A Note on Data-Driven Contaminant Simulation

Craig C. Douglas[1,2], Chad E. Shannon[1], Yalchin Efendiev[3], Richard Ewing[3], Victor Ginting[3], Raytcho Lazarov[3], Martin J. Cole[4], Greg Jones[4], Chris R. Johnson[4], and Jennifer Simpson[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,simpson}@cs.utah.edu

Abstract. In this paper we introduce a numerical procedure for performing dynamic data driven simulations (DDDAS). The main ingredient of our simulation is the multiscale interpolation technique that maps the sensor data into the solution space. We test our method on various synthetic examples. In particular we show that frequent updating of the sensor data in the simulations can significantly improve the prediction results and thus important for applications. The frequency of sensor data updating in the simulations is related to streaming capabilities and addressed within DDDAS framework. A further extension of our approach using local inversion is also discussed.

1 Introduction

Dynamic data driven simulations are important for many practical applications. Consider an extreme example of a disaster scenario in which a major waste spill occurs in a subsurface near a clean water aquifer. Sensors can now be used to measure where the contamination is, where the contaminant is going to go, and to monitor the environmental impact of the spill. One of the objectives of dynamic data driven simulations is to incorporate the sensor data into the real time simulations. Many important issues are involved in DDDAS for this application (see [1,2]).

Subsurface formations typically exhibit heterogeneities over a wide range of length scales while the sensors are usually located at sparse locations and sparse data from these discrete points in a domain is broadcasted. Since the sensor data usually contains noise it can be imposed either as a *hard* or a *soft constraint.* To incorporate the sensor data into the simulations, we introduce a multiscale interpolation operator. This is done in the context of general nonlinear parabolic operators that include many subsurface processes. The main idea of this interpolation is that we do not alter the heterogeneities of the random field that drives the contaminant. Instead, based on the sensor data, we rescale the solution in a manner that it preserves the heterogeneities. The main idea of this rescaling is the use of local problems. This interpolation technique fits nicely with a new multiscale framework for solving nonlinear partial differential equations. The combination of the interpolation and multiscale frameworks provides a robust and fast simulation technique that is employed in the present paper.

We have tested our method on some synthetic examples by considering both linear and nonlinear cases. 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. Our numerical studies bear out that simulations that do not use the update or use it very infrequently produces large errors. Finally, we discuss the extension of the approach that uses some inversion of the local data.

In a real field, the sensors are typically in wells at different depths. A nice, tensor product mesh does not fit the reality of sensor locations. We are forced to work with unstructured, quite coarse grids if we want the sensors to be nodes on a mesh. The sensor data comes in the form of telemetry. The sensors can provide information at specified times, when specific conditions occur, be polled, or any number of combinations. We have developed a virtual telemetry package that is now part of SCIRun [1,2,3,4] so that we can simulate a field while computing remotely, as is typical in a real field simulation.

2 Data Driven Simulations

In this section we discuss the mapping of the sensor data to the finite dimensional space where the solution is calculated. This procedure is nontrivial in general since the solution space usually has high dimension while the sensors are located only at few locations. To demonstrate this in Fig. 1 we schematically plot a gray scale image of a 121×121 heterogeneous field with exponential variogram and the correlation lengths $l_x = 0.2$ and $l_x = 0.01$. Sensors in Fig. 1 are marked by an X. Due to uncertainties in the data, the random fields used for the simulations and the true field can differ significantly. Thus, we must be able to reconcile the streamed data from the sensors with that produced by our simulations.

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 we have to modify the solution conditioned to this data. We call this step *multiscale interpolation*. It consists of mapping the sensor data to the solution space. Before discussing the mapping, we demonstrate the main idea of our general approach handling dynamic data. It is depicted in Fig. 2 and consists of the following: At each time step the sensor data is received by the simulator. We can treat the data either as a hard or soft constraint. The latter means the data contains some noise and need not be imposed exactly.

