Multiscale Interpolation, Backward in Time Error Analysis for Data-Driven Contaminant Simulation

Craig C. Douglas[1,2], Yalchin Efendiev[3], Richard Ewing[3], Victor Ginting[3], Raytcho Lazarov[3], Martin J. Cole[4], Greg Jones[4], and Chris R. Johnson[4]

[1] University of Kentucky, Department of Computer Science, 325 McVey Hall, Lexington, KY 40506-0045, USA
{craig.douglas, ceshan0}@uky.edu
[2] Yale University, Department of Computer Science, P.O. Box 208285, New Haven, CT 06520-8285, USA
douglas-craig@cs.yale.edu
[3] Texas A&M University, ISC, College Station, TX, USA
{efendiev, ginting, lazarov}@math.tamu.edu, richard-ewing@tamu.edu
[4] Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT, USA
{gjones, mjc}@sci.utah.edu, crj@cs.utah.edu

Abstract. We describe, devise, and augment dynamic data-driven application simulations (DDDAS). DDDAS offers interesting computational and mathematically unsolved problems. In this paper, we discuss how to update the solution as well as input parameters involved in the simulation based on local measurements. The updates are performed in time. We test our method on various synthetic examples.

1 Introduction

In recent years, immense computing power has become available at the national and international supercomputer centers and local clusters of fast PCs. We also have had a proliferation of data acquisition and generation through the deployment of sophisticated new generations of sensors. The lack of coordination between current computational capacity and sensor technology impairs our ability to effectively utilize the flood of information available. This is a substantial barrier to achieving the potential benefit computational science can deliver to many application areas including contaminant tracking, wildfire modeling, transportation optimization, and many other fields.

Sensors and data generating devices may take many forms including other running computational simulations. The intent of this paper is to address several DDDAS enabling technologies in the context of a specific application area in order to provide techniques and tools to effectively demonstrate the potential

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3515, pp. 640–647, 2005.

of dynamic data driven simulations for other areas. Our primary application is contaminant tracking, which in groundwater reservoirs is modeled by strongly coupled systems of convection-reaction-diffusion equations. The solution process of such systems becomes more complicated when modeling remediation and clean-up technologies since they exhibit strong nonlinearities and local behavior. Many of these applications are essentially computer models that solve nonlinear, unsteady, coupled, partial differential equations. All require consistent initial conditions, adequate forcing fields, and boundary conditions to advance the solution in time. Collectively these fields represent the input data necessary to run the models. The input data can originate from observations, e.g., sensor based telemetry, can be internally generated from ensemble type simulations, or can be externally generated (e.g., providing boundary conditions for very high resolution regional models). The skill of these models to adequately represent realistic conditions is intimately tied to the quality, spatial and temporal coverage, and intelligent use of their input data sets. These applications in turn generate large amounts of output data that must be either analyzed or passed on to other more specialized subcomponents.

The update is performed based on the sensor measurements, which streamed from few spatial locations. As data is injected, we propose to update (1) the solution (2) the initial condition. We have also considered the update of the media properties, but this will not be discussed in this paper. Because of the heterogeneities of the porous media, we employ multiscale interpolation technique for updating the solution. This is done in the context of general nonlinear parabolic operators that include subsurface processes. The main idea of this interpolation is that we do not alter the heterogeneities of the random field that drives the contaminant. Rather based on the sensor data we rescale the solution in a manner that it preserves the heterogeneities. This rescaling uses the solution of the local problems. We compare the numerical results for simulations that employ both updating the data at sensor location and the simulations that do not update the locations. The simulation results that do not use the update or use it less frequently produces large errors. This was observed in our numerical studies.

The errors in the simulations will persist if one does not change the input parameters. As new data are obtained from sensors measurements, the initial data needs to be updated. This update reduces the computational errors associated with incorrect initial data and improves the predictions. In this paper, we consider linear subsurface flows involving convection and diffusion. Initial data is sought in a finite dimensional space. Using the first set of measurements, the approximation of the initial data is recovered. As new data are incorporated into the simulator, we update the initial data using an objective function. We note that the formulated problem is ill-posed. Two facts can be attributed to this ill-posedness. First, the data gathered from the sensor measurements always contain some defects that come from factors such as human errors and inherent factory errors of the sensors. Secondly, the number of sensors that can be installed are limited, and in general are much fewer than the finite dimensional space describing the initial data. For the latter, we can regularize the problem by

using the prior information about the initial data. This prior information is the updated initial data. The penalization constants depend on time of update and can be associated with the relative difference between simulated and measured values. Numerical examples are presented in the paper showing the improvement of the predictions as new data is taken into account.

