Virtual Telemetry for Dynamic Data-Driven Application Simulations

Craig C. Douglas[1,2], Yalchin Efendiev[3], Richard Ewing[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
{douglas}@ccs.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, College Station, TX, USA
{efendiev,lazarov}@math.tamu.edu and richard-ewing@tamu.edu
[4] Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT, USA
mjc@sci.utah.edu and crj@cs.utah.edu

Abstract. We describe a virtual telemetry system that allows us to devise and augment dynamic data-driven application simulations (DDDAS). Virtual telemetry has the advantage that it is inexpensive to produce from real time simulations and readily transmittable using open source streaming software. Real telemetry is usually expensive to receive (if it is even available long term), tends to be messy, comes in no particular order, and can be incomplete or erroneous due to transmission problems or sensor malfunction. We will generate multiple streams continuously for extended periods (e.g., months or years): clean data, somewhat error prone data, and quite lossy or inaccurate data. By studying all of the streams at once we will be able to devise DDDAS components useful in predictive contaminant modeling.

1 Introduction

Consider an extreme example of a disaster scenario in which a major waste spill occurs in a river flowing through the center of a major city. In short time, the waste will be on kilometers of the city's shoreline.

Sensors can now be dropped into an open water body to measure where the contamination is, where the contaminant is going to go, and to monitor the environmental impact of the spill. Whether or not the procedure to drop the sensors exists today is not relevant for the moment; only that it could be done. Scrambling aircraft to drop sensors in a river is no longer such a far fetched scenario.

A well designed DDDAS predictive contaminant tracking program will be able to determine where the flow will go, the neighborhoods that have to be

evacuated, and optimize a clean-up plan. A backward in time DDDAS program will offer help in determining where and when a contaminant entered the environment. For this to become reality data streaming tolerant algorithms with sensitivity analysis incorporated for data injection, feature extraction, and multiple scaling must be developed for this to scenario to become a reality.

There are several approaches to designing DDDAS systems for disaster management. One is to collect real data and replay it while developing new data dynamic algorithms. Another is to generate fictional data continuously over a several year period with disasters initiated at random times or by human intervention (possibly without informing the other researchers in advance).

In either case, data streaming is a useful delivery system. Like realistic telemetry, data arrives that is usually incomplete, not in any order, and occasionally wrong. In short, it is an appalling mess.

In Sect. 2, we define an application that is our first test case. In Sect. 3, we define the DDDAS components that are of interest in this paper. In Sect. 4, we define model reduction for our application. In Sect. 5, we describe the telemetry middleware that we are in the process of developing and what our choices are. In Sect. 6, we discuss the programming environment that we are using to generate useful prototypes quickly so that a final package, useful to others as well, can be built. In Sect. 8, we draw some conclusions.

2 An Example Application

As an example application we consider a single component contaminant transport in heterogeneous porous media taking into account convection and diffusion effects. This simple model will be further extended by incorporating additional physical effects as well as uncertainties. The mathematical formulation of the problem is given by coupled equations that describe pressure distribution and the transport equations, $\nabla \cdot \mathbf{k} \nabla p = f$, $S_t + \mathbf{v} \cdot \nabla S = \nabla \cdot D \nabla S$, $\mathbf{v} = -\mathbf{k} \nabla p$. We consider two different permeability field scenarios. For the first case we assume that a large horizontal permeability streak is located in the middle of the domain and there are two vertical low permeability zones. The background permeability is taken to be 1, the permeability of high streak region is taken to be 2, and vertical low permeability regions have permeability 0.01. The second permeability field is chosen as an unconditional realization of a fractal field whose semivariogram is given by $\gamma(h) = Ch^{0.5}$, where the semivariogram is defined as

$$\gamma(\mathbf{h}) = \frac{1}{2} E[(\xi(\mathbf{x} + \mathbf{h}) - \xi(\mathbf{x}))^2].$$

A horizontal high permeability streak is added at the center of the domain. To generate a realization of this field we use fractional Brownian motion theory developed in [1, 7, 15]. The realization of the field is generated from the generalized Weierstrass-Mandelbrot function with fractal co-dimension $\beta/2$ ($\beta = 0.5$)

$$\xi(\mathbf{x}) = \sum_{i=1}^{\infty} A_i \lambda^{-i\frac{\beta}{2}} \sin(\lambda^i \mathbf{x} \cdot \mathbf{e}_i + \phi_i),$$

where A_i are normally distributed random numbers with mean zero and variance 1, $\mathbf{e}_i$ are uniformly distributed on the unit sphere, ϕ_i are uniformly distributed over $[0, 2\pi]$, and $\lambda > 1$. This isotropic random field has an approximately power semivariogram, i.e., $0 < ch^\beta < \gamma(h) < Ch^\beta$. In all the examples the rectangular domain $[0, 2] \times [0, 1] \times [0, 1]$ is consider and the following initial and boundary conditions are imposed. We assume the pressure at the inlet $x = 0$ to be $p = 1$ and $p = 0$ at the outlet $x = 2$. For the other faces we assume no flow boundary conditions for the pressure equation. For the concentration field we assume that the concentration is $S = 1$ at the inlet $x = 0$ and $\partial S/\partial n = 0$ is imposed on the other faces. Furthermore, $D = 0.02$ is taken to be constant. The computations are implemented using 3-D finite element simulator developed at Texas A&M employing the mesh generator NETGEN [16].