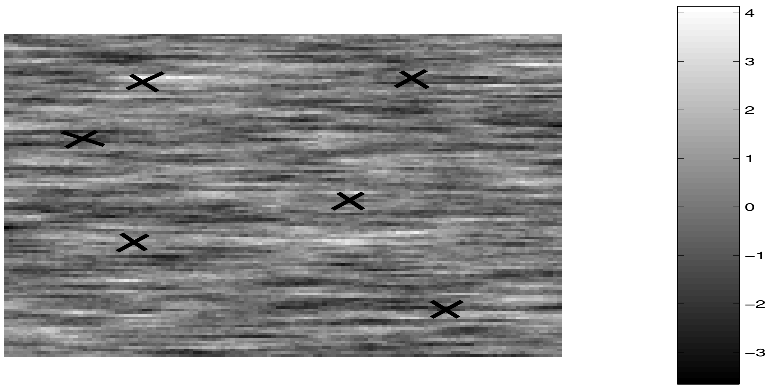

Fig. 1. Permeability field with spherical variogram and correlation lengths $l_x = 0.2$, $l_z = 0.02$. The locations of the sensors are marked.

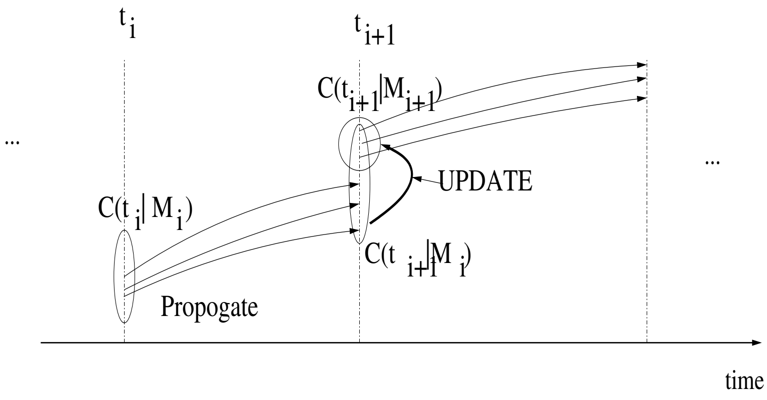

Fig. 2. Schematic description of data driven simulations. $C(t_i|M_i)$ designates the quantity of interest, such as the concentration, at time t_i, conditioned to some data M_i. In our basic methodology we consider M_i to be the sensor data. In the extension of our basic method the local heterogeneous field is also updated at each time based on sensor readings.

Consider the hard constraint case. At the beginning of each time step the sensor data needs to be mapped to the discrete solution space. This is performed using our DDDAS mapping operator, whose main goal is *not* to alter the heterogeneous field, i.e., at each time we update the data while not seeking the error source.

The proposed mapping for the sensor data is very general and applicable to various classes of equations. We consider general nonlinear parabolic equations of the form

$$D_t u_\epsilon = div(a_\epsilon(x, t, u_\epsilon, D_x u_\epsilon)) + a_{0,\epsilon}(x, t, u_\epsilon, D_x u_\epsilon), \text{ in } Q_0 \times [0, T], \qquad (1)$$

where Q_0 refers to the spatial domain and ϵ indicates the presence of small scale heterogeneities. Equation (1) includes various physical processes that occur in subsurfaces.

Assume the domain is divided into a coarse grid such that the sensor points are 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,

$$S_h = \{v_h \in C^0(\overline{Q_0}) : \text{the restriction } v_h \text{ is linear for each triangle } K \in \Pi_h\}.$$

Note that the K's are the coarse elements.

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 *a priori* information about the field using geostatistical packages. Denote by the operator E the mapping from the coarse dimensional space into the fine dimensional space:

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

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

$$D_t u_{\epsilon,h}(x,t) = div(a_\epsilon(x,t,\eta, D_x u_{\epsilon,h})) \tag{2}$$

in each coarse element K, where η is the average of u_h. $u_{\epsilon,h}(x,t)$ and is calculated by solving (2) on the fine grid, and thus is a fine scale function.

To complete the construction of E we need to set boundary and initial conditions for (2). We can set different boundary and initial conditions, giving rise to different maps that only differ from each other slightly. The main underlying property of our map is that it is constructed as a solution of local problems. The latter guarantees that the solution is consistent with prescribed heterogeneities.