2 Backward Error Analysis and Initial Data Recovery

As new data is injected from sensor measurements, "the initial data" can be updated. Here, by initial data we mean contaminant distribution at some early time. Due to poor knowledge of the initial location of contaminant, this type of errors can be dominant in simulations. We consider a linear transport dominated by convection and diffusion

$$\frac{\partial C}{\partial t} + v \cdot \nabla C - \nabla \cdot (D \nabla C) = 0 \text{ in } \Omega, \tag{1}$$

where by Darcy's Law, we have $v = -k\nabla p$, with the pressure p satisfies

$$-\nabla \cdot (k \nabla p) = 0 \tag{2}$$

with some prescribed boundary conditions and initial condition/data $C(\mathbf{x}, 0) = C^0(\mathbf{x})$. Here the variable $C(\mathbf{x}, t)$ is designated for a contaminant concentration over the porous medium Ω and at time level t, k is the permeability of the porous medium, and D is the diffusion coefficient.

We seek the initial data in a finite dimensional space. The dimension of this space is an important factor in our simulations. The problem becomes more ill-posed if this dimension increases. In general, one can reduce this dimension using multiscale representation of the initial data (note that initial data represents the solution at early times, not necessarily at time zero). As new data are incorporated into the simulator, we update the initial data using an objective function. Before we formulate the objective function, we note that the formulated problem is ill-posed. Two facts can be attributed to this ill-posedness. First, the data gathered from the sensor measurements always contain some defects that come from factors such as human errors and inherent factory errors of the sensors. Secondly, the number of sensors that can be installed are limited, and in general are much fewer than the finite dimensional space describing the initial data. For the latter, we can regularize the problem by using the prior information about the initial data. This prior information is the updated initial data. The penalization constants depend on time of update and can be associated with the relative difference between simulated and measured values.

To formulate the objective function we introduce some notations. Let N_s be the number of sensors installed in various points in the porous medium and $\{\mathbf{x}_j\}_{j=1}^{N_s}$ denote such points. Let N_t be the number of how many times the concentration is measured in time and $\{t_k\}_{k=1}^{N_t}$ denote such time levels. Furthermore,

$\gamma_j(t_k)$ denotes the measured concentration at sensor located in $\mathbf{x}_j$ and at time t_k. We seek initial data in a finite dimensional space spanned by $\tilde{C}_i^0(\mathbf{x})$,

$$\tilde{C}^0(\mathbf{x}) = \sum_{i=1}^{N_c} \alpha_i \tilde{C}_i^0(\mathbf{x}), \tag{3}$$

for some $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_{N_c})$. Furthermore, let $\tilde{C}_i(\mathbf{x}, t)$ be the solution of (1) using an initial condition $\tilde{C}_i^0(\mathbf{x})$. Then by superposition principle, the solution of (1) using $\tilde{C}^0(\mathbf{x})$ in (3) as an initial condition has the following form:

$$\tilde{C}(\mathbf{x}, t) = \sum_{i=1}^{N_c} \alpha_i \tilde{C}_i(\mathbf{x}, t). \tag{4}$$

The objective function which can be formulated in terms of α, quantifies the difference between the measured concentration and the simulated concentration, $\tilde{C}(\mathbf{x}, t)$. In general the number of the sensors are less than the dimension of the space for initial data. Hence, an attempt to minimize an objective function that only contains the difference between measurements and simulations will lead to an ill-posed problem. To regularize the problem, we add a penalty term that contains the prior information related to the initial data, and consider the following objective function

$$F(\alpha) = \sum_{j=1}^{N_s} \left(\sum_{i=1}^{N_c} \alpha_i \tilde{C}_i(\mathbf{x}_j, t) - \gamma_j(t) \right)^2 + \sum_{i=1}^{N_c} \kappa_i \left(\alpha_i - \beta_i \right)^2. \tag{5}$$

Here $\kappa = (\kappa_1, \kappa_2, \cdots, \kappa_{N_c})$ is the penalty coefficients for an *a priori* vector $\beta = (\beta_1, \beta_2, \cdots, \beta_{N_c})$. This prior information will be updated during the simulation to achieve higher accuracy.