The concentration fields cannot easily be rendered in a grayscale. We have decided to post the concentrations on the web at URL

$$http://www.dddas.org/itr-0219627/papers.html\#iccs2003.$$

3 DDDAS Components

In contaminant movement predictions, it is common to run simulations for a few hours or days as a batch process. Although the individual application simulation periods are a few wall clock hours, they are built to update the early time steps with real data as it becomes available.

Converting software from a data set oriented batch code to a data stream continuously running code requires significant, nontrivial changes. We are dissecting contaminant transport models and associated codes in order to understand how to convert these models into the DDDAS model. Underground situations (e.g., nuclear waste contamination from a leaky containment vessel) are different from an above ground situation (contaminants in a lake and/or river). We are investigating both of these situations as well as the combination of the two.

The addition of DDDAS features to a system requires that aspects related to the initial data must be identified. Additionally, the time-related updating procedures must investigated. As we approach these requirements we will focus on maintaining a solution that is generally useful to other application fields that are initial boundary value problem (IBVP) based. Almost all IBVP formulations preclude dynamic data or backward in time error checking, sometimes in quite subtle ways.

The use of continuous data streams instead of initial guess only data sets presents an additional challenge for data driven simulations since the results vary based on the sampling rate and the discretization scheme used. Dynamic data assimilation or interpolation might be necessary to provide a feedback to experimental design/control. DDDAS algorithms also need to dynamically assimilate new data at mid-simulation as the data arrives, necessitating "warm restart" capabilities.

Modifying application programs to incorporate new dynamically injected data is not a small change in the application program, particularly since the

incoming data tends to be quite messy. It requires a change in the application design, the underlying solution algorithms, and the way people think about the accuracy of the predictions.

Uncertainties in DDDAS applications emanate from several sources, namely uncertainty associated with the model, uncertainties in the input data (streamed), and the environment variables. Identifying the factors that have the greatest impact on the uncertainty output of the calculations is essential in order to control the overall processes within specific limits. Handling all output distributions to provide error bounds is, for most realistic problems, a computationally prohibitive task. Hence, using prior observations to guide the output distribution estimations presents a possible approach to incorporating uncertainty in control decisions.

Incorporating these statistical errors (estimations or experimental data uncertainties) into computations, particularly for coupled nonlinear systems, is difficult. This is compounded by the fact that tolerances may also change adaptively during a simulation. Error ranges for uncertainty in the data must be created and analyzed during simulations. Sensitivity analysis must be performed continuously during simulations with options in case of a statistical anomaly.

The common mathematical model in many DDDAS applications may be formulated as solving a time dependent, nonlinear problem of the form $F(x+Dx(t))=0$, by iteratively choosing a new approximate solution x based on the time dependent perturbation $Dx(t)$.

At each iterative step, the following three issues may need to be addressed. Incomplete solves of a sequence of related models must be understood. In addition the effects of perturbations, either in the data and/or the model, need to be resolved and kept within acceptable limits. Finally, nontraditional convergence issues have to be understood and resolved. Consequently, there will be a high premium on developing quick approximate direction choices, such as, lower rank updates and continuation methods. It will also be critical to understand the behavior of these chosen methods.

By generating telemetry in real time, we allow for the new, DDDAS code to determine how well it has predicted the past, assuming the DDDAS code runs much faster than real time. We run a simulation backwards in time to a point where we can verify that we have not introduced too great an error into a simulation at a future time. This is particularly important when deciding whether or not to introduce all or just part of a data stream. For example, if a data stream update causes a loss or addition of mass when it is conserved, the data stream update will lead to an abrupt loss of usefulness unless a filtering process is developed to maintain the conservation of mass. We are developing new filters for our applications that resolve the errors introduced by converting the applications to data streams.