In our numerical simulations we take the boundary and initial condition for the local problems to be linear with prescribed nodal values. The nodal values are obtained from the sensor data, if available. If the sensor data is unavailable at some location we use the values obtained from the simulations at previous time. Note that we cannot impose the values of the solutions directly at some locations since it can cause artificial discontinuities in the solution. See [5] for mathematical aspects of this interpolation operator, including convergence properties.

Once the solution at time $t = t_n$ is computed its values with sensor data at the sensor locations can be 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 a multiscale approach which is computationally efficient. In particular, the solution at the next time step is calculated based on

$$\int_{Q_0} (u_h(x,t_{n+1}) - u_h(x,t_n))v_h dx + \sum_K \int_{t_n}^{t_{n+1}} \int_K ((a_\epsilon(x,t,\eta, D_x u_{\epsilon,h}), D_x v_h) + a_{0,\epsilon}(x,t,\eta, D_x u_{\epsilon,h})v_h) dxdt = \int_{t_n}^{t_{n+1}} \int_{Q_0} f v_h dxdt. \tag{3}$$

Recall that Q_0 refers to the spatial domain and the K's are the coarse elements. This approach, combined with the interpolation technique, has great CPU advantages over just a fine scale calculations (see [5]).

3 Numerical Examples

We now present numerical results that demonstrate the accuracy and limitations of our proposed method. We have explored both linear and nonlinear heterogeneous diffusion cases. Due to space limitations here, we only provide a representative example. We will provide more examples and discussions in a future work.

The systems we consider are intended to represent cross sections (in the x-z plane) of the subsurface. For that reason we take the system length in x (L_x) to be five times the length in z (L_z). All of the fine grid fields used in this study are 121×121 realizations of prescribed overall variance (σ^2) and correlation structure. The fields were generated using GSLIB algorithms [6] with an exponential covariance model. In the examples below, we specify the dimensionless correlation lengths l_x and l_z, where each correlation length is nondimensionalized by the system length in the corresponding direction. For example, $l_x = 0.3$ means that the actual correlation length of the permeability field is $0.3L_x$.

In our calculations, we consider (1) with a fixed concentration at the inlet edge of the model ($x = 0$) and a fixed concentration at the outlet edge ($x = L_x$) of the model. The top and bottom boundaries are closed to flow. Initially zero contaminant concentration is assumed in the domain. For comparison purposes most results are presented in terms of cross sectional averages and l_2 norm errors. The computations are carried out until the final time $t = 0.1$ with a different frequency of updating.

We have tested both linear and nonlinear problems and observed similar numerical results. Here we will only present numerical results for the nonlinear case which represents simplified Richards' equation. Consider

$$D_t u_\epsilon = div(a_\epsilon(x, u_\epsilon) D_x u_\epsilon),$$

where $a_\epsilon(x,\eta) = k_\epsilon(x)/(1+\eta)^{\alpha_\epsilon(x)}$. $k_\epsilon(x) = \exp(\beta_\epsilon(x))$ is chosen such that $\beta_\epsilon(x)$ is a realization of a random field with the exponential variogram and with some correlation structure. $\alpha_\epsilon(x)$ is chosen such that $\alpha_\epsilon(x) = k_\epsilon(x) + const$ with the spatial average of 2. For the numerical examples we will only specify $\beta_\epsilon(x)$.

In all of the figures, solid line designates the true solution, dotted line designates the solution obtained using our simulations with some number of updates, and the dashed line designates the solution that has not been updated.

For the first example we consider the true field to be a random field with exponential variogram and with $l_x = 0.2$, $l_x = 0.02$, and $\sigma = 1$. For our simulations we use the same field with $\sigma = 2$. In Fig. 3 we compare the average solutions for the case with 20 updating, i.e., the simulated solution is updated 20 times during the course of the simulations. Fig. 3 demonstrates that the predictions that do not use the sensor data perform very poorly. The l_2-norm of the difference

