A general technique for assessing the numerical accuracy of solute transport models

Feng Ruan
Advanced Technology, Schlumberger, GeoQuest, Houston, Texas

Dennis McLaughlin
Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge

Shuguang Li
Department of Civil Engineering, Portland State University, Portland, Oregon

Abstract. This note describes a procedure for evaluating the accuracy of numerical solute transport models in situations where exact closed-form solutions are difficult or even impossible to obtain. The procedure attempts to match a specified closed-form “test solution” by adding forcing terms to the original equation, which is solved numerically. The quality of the match provides valuable information about the performance of the numerical algorithm. We illustrate this “prescribed forcing method” with an example which simulates solute transport in a heterogeneous velocity field. The numerical solver considered in the example is based on the Eulerian-Lagrangian method with linear velocity and concentration interpolation. Two test solutions of different degrees of difficulty are considered. Differences between the exact and numerical test solutions for the example clearly reveal the influence of grid resolution on model accuracy. The example demonstrates that the prescribed forcing method can be used to assess numerical accuracy in practical situations where model inputs are highly variable and the true solution is unknown.

1. Introduction

Numerical solutions to partial differential equations are typically checked by comparing them to closed-form (or analytical) solutions. If the closed-form solution is evaluated correctly, any discrepancies between the numerical and closed-form results can be attributed to errors in the numerical algorithm. Although such checks are useful, they test the numerical algorithm only under the simplified conditions needed to obtain a closed-form solution. In solute transport applications the closed-form solutions used for testing purposes typically apply for spatially uniform and time-invariant velocity fields, simple source geometries, and semi-infinite or infinite solution domains [Javandel et al., 1984]. Numerical algorithms which perform adequately under such conditions may not do as well when the velocity field is highly variable, the source geometry is irregular, or the boundaries are complex.

In order to properly evaluate the performance of numerical algorithms, we need to go beyond comparisons with simple closed-form solutions. A number of different options are routinely used in solute transport applications. One is to check the ability of the numerical algorithm to conserve mass in the presence of a nonuniform velocity field, irregular boundaries, etc. While mass conservation is an important aspect of algorithm performance, it does not insure that the numerical solution actually satisfies the original transport equation. Mass conservative algorithms can, for example, still introduce significant numerical dispersion. Additional insight can sometimes be obtained by comparing alternative numerical solutions to a “benchmark problem” which cannot be solved in closed form [Baptista et al., 1995]. Although this provides information about the relative performance of the selected algorithms, it does not test absolute accuracy. Another option is to examine the behavior of the numerical solution as its spatial computational grid and time step are progressively refined. If the numerical algorithm is working properly, the resulting set of solutions should gradually converge to a stable solution as the grid spacing and time step are progressively reduced. This approach is useful, but it can be computationally demanding.

Some of the deficiencies of traditional methods for testing numerical algorithms are addressed by the “prescribed forcing method,” originally proposed by Dee [1991]. This procedure attempts to match a specified closed-form “test solution” by adding forcing terms to the original equation, which is solved numerically. The quality of the match provides valuable information about the performance of the numerical algorithm. The prescribed forcing method has been used primarily to evaluate multidimensional hydrodynamic flow solvers [Dee, 1991; Dietrich et al., 1990; Dee et al., 1992; Roach, 1994, 1995]. In this note we generalize the prescribed forcing concept to include a wider range of test solutions. We also use the method to investigate the accuracy of a popular solute transport solver.

The particular problem of interest here is one where the velocity field is time invariant but highly variable in space (i.e., the velocity is a stationary random field with a correlation scale much smaller than the size of the solution domain). This is the situation often considered in theoretical, experimental, and numerical studies of macrodispersion [Gelhar, 1993]. Since
heterogeneous velocity transport problems are difficult to solve, some care must be taken when using numerical models to verify theoretical predictions or analyze field experiments. Inaccurate solvers which are prone to numerical dispersion or oscillation may yield misleading macrodispersivity estimates or may be reliable only when used with unrealistically large local dispersivity values. The prescribed forcing method provides a useful way to check the adequacy of the grid resolution and time step used to solve transport problems with heterogeneous velocity fields. More generally, the method may be used to identify coding errors, compare alternative solution techniques, and provide quantitative measures for a wide range of applications [Ruan, 1997].