Next we present a representative numerical example. More numerical and theoretical studies have been performed and the results will be reported elsewhere. We use $\Omega = [0, 1] \times [0, 1]$. The boundary conditions in the subsurface flow for the pressure equation (2) are given pressure at the inlet and outlet edges (i.e., $x = 0$ and $x = 1$, respectively), and no flow at the bottom and top edges (i.e., $z = 0$ and $z = 1$, respectively). The permeability k is generated with given correlation length $l_x = 0.25$ and $l_z = 0.02$, with a spherical variogram using GSLIB algorithms [2]. For the convection-diffusion equation (1), we set the diffusion coefficient $D = 0.1$ over all domain. We assume zero concentration at the inlet, bottom, and top edges, and a zero diffusion, i.e., $(D\nabla C) \cdot \boldsymbol{n} = 0$, at the outlet edge, with $\boldsymbol{n}$ being the unit normal vector pointing outward on the outlet edge. The initial condition $C^0(x, z)$ is set to be nonzero in the region $(0.2, 0.4) \times (0.2, 0.4)$ and zero elsewhere.

Both pressure and convection-diffusion equations are solved by the finite volume method using rectangular grids. We discretize the domain into 100×100 elements, i.e., 100 elements in each direction. The sensors information are obtained from the concentration solved with the given "true" initial condition. We

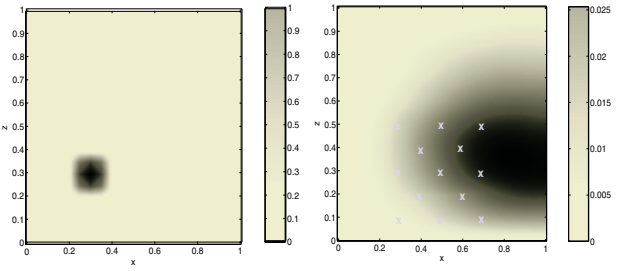

Fig. 1. Left: The initial condition profile - Right: Concentration at t=0.4 ((x) indicates the sensor location)

use time step $\Delta t = 0.01$. For the example presented in this paper we use initial condition shown on the right side of Figure 1.

The data from measurement are taken from multiple set of numerical simulations with the initial condition mentioned earlier. The measurement is assumed to be conducted at time level $t_1 = 0.1$, $t_2 = 0.2$, $t_3 = 0.3$, and $t_4 = 0.4$. A number of sensors are installed at various locations in the porous medium. Figure 1 shows the concentration profile at $t = 0.4$ along with the sensor locations which are denoted by (x) indicator.

We have performed numerical tests by only updating β as well as updating β and κ. In all cases, we observed significant improvement as the initial data is updated. Here we present only the case of both β and κ are updated. In particular, κ is increased during the simulations to reflect the fact that the updated initial condition is a better representation of the true initial condition. In general, one can change κ in various ways. In our examples, we update κ_i by increasing 10 times after each update. Figure 2 shows the updated initial condition with both β and κ updated for the case of larger support. The prior for κ is $\kappa_i^0 = 2 \times 10^{-12}$ for all i, and when updated it is multiplied by ten. The figure shows significant improvement on the predicted initial condition.

As mentioned earlier, sensor measurements contain errors and uncertainties. In our numerical simulations, we can take into account these uncertainties by sampling the sensor data from the known distribution. As a result, one obtains