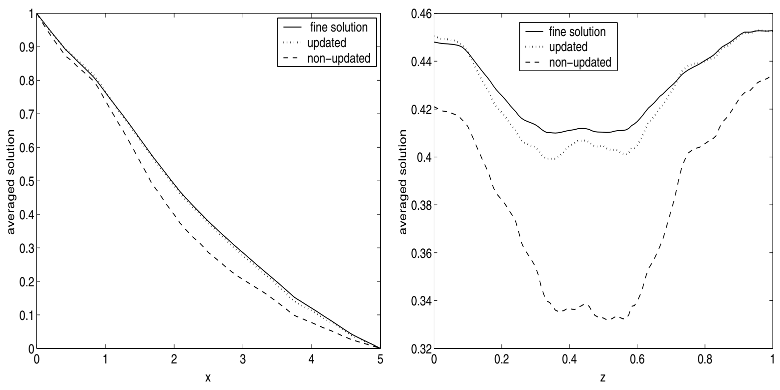

Fig. 3. Comparisons of the average solutions across x and z directions for nonlinear case. The true field is a realization of a random field with $l_x = 0.2$, $l_z = 0.01$, $\sigma = 1$, while the random field used for the simulations has $\sigma = 2$.

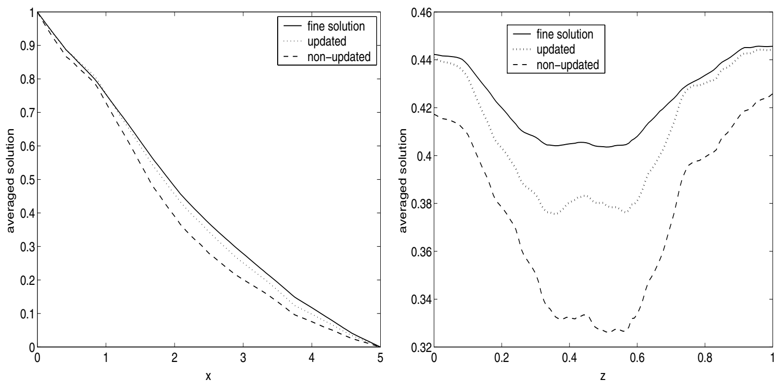

Fig. 4. Comparisons of the average solutions across x and z directions for nonlinear case. The true field is a realization of a random field with $l_x = 0.2$, $l_z = 0.01$, $\sigma = 1$, while random field used for the simulations has $\sigma = 2$.

between the solutions for the data that are frequently updated is 2.5% while the data that is not updated is 14%.

In our next example we ran the same case with less frequent updating. In particular, we use 4 updates during the simulations. The results are depicted in Fig. 4. The l_2-errors of the solution that is updated is 5.7% while the l_2 error of the non-updated solution is still 14%.

Finally we have considered a case where three different heterogeneous fields with exponential variograms are used with different probabilities. In particular the true field consists of $l_x = 0.1$, $l_z = 0.02$ with probability 0.1, $l_x = 0.2$, $l_z = 0.02$ with probability 0.8, $l_x = 0.4$, $l_z = 0.02$ with probability 0.1. For all these fields we take $\sigma = 1$. For our simulations we use these random fields with $\sigma = 2$ and with different probabilities. In particular, we take the field with

$l_x = 0.1$, $l_z = 0.02$ with probability 0.3, $l_x = 0.2$, $l_z = 0.02$ with probability 0.4, $l_x = 0.4$, $l_z = 0.02$ with probability 0.3. This is a typical field scenario when one does not know the probability distributions of the random fields. Our numerical results showed that the l_2 error between true and updated solution is 5%, while the error between the true solution and non-updated solution is 9%. The numerical results with frequent updates demonstrated higher accuracy.

4 Fixing Error Sources Using Inversion

So far we have presented a numerical method that incorporates the sensor data from sparse locations into the simulations. Currently we are working on the possible extension of the methodology that involves some kind of inversion during the simulations.

In the methodology described previously we only correct the solution without fixing the error source. One of the error sources is the incorrect heterogeneous diffusion field.

Consider the linear heterogeneous diffusion model $D_t u_\epsilon = div(a_\epsilon(x) D_x u_\epsilon)$. Our objective is to compare the misfit of the data from the sensors and the data from our simulator. If this misfit is larger than a certain threshold for some sensor points we perform an inversion in the neighborhood of the sensor point by modifying the diffusion field $a_\epsilon(x)$ in the neighborhood of the sensor point. By considering the neighborhood of the sensor location as a coarse block we determine the effective diffusivity, a^*_{sim}, corresponding to the existing heterogeneous diffusivity field. This quantity is determined such that the local problem with a single diffusion coefficient, a^*_{sim}, will give the same response as the underlying fine heterogeneous diffusion field.

