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

Craig C. Douglas

University of Kentucky and Yale University
douglas-craig@cs.yale.edu

In cooperation with
Li Deng, Wei Li, Ryan McKenzie, Yalchin Efendiev, Richard Ewing, Victor Ginting, Raytcho Lazarov, Martin Cole, Greg Jones, Chris Johnson.

This work has been supported in part by NSF grants
EIA-0219627 and ACI-0324876 (Kentucky),
EIA-0218229 (Texas A&M), and EIA-0218721 (Utah).

Outline

- Introduction and Motivation
- Backward Error Analysis and Initial Data Recovery
- Multiscale Interpolation Techniques

Intent of Talk

- Address several DDDAS enabling technologies in the context of a specific application area: *contaminant tracking*.
  - Groundwater reservoirs are 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.

- Provide techniques and tools to effectively demonstrate the potential of DDDAS for other areas.

Motivation: DDDAS Specific→General Solution?

- We want to run a simulation for a very long time: potentially for weeks, months, or years of wall clock time (versus simulation prediction time).

- We are interested in a sliding window in time of predictive solutions without doing restarts. We can always think of the earliest time in the window as the initial time. We can correct the initial time's data and easily update the solutions within the time window.

- Our problems do not provide us with accurate boundary or initial conditions. We have to guess both, which we do iteratively, in order to convert an *ill posed* problem into a *well posed* one.

Commonality in Applications

- Computer models that solve nonlinear, unsteady, coupled, partial differential equations that generate very large output sets.
- Require consistent initial conditions, adequate forcing fields, and boundary conditions to advance the solution in time.
- Input data originates from sensors, internally generated from ensemble type simulations, or can be externally generated (e.g., providing boundary conditions for very high resolution regional models).
- 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.

Sensor Measurement Based Updates

- Data streamed from few spatial locations.
- As data is injected, we update
  - the solution, using a multiscale interpolation technique for updating the solution.
  - the initial condition.
  - the media properties - currently beyond project.

Solution Update Methodology

- Multiscale interpolation technique used for updating the solution due to heterogeneities of the porous media. Done in the context of general nonlinear parabolic operators that include subsurface processes.
- The interpolation does not alter the heterogeneities of the random field that drives the contaminant.
- Based on the sensor data the solution is rescaled so that it preserves the heterogeneities. The rescaling uses the solution of the local problems.
- Simulations that do not use the update or use it less frequently produce large errors. Update often enough and errors are small.

Persistent Errors in Simulation

- The errors in the simulations will persist if one does not change the input parameters. As new data is obtained from sensors measurements, the initial data should be updated.

- Update reduces the computational errors associated with incorrect initial data and improves the predictions.

- We consider linear subsurface flows involving convection and diffusion.

- Initial data is sought in a finite dimensional space. The approximation of the initial data is recovered using the first set of measurements. As new data is incorporated into the simulator, the initial data using an objective function are updated.

Ill Posed Problem: 2 Facts, 1 Folklore

- The data gathered from the sensor measurements always contains some defects, e.g., human errors and inherent factory errors of the sensors.

- The number of sensors that can be installed is limited and is quite low order in comparision to the finite dimensional space describing the initial data. We regularize the problem by using the filtered updated initial data instead of the initial data. Penalization constants depend on time of update and can be associated with the relative difference between simulated and measured values.

- Numerical examples show the improvement of the predictions as new data is taken into account.

Backward Error Analysis and Initial Data Recovery

- As new data is injected from sensor measurements, *earlier data*, i.e., the contaminant distribution, can be updated. Due to poor knowledge of earlier locations of contaminants, this type of error can be dominant in simulations.

- We seek the earlier data in a finite dimensional space. The problem becomes more ill-posed as this dimension grows.

- The dimension can be reduced using multiscale representation of earlier data.

- As new data are incorporated into the simulator, we update earlier data using an objective function that we will define shortly.

Model Equations

Linear transport model dominated by convection and diffusion:

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

By Darcy's Law, $v = -k \nabla p$. Pressure p satisfies

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

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

Object Function Notation

- N_s is the number of sensors installed in various points in the porous medium and $\{\mathbf{x}_j\}_{j=1}^{N_s}$ denote such points.

- N_t is how many times the concentration is measured in time and $\{t_k\}_{k=1}^{N_t}$ denote such time levels.

- $\gamma_j(t_k)$ is 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 $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_{N_c})$.

- 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:

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

Object Function Formulation

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

- Includes a penalty term that contains the prior information related to the initial data to regularize the problem.

- $\kappa = (\kappa_1, \kappa_2, \cdots, \kappa_{N_c})$ 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.

Numerical Examples' Configuration

- Unit square $\Omega = [0,1] \times [0,1]$.
- 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.
- For the convection-diffusion equation (1), we set the diffusion coefficient $D = 0.1$ over all domain.