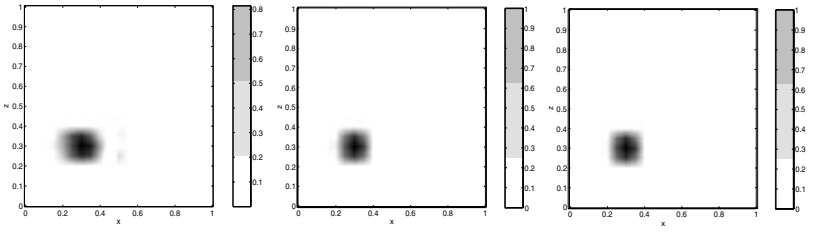

Fig. 2. Updated initial condition: $t = 0.1$ (left), $t = 0.3$ (middle), $t = 0.4$ (right) - Both β and κ are updated

various realizations of the initial data. In our subsequent work [4], we employ the least squares approach in developing Bayesian methods for nonlinear problems. To quantify uncertainties in the measurements and *a priori* knowledge about the initial data, the Markov Chain Monte Carlo method (MCMC) can be used. Because this method is expensive due to rejection of the proposals, we propose an approach that combines the least squares method with Bayesian approaches that will give high acceptance rates.

3 Multiscale Interpolation Techniques

Our goal in this section is to discuss the mapping of the sensor data to the finite dimensional space where the solution is calculated. This procedure is nontrivial in general, because the solution space usually has high dimension, while the sensors are located only at few locations. Our simplified approach presented in this paper consists of passing the sensor data to the simulations and its use for the next time step simulations. Since the sensor data represents the solution only at few coarse locations one has to modify the solution conditioned to this data. This step we call multiscale interpolation which consists of mapping the sensor data to the solution space. At each time step the sensor data is received by the simulator. There are two options to handle the data. We can treat it as hard data or as soft data. The latter means that the data contains some noise and not needed to be imposed exactly. In this paper the first approach, "hard constraint", will be considered. At the beginning of each time step we need to map the sensor data to the solution space. This is performed using DDDAS mapping operator, the main feature of which is not to alter the heterogeneous field.

The proposed mapping for the sensor data is general and applicable to various classes of equations. To demonstrate this we consider general nonlinear parabolic equations

$$\frac{\partial}{\partial t}u_\epsilon = \nabla \cdot (a_\epsilon(x,t,u_\epsilon,\nabla u_\epsilon)) + a_{0,\epsilon}(x,t,u_\epsilon,\nabla u_\epsilon), \text{ in } \Omega \times [0,T], \tag{6}$$

where ϵ indicates the presence of the small scales heterogeneities. This equation includes various physical process that occur in subsurfaces. In the next section numerical examples for particular cases of (6) will be discussed. Assume the domain is divided into the coarse grid such that the sensor points are the nodal points of the coarse grid. Note that we do not require all nodal points to be sensor locations. Further denote by S^h the space of piecewise linear functions on this partition,

$$S_h = \{v_h \in C^0(\overline{\Omega}) : \text{the restriction } v_h \text{ is linear for each triangle } K \in \Pi_h\}.$$

Our objective now is to map the function defined on S^h to the fine grid that represents the heterogeneities. This grid is obtained from *apriori* information about

the field using geostatistical packages. Denote by the operator E the mapping from the coarse dimensional space into the fine grid,

$$E : S^h \to V_\epsilon^h,$$

which is constructed as follows. For each element in $u_h \in S^h$ at a given time t_n we construct space time function $u_{\epsilon,h}(x,t)$ in $K \times [t_n, t_{n+1}]$ such that it satisfies

$$\frac{\partial}{\partial t} u_{\epsilon,h}(x,t) = \nabla \cdot (a_\epsilon(x,t,\eta,\nabla u_{\epsilon,h})) \tag{7}$$

in each coarse element K, where η is the average of u_h. $u_{\epsilon,h}(x,t)$ is calculated by solving (7) on the fine grid, and thus it is a fine scale function. To complete the construction of E we need to set boundary and initial conditions for (7). One can set different boundary and initial conditions and this will give rise to different maps. In our numerical simulations we will take the boundary and initial condition for the local problems to be linear with prescribed nodal values. These values are obtained from the sensor data, if available. If the sensor data is not available at some location we use the values obtained from the simulations. Different local boundary conditions can be also imposed and will be discussed later. Mathematical aspects of this interpolation operator, such as convergence and etc, are described in [5].