Our next goal is to find an a^*_{true} that will give us the same response as the one from the sensor data. This is done using some classical inversion, such as parameter estimation. Since we only have to fit one parameter this problem can be easily handled without extensive computations.

Now we can impose a^*_{true} both as a soft as well as a hard constraint. For the hard constraint we must rescale the local heterogeneous field based on the ratio a^*_{sim}/a^*_{true}. Note that the hard constraint does not change the structure of the local heterogeneities and can be easily implemented since it only requires local rescaling.

For the soft constraint we use Bayesian framework (see [7]). Assume the *a priori* distribution for the local coarse heterogeneous field is given. Our objective is to generate fine-scale random diffusivity fields conditioned on the coarse-scale data. Denote a^c and a^f to be coarse and fine scale diffusivity fields respectively. Using Bayes theorem we obtain

$$P(a^f|a^c) \propto P(a^f)P(a^c|a^f), \tag{4}$$

where $P(a^f)$ is the probability distribution for the fine-scale field. In (4) $P(a^f)$ is *a priori* distribution of the fine-scale field which is prescribed.

To impose the coarse-scale data as a soft constraint we take $a^c = J(a^f) + \epsilon$, where J is a local upscaling operator and ϵ is a random noise, $\epsilon \sim N(0, \sigma^2)$. The local upscaling operator involves the solution of the local partial differential equations similar to (2). A soft constraint assumes that the coarse scale information is not accurate. Note that letting $\sigma \to 0$ we get a hard constraint.

The realizations from the posterior distribution (4) is generated using Markov Chain Monte Carlo (MCMC) [8]. This approach is known to be very general and can handle complex posterior distributions. The main idea of MCMC is to use a Markov Chain with a specified stationary distribution. The random drawing from the target distribution can be accomplished by a sequence of draws from full conditional distribution. Since the full conditionals have non-explicit form involving local problems Metropolis-Hastings algorithm is used.

We have tested our methodology with both soft and hard constraint on simple examples using Markov Random Fields as prior distributions for fine scale fields. Our numerical results indicated very good correspondence between the true and simulated diffusion fields on the coarse level. Further research into this area is warranted.

References

1. Douglas, C.C., Efendiev, Y., Ewing, R., Lazarov, R., Cole, M.R., Johnson, C.R., Jones, G.: Virtual telemetry middleware for DDDAS. Computational Sciences - ICCS 2003, P. M. A. Sllot, D. Abramson, J. J. Dongarra, A. Y. Zomaya, and Yu. E. Gorbachev (eds.) **4** (2003) 279–288
2. Douglas, C.C., Shannon, C.E., Efendiev, Y., Ewing, R., Lazarov, R., Cole, M.R., Johnson, C.R., Jones, G., Simpson, J.: Virtual telemetry middleware for DDDAS. In Darema, F., ed.: Dynamic Data-Driven Application Systems. Kluwer, Amsterdam (2004)
3. Johnson, C., Parker, S.: The scirun parallel scientific computing problem solving environment. In: Ninth SIAM Conference on Parallel Processing forScientific Computing, Philadelphia, SIAM (1999)
4. SCIRun: A Scientific Computing Problem Solving Environment. Scientific Computing and Imaging Institute (SCI), http://software.sci.utah.edu/scirun.html, 2002.
5. Efendiev, Y., Pankov, A.: Numerical homogenization of nonlinear random parabolic operators. SIAM Multiscale Modeling and Simulation (to appear) Available at http://www.math.tamu.edu/~yalchin.efendiev/ep-num-hom-parab.ps.
6. Deutsch, C.V., Journel, A.G.: GSLIB: Geostatistical software library and user's guide, 2nd edition. Oxford University Press, New York (1998)
7. Lee, S.H., Malallah, A., Datta-Gupta, A., Higdon, D.: Multiscale data integration using markov random fields. SPE Reservoir Evaluation and Engineering (2002)
8. Gilks, W., Richardson, S., Spegelhalter, D.: Markov Cain Monte Carlo in Practice. Chapman and Hall/CRC, London (1996)