groundwater flow models is the budget calculator. Fluxes of different type (e.g. inflow, outflow, well recharge, well discharge, groundwater recharge) can be calculated for the entire model region or for user-specified parts of it. For most models such a tool enables quantitative information on the basic fluxes. That is not only a basic information from a completed model, before the completion it is also a basic indicator for errors.

Contour lines (for 2D models) or contour surfaces (for 3D models), which can be plotted for the dependent variable, provide a picture of the variables distribution. Isopleths for hydraulic head, isobars for pressure, isotherms for temperature, isohalines for salt concentration, and streamlines for streamfunction visualize the results of a model run. They not only show where the variable is on an equal level, the minima and maxima also can clearly be identified when the contour levels are chosen appropriate. The different GUIs offer different options for the user to make such a choice.

Contour plots for hydraulic head provide additional information on the flow because flow is normal to the contour lines. When contour levels are equidistant, the density of the isolines gives an impression of the relative amount of the velocity. Regions with dense isolines have higher velocities than regions with less isoline density if the compared regions have the same hydraulic conductivity. Arrow plots give a direct impression of the velocity distribution and flow direction. Long thick arrows represent high velocities in contrast to short thin arrows representing small velocities. In groundwater, the velocities within a model region usually differ by at least 1 order of magnitude, which causes some problem of the arrow plot representation.

Prominent postprocessing tools are codes for particle tracking. For MODFLOW, several of such codes are available (MODPath, PMPath, PATH3D) that visualize streamlines and flowpaths. Some tracking software is designed for steady-state flow fields only, while others can be used for transient flow fields also. For a given flow field, particles can be traced forward or backward in time. There are various different options to set clusters of starting points for the algorithm at inflow or outflow boundaries and/or in the vicinity of wells. Backward particle tracking from positions around a well enables the visualization of the catchments. Similarly the watersheds of groundwater lakes can be determined (Holzbecher, 2001).

Time markers on streamlines or flowpaths indicate isochrones. Most tracking software provides several options to select time markers appropriately, concerning the time levels and the outlook of the marker. In order to calculate traveltimes for a flow field, the porosity has to be specified by the modeler.

Figure 6 depicts a typical output of a groundwater flow model worked out with several postprocessing tools. Added to a background map are well galleries, hydraulic head contour lines, and flowpaths with time markers.

SOFTWARE OVERVIEW

The most prominent code for groundwater modeling is MODFLOW. The most recent version is MODFLOW2000, described by Harbaugh *et al.* (2000). The origins of MODFLOW can be traced back to the beginning of the 1980s. An overview on the history of MODFLOW is given by McDonald and Harbaugh (2003).

Today there are several graphical user interfaces that are 'built around' MODFLOW, like ProcessingMODFLOW, Visual MODFLOW, GMS, and MODFLOW-GUI. These programs assist the user with various pre- and postprocessing tasks; they call MODFLOW for groundwater flow