2. Method

The prescribed forcing method can be applied to many types of partial differential equations [Dee, 1991; Dietrich et al., 1990; Dee et al., 1992]. For specificity, we focus here on the following multidimensional solute transport (advection-dispersion) equation, defined over the spatial domain \( \Omega \) and time period \( J \):

\[
\frac{\partial c}{\partial t} + \nabla \cdot ( vc ) - \nabla \cdot ( D \nabla c ) = s \quad ( x, t ) \in \Omega \times J
\]

\[c( x, 0 ) = c_0( x ) \quad x \in \Omega \quad (1)
\]

\[ac + \beta \nabla c \cdot n = \gamma( x, t ) \quad ( x, t ) \in \Gamma \times J,
\]

where \( c( x, t ) \) is the solute concentration, \( v(x) \) is the time-invariant velocity, \( D \) is a constant scalar dispersion coefficient, \( s \) is an interior source/sink, \( \Gamma \) is the boundary with outward unit normal \( n \), \( \alpha, \beta, \) and \( \gamma( x, t ) \) are coefficients used to determine the type of boundary condition applied, and \( c_0( x ) \) is the initial condition.

The generalized prescribed forcing method used here modifies the original equation by adding the specified (or prescribed) forcing terms \( s'( x, t ) \), \( c_0'( x ) \), and \( \gamma'( x, t ) \) to the source, initial condition, and boundary condition. The resulting modified equation has a solution \( c^*( x, t ) \) (called the test solution), which will be equal to \( c( x, t ) \) when the prescribed forcing terms are all zero but will generally be different from \( c( x, t ) \) when any of these terms is nonzero. The modified equation has the following form:

\[
\frac{\partial c^*}{\partial t} + \nabla \cdot ( vc^* ) - \nabla \cdot ( D \nabla c^* ) = s + s' \quad ( x, t ) \in \Omega \times J
\]

\[c^*( x, 0 ) = c_0( x ) \quad x \in \Omega \quad (2)
\]

\[ac^* + \beta \nabla c^* \cdot n = \gamma( x, t ) + \gamma'( x, t ) \quad ( x, t ) \in \Gamma \times J.
\]

In principle, a specified differentiable test solution \( c^* \) will satisfy (2) if the prescribed forcing terms are computed as follows:

\[s'( x, t ) = \frac{\partial c^*}{\partial t} + \nabla \cdot ( vc^* ) - \nabla \cdot ( D \nabla c^* ) - s \quad ( x, t ) \in \Omega \times J
\]

\[c_0'( x ) = c^*( x, 0 ) - c_0( x ) \quad x \in \Omega \quad (3)
\]

\[\gamma'( x, t ) = ac^* + \beta \nabla c^* \cdot n - \gamma( x, t ) \quad ( x, t ) \in \Gamma \times J.
\]

When the prescribed forcings are selected in this way, the numerical solution to (2), which we represent by \( \tilde{c}( x, t ) \), should be equal to \( c^*( x, t ) \). In practice, these two solutions will be different because the numerical algorithm is not perfect. The differences between \( \tilde{c}( x, t ) \) and \( c^*( x, t ) \) provide valuable quantitative information about the performance of the algorithm and about possible sources of numerical error. For example, we can measure the aggregate “goodness of fit” for a two-dimensional problem with the following spatially aggregated root-mean-square error measure:

\[J(t) = \frac{\Delta x_1 \Delta x_2}{M(t)} \sum_{k=1}^{\infty} \left| c^*( x_k, t ) - \tilde{c}( x_k, t ) \right|^2, \quad (4)
\]

where \( M(t) \) is the plume mass (practically constant in test problems considered here), the subscript \( k \) refers to a particular node of the numerical grid, \( N \) is the total node number of grid nodes, and \( \Delta x_1 \) and \( \Delta x_2 \) are the grid cell dimensions (node spacings) in the \( x_1 \) and \( x_2 \) directions, respectively. It is easy to think of other quantitative measures of difference between \( c^* \) and \( \tilde{c} \), such as the cumulative distribution of errors (ranked from smallest to largest). The root-mean-square measure provides a good overall indication of the quality of the numerical solution. It seems reasonable to expect that some test solutions will be better suited to the purpose of identifying numerical errors than others. We can state three general criteria which will be satisfied by a “good” test solution:

1. The differentiable test solution should be expressed in a closed form which can be evaluated with essentially no error. Moreover, it should be possible to evaluate the prescribed forcings analytically. This eliminates the possibility that discrepancies between the numerical and test solutions are due to errors in evaluation of the test solution or in computation of the prescribed forcing terms.

2. The prescribed forcing terms should not be much larger than the largest term in the original differential equation (as measured by an appropriate norm). This insures that the test checks the algorithm’s ability to solve the original problem, rather than a much different problem which is dominated by the prescribed forcings. This criterion is satisfied in the solute transport problem considered here if the shape and trajectory of the test solution plume (as measured by first and second spatial moments) are similar to those of the original solution.

3. The test solution should challenge the numerical solver. For example, the space and time gradients should be sufficiently large to insure that the fit between test and numerical solution becomes noticeably worse when the space and time steps are increased beyond a certain point. This makes it easier to discriminate between alternative numerical solution algorithms, grid resolutions, etc.

In summary, the art of using the prescribed forcing method is to find a test solution which can be expressed in closed form but which is sufficiently complex to reveal the deficiencies of the numerical algorithm.

3. Examples

In this note we evaluate a classical Eulerian-Lagrangian transport solver for a two-dimensional problem with a random velocity field, a simple localized instantaneous source (or initial condition), and boundaries sufficiently far removed from the source for \( \Omega \) to be considered an infinite domain. The Eulerian-Lagrangian solution algorithm is based on operator-splitting concepts and relies on linear interpolation of the ve-
locity and concentration fields in the advective step and a finite difference approximation in the dispersive step [Cheng et al., 1984; Ruan and McLaughlin, 1999; Wheeler and Dawson, 1988]. We focus on the effect of spatial grid resolution on the numerical accuracy of the algorithm. The computational grid consists of square cells ($\Delta x = \Delta x = \Delta x$) and extends over a rectangular region of 480 by 240 m. (see Plate 1). The solution is computed over a 2400 day period, with 120 time steps of 20.0 days each. The

Plate 1. Solute contours obtained (first or left column) for the original problem, (second column) for the Gaussian test solution, and (third column) for the sinusoidal Gaussian test solution. The four rows, arranged from top to bottom, are obtained with grid spacings of $\Delta x = 2.0$ m, 1.0 m, 0.5 m, and 0.25 m, respectively. The initial center of mass location is at $x_0 = 48.0$ m and $y_0 = 60.0$ m. The contour axes units are meters.
lowest resolution grid we consider has 241 by 121 nodes and a grid cell size $\Delta x$ of 2 m. The more refined grids have progressively smaller grid spacings and larger numbers of nodes.

The random velocity field for the coarsest grid problem is obtained from a spectrally based random field generator [Ruan and McLaughlin, 1998]. Its spatial structure is defined by velocity spectra derived from a linearized analysis of the groundwater flow equation and Darcy’s law and an assumed log hydraulic conductivity spectral density function. The mean velocity $\bar{v}$ is aligned with the $x_1$ direction and has a magnitude of 0.042 m/d. The log hydraulic conductivity has a Gaussian spectral density with a variance of 1.0. The isotropic correlation length is 4$\Delta x$ or 8 m. The spatial discretization process can be viewed in the spectral domain as a low-pass filter which tends to attenuate velocity fluctuations at wave numbers above a cutoff of approximately 2/$\Delta x$ [Papoulis, 1984]. Grids with different node spacings will have different cutoffs, and the resulting filtered velocity fields will have different spectral properties. In order to eliminate this effect, the velocity fields for the higher-resolution grids are obtained by interpolating the coarsest resolution field with a cubic spline. This insures that the velocity spectrum cutoff will always be approximately equal to the value associated with the coarsest grid.