- We assume zero concentration coefficient $D = 0.1$ over all domain.
- We assume zero concentration at the inlet, bottom, and top edges, and a zero diffusion, i.e., $(D\nabla C) \cdot \vec{n} = 0$, at the outlet edge, with $\vec{n}$ 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.
- The domain is descretized into 100 elements in each direction.
- Time step $\Delta t = 0.01$.

Example 1: Only β Updated

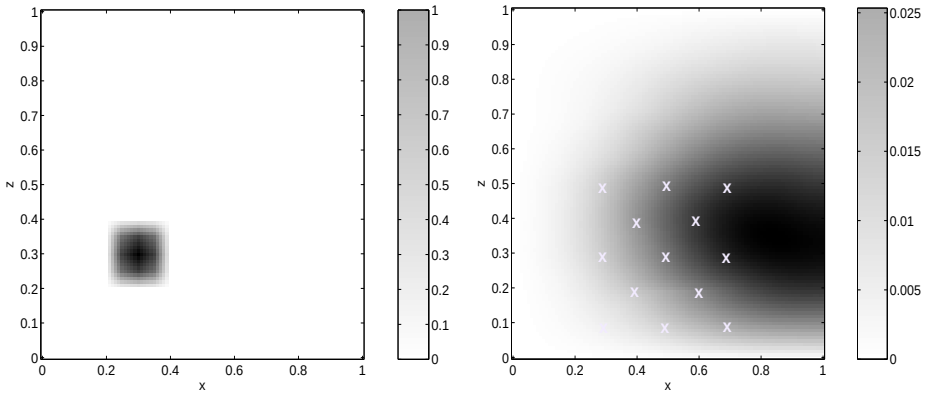

Figure 1: Left: The initial condition profile. Right: Concentration at t=0.4 ((x) indicates the sensor location).

- Measurement at time level $t_1 = 0.1$, $t_2 = 0.2$, $t_3 = 0.3$, and $t_4 = 0.4$.

Example 2: Both β and κ Updated

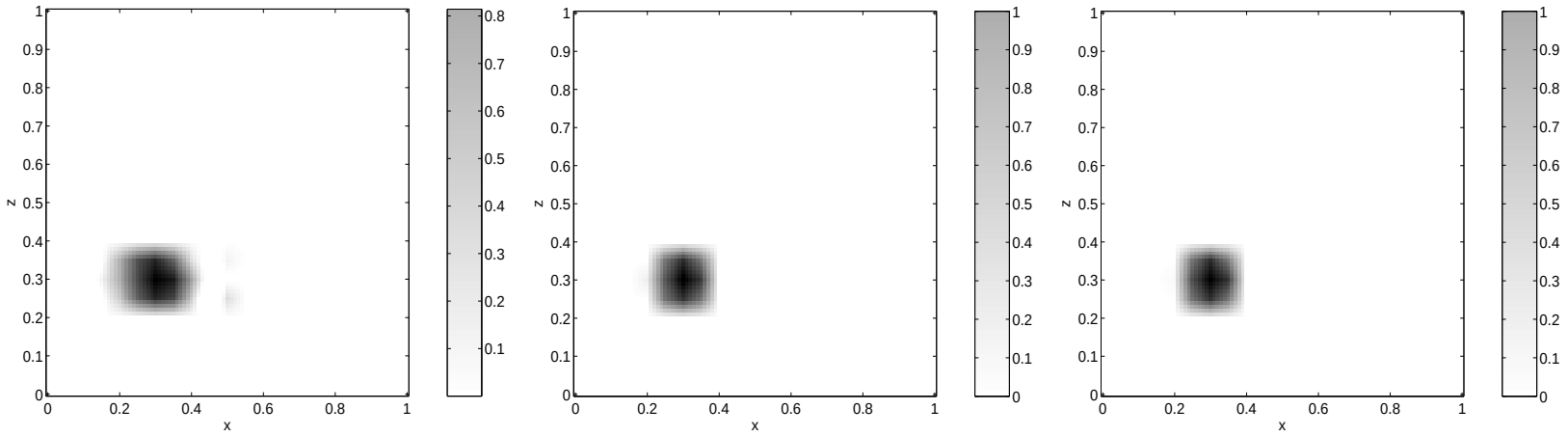

Figure 2: Updated initial condition: $t = 0.1$ (left), $t = 0.3$ (middle), $t = 0.4$ (right). Both β and κ are updated.