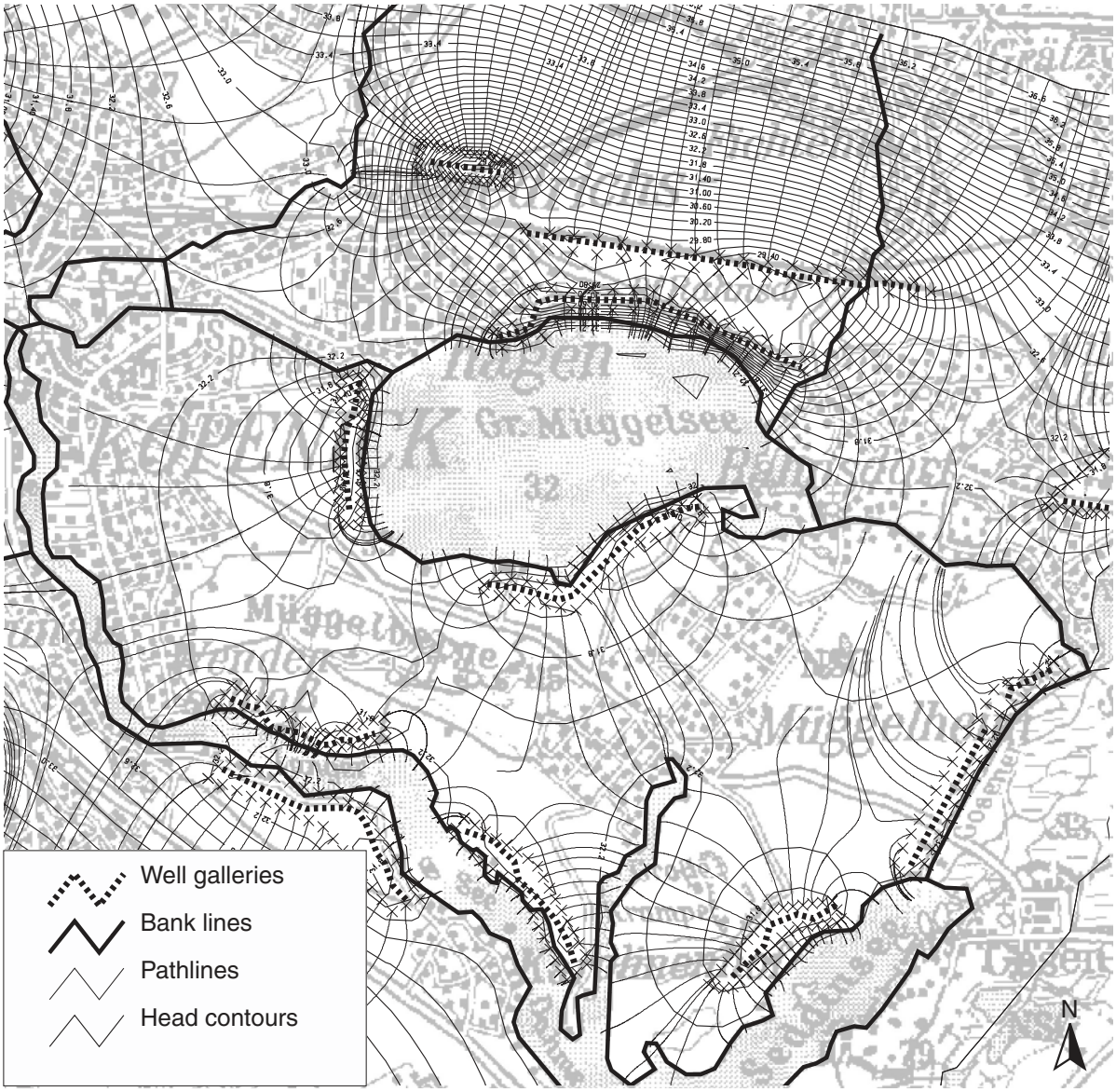

Figure 6 A typical output from a groundwater flow model, depicting background map with well galleries, and the output from several postprocessing tasks: head contours and flowpaths with time markers for illustration of isochrones. The model was setup by M. Zippel using FEFLOW code, postprocessing, and GIS-connection. A color version of this image is available at http://www.mrw.interscience.wiley.com/ehs

models or other codes for transport simulations or other tasks. Moreover, it is possible to use the GUI for direct and inverse modeling.

A surely incomplete list of groundwater modeling codes is given at the end of this chapter. The second column explains if the code solves the resulting systems of equations itself or if it is build around another solver code as a GUI. Some programs are simply tools for pre- and postprocessing. GUIs include tools. The third column indicates the numerical method used for the discretization (FD = Finite Differences, FE = Finite Elements, FV = Finite Volumes). GUIs, which are connected to different solvers, can have more than one entry. For tools, the discretization method does not have to be given as it depends on the solver.

Acknowledgments

The authors are grateful to Matthias Zippel for the presentation of results of the FEFLOW model of well galleries around Lake Müggelsee, which is to be published as part of his doctorate thesis.