The initial condition for the example problem is

$$c_0(x) = c_m \exp \left[ -\frac{(x_1 - x_{10})^2}{2\sigma_{10}^2} - \frac{(x_2 - x_{20})^2}{2\sigma_{20}^2} \right].$$ (5)

This defines a spatially distributed instantaneous source centered at $[x_{10}, x_{20}]$ with characteristic length scales $\sigma_{10}$ and $\sigma_{20}$. For our example, $x_{10} = 48.0$ m, $x_{20} = 60.0$ m, $\sigma_{10} = 12$ m, $\sigma_{20} = 4$ m, and the initial peak concentration at $[x_{10}, x_{20}]$ is $c_m = 1$ ppm. The isotropic scalar dispersion coefficient is 0.001 m$^2$/d.

The example problem defined above must be solved numerically because the velocity field is random. We use two different test solutions to evaluate the performance of the Eulerian-Lagrangian numerical algorithm. The first of these is the test solutions to evaluate the performance of the Eulerian test and random velocity plumes generally increase as the velocity variance increases.

The prescribed forcing terms needed to generate the Gaussian test solution are obtained by substituting (6) into (3). The result is

$$s'_{c}(x, t) = (v_1 - \bar{v}) \frac{\partial c^*}{\partial x_1} + v_2 \frac{\partial c^*}{\partial x_2} \quad (x, t) \in \Omega \times J$$

$$c_{0c}(x) = 0 \quad x \in \Omega \quad \gamma'_{c}(x, t) = 0 \quad (x, t) \in \Gamma \times J.$$ (7)

The prescribed forcing terms for the initial and boundary condition are zero because the selected test solution satisfies the initial and boundary conditions of the original random velocity problem. In general, these terms will not be zero. Note that the problem which results when (7) is substituted into (2) has both spatially variable velocity and spatially variable prescribed forcing terms. If the numerical algorithm is accurate, the effects of these terms must cancel in such a way as to produce the desired Gaussian test solution.

The second test solution we consider is not the solution to a transport equation but is obtained by modifying the Gaussian test solution in a way which increases the difficulty of the numerical problem. In particular, we multiply the Gaussian test solution by a harmonic term which induces more spatial variability in both the test solution and the prescribed forcing terms:

$$c^*_{sg}(x, t) = c_{0c}(x, t) \frac{\sin(2\pi \phi(x_1 - x_{10} - \bar{v}t) + B)}{\sin(2\pi \phi(x_1 - x_{10} - \bar{v}t) + B)} c^*_{c}(x, t, t)$$

$$= A \sin \left[ 2\pi \phi(x_1 - x_{10} - \bar{v}t) + B \right] c^*_{c}(x, t, t),$$ (8)

where $c^*_{c}(x, t, t) = A \sin \left[ 2\pi \phi(x_1 - x_{10} - \bar{v}t) + B \right]$ and the spatial frequency $\phi = k/\sigma_{10}$ is a function of the initial plume dimension $\sigma_{10}$. The coefficients $k, A$, and $B$ are the fluctuation length, amplitude, and phase coefficients, respectively. We call $c^*_{sg}$ the “sinusoidal Gaussian test solution.”