Once the solution at time $t = t_n$ is computed its values with sensor data at the sensor locations are compared. After changing the values of the solution we interpolate it to the fine grid and use it for the next time step. At the last step we use multiscale approach which is computationally efficient. In particular, the solution at the next time step is calculated based on

$$\int_\Omega (u_h(x,t_{n+1}) - u_h(x,t_n))v_h dx + \sum_K \int_{t_n}^{t_{n+1}} \int_K ((a_\epsilon(x,t,\eta,\nabla u_{\epsilon,h}),\nabla v_h) + a_{0,\epsilon}(x,t,\eta,\nabla u_{\epsilon,h})v_h)dxdt = \int_{t_n}^{t_{n+1}} \int_\Omega f dxdt. \tag{8}$$

Here Ω refers to the spatial domain and K are the coarse elements. We would like to note that our approach has limitations and it is not applicable when there are large deviations between sensor data and the solution.

We now present a representative numerical example that demonstrates the accuracy and limitations of our proposed method. The detailed numerical studies will be reported elsewhere. The systems we consider are intended to represent cross sections (in $x - z$) of the subsurface. The fine-scale permeability field is generated on 121×121 grid using GSLIB algorithms [2] with an exponential covariance model. We consider $\frac{\partial}{\partial t} u = \nabla \cdot (a_\epsilon(x)\nabla u)$, where the original "true" diffusion coefficient $a_\epsilon(x) = \exp(\alpha_\epsilon(x))$, where $\alpha_\epsilon(x)$ is a realization of the random field with correlation lengths $l_x = 0.3$, $l_z = 0.02$ and variance $\sigma = 0.75$. For the simulation purposes we consider the diffusion coefficients to be the same realization of the random field but with $\sigma = 1.5$. Thus we assume that the heterogeneities have the same nature and the only difference between the true field

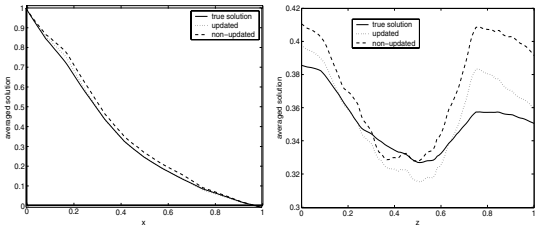

Fig. 3. Comparisons of the average solutions across x and z directions: Solid line designates the true solution, dotted line designates the solution obtained using our simulations with 4 updates, and the dashed line designates the solution that has not been updated

and the one used in the computations is associated with the scaling. Objective of this numerical results is to demonstrate how frequency of updating sensor data in the simulations improves the accuracy of the method. In this example the sensor data is used four times during simulations, i.e., the frequency of updating is 4. In Figure 3 we plot the average of the solutions. Solid line designates the fine scale (true) simulation results, while dotted line represents the results obtained using our methodology with 4 updating. The dashed line represents the simulation results with no updating. As we see from this figure the simulation with data updating performs better compare to that with no updating. The l_2 error between the true solution and the one corresponding with 4 updating is about 6 percent, while l_2 error corresponding with no updating is almost 9 percent. Simulations with more frequent update indicate that the frequent updating improve the accuracy of the predictions and thus important for DDDAS. This is also observed in various numerical examples for both linear and nonlinear equations which we have tested (see [3]).

References

1. J. Bear, *Dynamics of Fluids in Porous Media*, Elsevier, 1972.
2. C. V. Deutsch and A. G. Journel, *GSLIB: Geostatistical software library and user's guide, 2nd edition*, Oxford University Press, New York, 1998.
3. C. C. Douglas, C. Shannon, Y. Efendiev, R. Ewing, V. Ginting, R. Lazarov, M. Cole, G. Jones, C. Johnson, and J. Simpson, *A note on data-driven contaminant simulation*, Lecture Notes in Computer Science, Springer-Verlag, 3038 (2004), pp. 701–708.
4. C. C. Douglas, Y. Efendiev, R. Ewing, V. Ginting, and R. Lazarov, *Bayesian approaches for initial data recovery in dynamic data-driven simulations*, in preparation.
5. Y. Efendiev and A. Pankov. Numerical homogenization of nonlinear random parabolic operators. *SIAM Multiscale Modeling and Simulation*, 2(2):237–268, 2004.
6. C.R. Johnson, S. Parker, and et al. SCIRun: A scientific computing problem solving environment. http://software.sci.utah.edu/scirun.html.