SOFTWARE LINKS

Code name	Solver GUI Tool	FD FE FV	Commercial PD	Short Description	Internet Link
CFEST	S	FE		Coupled fluid-energy-solute transport	http://db.nea.fr/abs/html/nesc9537.html
CHAIN-2D	S	FE	PD	2D transport of decay chain	http://www.ussl.ars.usda.gov/MODELS/CHAIN2D.HTM
FAST	S, GUI	FV		3D flow, 2D transport, 2D density driven	http://www.igb-berlin.de/abt1/mitarbeiter/holzbecher/index_e.shtml
FEFLOW	S, GUI	FE	Com.	3D flow, solute, and heat transport	http://www.wasy.de/english/produkte/feflow/index.html
FEMWATER	S, GUI	FE		3D groundwater flow	http://www.scisoftware.com/products/gms_fem/gms_fem.html
GMS	GUI	FE, FV	Com.	for FEMWATER, MODFLOW, MT3D, RT3D, SEEP2D, SEAM3D	http://www.ems-i.com
HST3D	S	FD	PD	3D flow, solute, and heat transport	http://water.usgs.gov/software/hst3d.html
MOC3D	S		PD	3D method of characteristics (flow and transport)	http://water.usgs.gov/software/moc3d.html
MODFLOW	S	FV	PD	3D flow	http://water.usgs.gov/software/modflow.html
MODFLOW-GUI	GUI	FV	PD	for MODFLOW and MOC3D, works under ARGUS ONE only	http://water.usgs.gov/nrp/gwsoftware/mfgui4/modflow-gui.html
MODPATH	Tool	–	PD	Particle tracking for MODFLOW	http://water.usgs.gov/software/modpath.html
Model Viewer	Tool	–	PD	Visualization of 3D model results	http://water.usgs.gov/nrp/gwsoftware/modelviewer/ModelViewer.html
MT3D	S	FV	PD	3D transport	http://hydro.geo.ua.edu/
MT3D-MS	S	FV	PD	3D multiple species transport	http://hydro.geo.ua.edu/
PATH3D	Tool	–		Particle tracking for MODFLOW	http://hydro.geo.ua.edu/mt3d/path3d.htm
PHREEQC	S	Cells[a]	PD	Geochemistry and 1D Transport	http://water.usgs.gov/software/phreeqc.html
PEST	Tool	–	Com.[b]	Parameter Estimation	http://www.parameter-estimation.com
PMWIN	GUI	FV	Com.[b]	for MODFLOW, MOC, MT3D, PEST and UCODE	http://www.scisoftware.com/products/pmwin_details/pmwin_details.html

(*continued*)

Code name	Solver GUI Tool	FD FE FV	Commercial PD	Short Description	Internet Link
PORFLOW		FD	Com.	3D flow, solute, and heat transport	**http://www.acri.fr/English/Products/PORFLOW/porflow.html**
ROCKFLOW		FE		3D flow, solute, and heat transport	**http://www.hydromech.uni-hannover.de/Projekte/Grundwasser/misc/news.html**
RT3D		FV	PD	Reactive transport based on MT3D-MS	**http://bioprocess.pnl.gov/rt3d.htm**
SEAM3D	GUI	FV		Reactive transport based on MT3D-MS	**http://modflow.bossintl.com/html/seam3d.html**
SUTRA	S, GUI	FE	PD	Flow and Transport	**http://water.usgs.gov/software/sutra.html**
SWIFT	S	FV	Com.	3D fluid, solute, and heat transport	**http://www.scisoftware.com/products/swift_overview/swift_overview.html**
TBC	S	FE, FV	PD	Transport, Biochemistry, and Chemistry	**http://www.iwr.uni-heidelberg.de/~Wolfgang Schafer/tbc201.pdf**
UCODE	Tool	–	PD	Parameter Estimation	**http://water.usgs.gov/software/ucode.html**
Visual MODFLOW	GUI	FV	Com.	for MODFLOW, MT3D, RT3D and PEST	**http://www.flowpath.com/software/visualmodflow/visualmodflow.html**

[a]Can be regarded as special type of Finite Volumes, for 1D only.
[b]Limited version is freeware.

FURTHER READING

Holzbecher E. (2002) *Groundwater Modeling – Computer Simulation of Groundwater Flow and Pollution*, FiatLux Publications: Fremont, **http://envirocomp.org**

REFERENCES

Appelo C.A. and Postma D. (1993) *Geochemistry, Groundwater and Pollution*, Balkema: Rotterdam.

Barrett R., Berry M., Chan T.F., Demmel J., Donato J.M., Dongarra J., Eijkhout V., Pozo R., Romine C. and van der Vorst H. (1994) *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM: Also available on the World Wide Web: **http://netlib2.cs.utk.edu/linalg/html_templates/Templates.html**

Bear J., Sorek S. and Borisov V. (1997) On the Eulerian-Lagrangian formulation of balance equations in Porous media. *Numerical Methods for Partial Differential Equations*, **13**(5), 505–530.