The prescribed forcing terms needed to generate the sinusoidal Gaussian test solution are obtained by substituting (8) into (3). The result is

$$s'_{sg}(x, t) = c^*_{H} \left[ (v_1 - \bar{v}) \frac{\partial c^*}{\partial x_1} + v_2 \frac{\partial c^*}{\partial x_2} \right]$$

$$+ c^*_{H} \left[ \frac{\partial c^*}{\partial t} + \bar{v} \frac{\partial c^*}{\partial x_1} - \frac{\partial}{\partial x_1} \frac{\partial c^*}{\partial x_1} - 2D \frac{\partial c^*}{\partial x_1} \right]$$

$$\left[ \frac{\partial c^*}{\partial x_1} \right] (x, t) \in \Omega \times J$$

$$c_{sg}(x) = c_{0c}(x) [A \sin \left[ 2\pi \phi(x_1 - x_{10}) + B \right] - 1] \quad x \in \Omega \quad \gamma'_{sg}(x, t) = 0 \quad (x, t) \in \Gamma \times J.$$ (9)

In this case the prescribed initial condition forcing term is no longer zero since the sinusoidal test solution does not satisfy the initial condition of the original problem. However, the test...
solution still satisfies the original infinite domain boundary condition, so the prescribed boundary condition forcing term is zero. The various spatial and temporal derivatives appearing in (9) can be computed in closed form from the harmonic and Gaussian portions of the sinusoidal Gaussian test solution given in (8).

The accuracy of the Eulerian-Lagrangian algorithm can be examined by solving (2) numerically, with the prescribed forcings \( s'(x, t), c_0(x), \) and \( \gamma(x, t) \) set either to the Gaussian or sinusoidal Gaussian expressions given in (7) or (9). We can then compare the resulting numerical solutions \( \hat{c}_G(x, t) \) or \( \hat{c}_{SG}(x, t) \) with the corresponding closed-form test solutions \( c^*_G(x, t) \) or \( c^*_SG(x, t) \).

In order to show how the prescribed forcing method reveals numerical errors, we have computed numerical approximations to each of our two test solutions on a set of computational grids of increasing resolution. The coarsest grid (with a grid spacing of 2 m and 241 by 121 nodes) is selected as the reference. The finer-resolution grids have grid spacings of 1.0, 0.5, and 0.25 m, respectively. The number of nodes for each of these cases is increased as required to keep the domain size constant at 480 m by 240 m.

Concentration contour plots provide a convenient way to examine the performance of the numerical algorithm at any given time. Plate 1 shows the concentration contours at the final simulation time (2400 days) obtained from the numerical solutions to the (1) original problem (first column); (2) the problem which should reproduce the Gaussian test solution (second column); and (3) the problem which should reproduce the sinusoidal Gaussian test solution (third column). Each row corresponds to one of the grid resolutions, with the coarsest grid in the first row (from the top) and the finest grid in the fourth row. The exact (closed form) Gaussian and sinusoidal test solutions are provided for reference in the fifth row.

It is revealing to note the changes in the plume shape and peak concentration for the original problem (first column) as the grid is refined. The fine-grid solver provides a complex plume with quite a bit of fine structure but no oscillations (oscillatory solvers typically generate negative concentrations on the fringes of the plume). The coarsest-grid solution is much smoother, with lower peak values, than the fine-grid solution, suggesting the presence of numerical dispersion. This is typical of coarse-grid Eulerian-Lagrangian solvers which use linear interpolators [Ruan and McLaughlin, 1999].

The effect of grid resolution on the accuracy of the numerical solution is confirmed by plots of the numerical approximations to the two test solutions (second and third columns of Plate 1). The approximate solutions differ significantly from the exact test solutions for coarse resolutions but converge to the exact solutions as the grid resolution increases. The sinusoidal Gaussian test case seems to be more sensitive to grid spacing since it converges more slowly. Even at the finest resolution, there are visible differences between the numerical and exact solutions. This is to be expected since the exact sinusoidal Gaussian test solution is more spatially variable and more difficult to reproduce numerically than the exact Gaussian test solution. Of course, the sinusoidal Gaussian solution can be made still more difficult if the fluctuation length is decreased or the fluctuation amplitude is increased.

Plate 1 nicely illustrates the practical advantage of using the prescribed forcing method as a substitute for a conventional variable-grid convergence analysis. The conventional convergence analysis reveals the presence of numerical dispersion at coarser resolutions only when solutions are computed over grids that are significantly finer than the reference (coarsest) grid. By contrast, the prescribed forcing method reveals numerical problems with only one simulation at the coarsest resolution.