Example 3: Twin Peaks

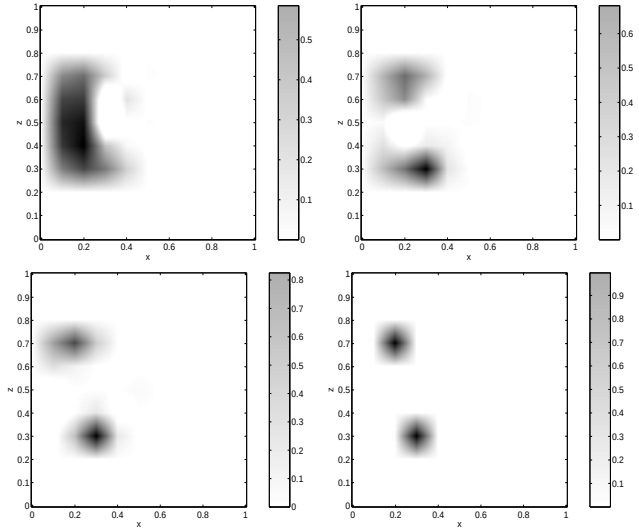

Figure 3: Updated initial condition: $t = 0.1$ (top left), $t = 0.2$ (top right), $t = 0.3$ (bottom left), $t = 0.4$ (bottom right). The prior for β assumes a larger support.

Multiscale Interpolation Principles

- Nontrivial since the solution space usually has high dimension while the sensors are located at just a few locations. We pass the sensor data to the simulations its use in the next time step.

- Since the sensor data represents the solution only at a few coarse locations we modify the solution conditioned to this data. Two options for data:
  - hard data is assumed to be correct, and
  - soft data contains some noise and does not have to be imposed exactly.

- At the beginning of each time step we need to map the sensor data to the solution space. This is performed using our DDDAS mapping operator, which tries not to alter the heterogeneous field.

Multiscale Interpolation Techniques

- Consider a general nonlinear parabolic equation:

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

  where ϵ indicates the presence of the small scales heterogeneities. This equation includes various physical process that occur in subsurfaces.

- Assume the domain is divided into the coarse grid such that the sensor points are among the nodal points of the coarse grid. S^h is the space of piecewise linear functions on this partition:

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

- We will map the function defined on S^h onto the fine grid representing the heterogeneities by constructing

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

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). Different boundary and initial conditions result in different maps.

  - We take the boundary and initial condition for the local problems to be linear with prescribed nodal values.
  - 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 imposed.

- 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.

– The solution at the next time step is calculated based on

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

Ω refers to the spatial domain and K are the coarse elements.

– Note that our approach has limitations and it is not applicable when there are large deviations between sensor data and the solution.

Example 4: Configuration

- 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 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 $a_\epsilon(x) = \exp(\alpha_\epsilon(x))$, of the random field with correlation lengths $l_x = 0.3$, $l_z = 0.02$ and variance $\sigma = 0.75$.
  - We consider the diffusion coefficients to be the same realization of the random field, but with $\sigma = 1.5$.

- Heterogeneities have the same nature. Only scaling gives difference between the true field and the one used in the computations.

Example 4: 4 Updates

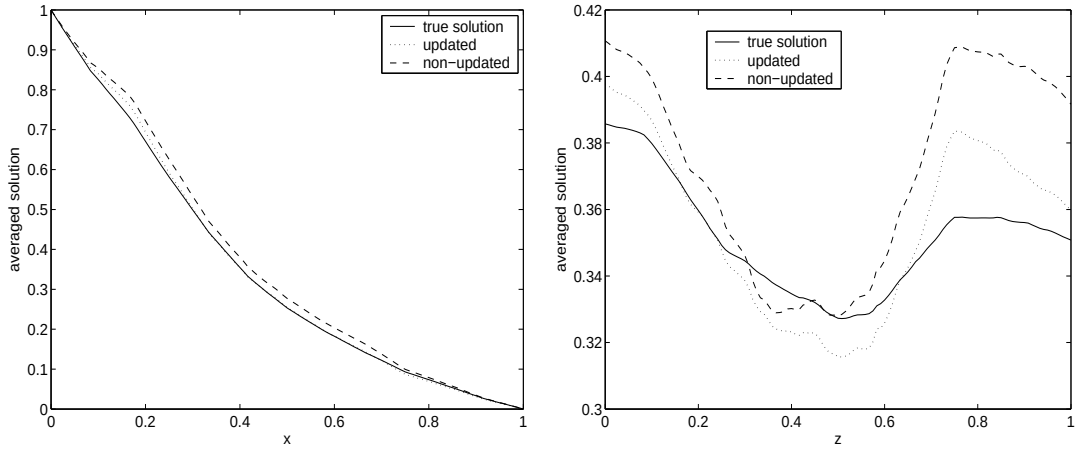

Figure 4: Comparisons of the average solutions across x and z directions. Solid line designates the true solution, dotted line designates the solution obtained using our simulations with 4 updates, and the dashed line designates the solution that has not been updated.

- As expected, more updating produces a more accurate DDDAS.

Conclusions

- Multiscale methodology works well and is inexpensive.
- Useful for correcting simulations' past and future predictions and leads to far fewer restarts.
- Technique can be applied to many (time dependent, nonlinear) PDE formulations used in the DDDAS field.

Request: Please send URLs (and NSF award numbers if you have them) for DDDAS projects to douglas-craig@cs.yale.edu for inclusion in the new http://www.dddas.org. Thanks!