Harbaugh A.W., Banta E.R., Hill M.C. and McDonald M.G. (2000) *MODFLOW-2000, the U.S. Geological Survey Modular Ground-water Model – User Guide to Modularization Concepts and the Ground-Water Flow Process*, U.S. Geological Survey: Open-File Report 00-92.

Holzbecher E. (1996) *Modellierung Dynamischer Prozesse in Der Hydrologie: Grundwasser und Ungesättigte Zone*, Springer Publication: Heidelberg.

Holzbecher E. (1998) *Modeling Density-Driven Flow in Porous Media*, Springer Publication: Heidelberg.

Holzbecher E. (2001) The dynamics of subsurface water divides. *Hydrological Processes*, **15**, 2297–2304.

Huyakorn P.S. and Pinder G. (1983) *Computational Methods in Subsurface Flow*, Academic Press: New York.

IAEA International Atomic Energy Agency (1988) *Radioactive Waste Management Glossary, IAEA TECDOC-447, Second Edition*, IAEA: Vienna.

Il'in A.M. (1969) Differencing scheme for a differential equation with a small parameter effecting the highest derivative. *Mathematical Notes of the Academy of Sciences USSR*, **6**, 596–602.

Krauskopf K.B. and Bird D.K. (1995) *Introduction to Geochemistry*, McGraw Hill: New York.

Lantz R.B. (1971) Quantitative evaluation of numerical diffusion (truncation error). *Transactions Society of Petroleum Engineers*, **251**, 315–320.

McDonald M.G. and Harbaugh A.W. (2003) The history of MODFLOW. *Groundwater*, **41**(2), 280–283.

Neuman S.P. (1984) Adaptive Eulerian-Lagrangian finite element method for advection-dispersion. *The International Journal for Numerical Methods in Engineering*, **20**, 321–337.

Neuman S.P. and Sorek S. (1982) Eulerian-Lagrangian methods for advection dispersion, *Proceedings of 4th International Conference on Finite Elements in Water Resources*, Hannover.

Parkhurst D.L. (1995) *PHREEQC: A Computer Program for Speciation, Reaction-Path, Advective Transport, and Inverse Geochemical Calculations*, Water-Resources Investigation Report 95-4227, U.S. Geological Survey, Lakewood.

Prommer H., Barry D.A. and Zheng C. (2003) MODFLOW/MT3DMS-based reactive multicomponent transport modelling. *Groundwater*, **41**(2), 247–257.

Saaltink M.W., Ayora C. and Carrera J. (1998) A mathematical formulation for reactive transport that eliminates mineral concentrations. *Water Resources Research*, **34**(7), 1649–1656.

Silling S.A. (1983) *Final Technical Position on Documentation of Computer Codes for High-Level Radioactive Waste Management*, NUREG-0856.

Sorek S. (1985) Eulerian-Lagrangian formulation for flow in soils: theory. *Advances in Water Resources*, **8**, 118–120.

Sorek S. (1988a) Two-dimensional adaptive Eulerian-Lagrangian method for mass transport with spatial velocity distribution. *Transport in Porous Media*, **3**, 473–489.

Sorek S. (1988b) Eulerian-Lagrangian method for solving transport in aquifers. *Advances in Water Resources*, **11**(2), 67–73.

Sorek S. and Braester C. (1988) Eulerian-Lagrangian formulation of the equations for groundwater denitrification using bacterial activity. *Advances in Water Resources*, **11**(4), 162–169.

Sorek S., Borisov V. and Yakirevich A. (2000) Numerical modeling of coupled hydrological phenomena using the modified Eulerian-Lagrangian method. In *Theory, Modeling and Field Investigation in Hydrogeology: A Special Volume in Honor of Shlomo P. Neuman's 60th Birthday*, Zhang D. and Winter C.L. (Eds.), Geological Society of America: Special Paper 348, pp. 151–160.

Steefel C.I. and MacQuarrie K.T.B. (1996) Approaches to modeling of reactive transport in porous media. In *Reactive Transport in Porous Media*, Vol. 34, Lichtner P.C., Steefel C.I. and Oelkers E.H. (Eds.), Mineralogical Society of America: pp. 83–129.

Zheng C. and Bennett G.D. (1995) *Applied Contaminant Transport Modeling*, Van Norstrand Reinhold: New York.