Additional insight can be obtained from Figure 1, which shows the log root-mean-square error measure \( \log_{10} J(t) \) at several different times plotted versus log grid spacing \( \log_{10} \Delta x \). The prescribed forcing error is quite stable over time, indicating that differences between the numerical and exact test solutions persist as the plume moves through the simulation domain. It is clear from Plate 1 that the log error measure decreases nearly linearly with decreasing \( \log_{10} \Delta x \). Note that the error measure is significantly larger for the more difficult sinusoidal Gaussian case.

4. Discussion

The prescribed forcing method described in this note is clearly able to reveal inaccuracies in numerical solvers. It is easy to implement, and it has the distinct advantage of being able to test performance when coefficients, boundary conditions, and forcing terms vary over space and comparisons with analytical solutions are not feasible. In its present form the method is best suited to identifying the presence rather than the source of numerical errors. However, it is often possible to diagnose problems revealed by the method in an indirect fashion.

For example, the linear Eulerian-Lagrangian approach used in our test problem is known to be numerically dispersive. Consequently, when the prescribed forcing method indicates that the numerical and exact test solutions differ, it is reasonable to suspect that numerical dispersion is the culprit. An appropriate corrective measure in this case is to reduce the grid spacing. The prescribed forcing method indicates that this does, in fact, improve solution accuracy. Similarly, if the prescribed forcing method gives poor results when applied to a numerical algorithm that is prone to oscillation (such as a
quadratic Eulerian-Lagrangian approach), we can suspect that oscillation is a problem. It would be useful if we could make this diagnostic process more precise, perhaps by a more detailed analysis of the errors between the exact and approximate test solutions. However, this is difficult to do with any generality. Part of the problem is that the spatial pattern of the errors is affected by the structure of the heterogeneous velocity field. This confounding influence needs to be taken into account in a detailed spatial error analysis.

It is reasonable to ask if we could extract more information from the prescribed forcing method if we looked at a wider range of test solutions. Certainly, it is possible that more can be done. In fact, this would be a useful topic for further research. However, we feel that the information provided by the present version of the method is sufficient to assess performance in practical situations where model inputs are highly variable and the true solution is unknown. If the method reveals problems, changes can be made, and the prescribed forcing test can be repeated until the problems are resolved. In the case of solute transport we feel that the prescribed forcing method provides a more realistic performance test than other criteria (such as a check of the grid Peclet number) which are based on constant velocity assumptions or other simplifications.

In closing, it is worth noting that the prescribed forcing method is quite general. It is not confined to closed-form test solutions or even to test solutions that satisfy particular differential equations (such as the transport equation). The advantage of using a closed-form solution is that numerical errors are contributed only by the algorithm used to solve the differential equation of interest (criterion 1). The advantage of using a test solution obtained from a differential equation similar in structure to the original equation is that it is generally easier to insure that the prescribed forcing terms do not dominate the numerical problem (criterion 2). However, these are not strict requirements, and we may choose to use any reasonable test solution, so long as the prescribed forcing terms can be accurately computed and are small enough to insure that the testing procedure checks the algorithm’s ability to solve the original problem. We encourage modelers and numerical analysts to try the prescribed forcing method and to document their experiences so that the capabilities and limitations of the method can be better defined.

Acknowledgment. The authors acknowledge the helpful insights and comments provided by Carsten Schwarz and Wolfgang Kinzelbach.

References


S.-G. Li, Department of Civil Engineering, Portland State University, Portland, OR 97207. (shuanglei@cas.pdx.edu)

D. McLaughlin, Ralph M. Parson’s Laboratory, Room 48-209, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139. (denism@mit.edu)

F. Ruan, Advanced Technology, Schlumberger, GeoQuest 5599 San Felipe, Suite 1700, Houston, TX 77056. (fruan@houston.geoquest.slb.com)

(Received October 13, 1998; revised July 6, 1999; accepted July 30, 1999.)