4 Model reduction and multiscale computations

Model reduction will largely be accomplished with the use of upscaling techniques. Due to complicated interactions and many scales, as well as uncertainties, upscaling is desirable. The upscaling is in general nontrivial because heterogeneities at all scales have a significant effect, and these effects must be captured in the coarsened subsurface description. Most approaches for upscaling are designed to generate a coarse grid description of the process which is nearly equivalent (for purposes of flow simulation) to an underlying fine grid description. We will employ both static and dynamic upscaling techniques. Static upscaling techniques will generally be used in coarsening the media properties, while the dynamic upscaling will be employed to understand the effect of the small scale dynamics on the larger scales. One of the important aspects of our upscaling approach is the use of carefully selected dynamic coarse scale variables in a multiscale framework.

To demonstrate the main idea of our static upscaling procedures we consider our application model, the pressure equation $\nabla \cdot \mathbf{k} \nabla p = f$. Within this approach the heterogeneities of the media are incorporated into the finite element (or finite volume element) base functions that are further coupled through the global formulation of the problem We seek the solution of this equation in a coarse space whose base elements $\phi_i(\mathbf{x})$ contain the local heterogeneity information, $V_h = span(\phi_i)$. The numerical solution u_h is found from

$$\int_D \mathbf{k} \nabla u_h \nabla v_h d\mathbf{x} = \int_D f v_h d\mathbf{x}, \ \forall v_h \in V_h.$$

This approach allows us to solve the saturation equation on the on the fine scale grid as well as on the coarse scale grid since the fine scale features of the velocity field can be recovered from the base functions. Consequently, we can adjust the permeability at the fine scale directly. One of the advantages of these approaches is that the local changes of the permeability field will only affect few base functions. Consequently, we will only need to re-compute few base functions and solve the coarse problem. This methodology will be useful in backward integration for finding sources of inconsistencies within our DDDAS application. We will also employ traditional approaches based on upscaling of permeability field [2] and use mainly the upscaling of permeability field with oversampling techniques [19]. The main idea of this approach is to use larger domain (larger than the coarse block) in order to reduce the boundary effects. To reduce the grid effects grid based upscaling techniques will be employed.

For the time dependent transport problems we will employ coarsening techniques that are based on dynamic upscaling. For these approaches the dynamic quantities (e.g., quantities that depend on concentration) are coarsened and their functionality is determined through analytical considerations. Determining the form of coarse scale equations is important for multiscale modeling. Our previous approaches on this issue were mainly in two directions. First approach that is based on perturbation techniques models subgrid effects as a nonlocal macrodispersion term [5]. This approach takes into account the long range interaction in

the form of diffusion term that grows in time. Our second approach based on homogenization techniques [6] computes the dynamic upscaled quantities using local problems [4, 3].

5 Telemetry Middleware

Real telemetry is usually expensive to receive (if it is even available on a long term basis), tends to be messy, comes in no particular order, and can be incomplete or erroneous due to transmission problems or sensor malfunction. For predictive contaminant telemetry, there are added problems that due to pesky legal reasons (corporation X does not want it known that it is poisoning the well water of a town), the actual data streams are not available to researchers, even ones providing the simulation software that will do the tracking.

Virtual telemetry has the advantage that it is inexpensive to produce from real time simulations. The fake telemetry can easily be transmitted using open source streaming software.

We will generate multiple streams continuously for extended periods (e.g., months or years): clean data, somewhat error prone data, and quite lossy or inaccurate data. By studying all of the streams at once we will be able to devise DDDAS components useful in predictive contaminant modeling.

Real telemetry used in predictive contaminant monitoring comes in small packets from sensors in wells or placed in an open body of water. There may be a few sensors or many. With virtual telemetry, we can vary the amount of telemetry that we broadcast and its frequency.

There are a number of issues that are being resolved in the course of our project.

1. Should the telemetry data be broadcast as an audio stream?
2. Should the telemetry data be broadcast as a movie stream?
3. Should a complete 3D visualization be broadcast (and in what form)?
4. Should only sparse data from discrete points in a domain be broadcast?
5. At what rate can the virtual telemetry be broadcast so that network administrators do not cut off the data stream?

We are building middleware to answer all of the questions above.

There are a number of open source projects for doing reliable data streaming. Palantir [8] is a former commercial product that has been re-released as open source. It supports audio and video streaming as well as general data streaming. Gini [9] is an audio and video streamer that is still in beta, but is fairly active in development. GStreamer [10] is more of a streaming framework developed by graduate students in Washington. It supports audio, video, and general data streaming. VideoLAN [18] is a video streamer developed by students at the École Centrale Paris. QuickTime Streaming Server is an audio and video streaming server that Apple offers. Of the five, GStreamer is the clear first choice to concentrate on.

Broadcasting the telemetry as audio has the advantage that there are many programs to choose from to generate the data streams and to "listen" to them on the receiving end.

Broadcasting the telemetry as a movie stream or a full 3D visualization has the advantage that it can be trivially visualized on the receiving end. This is particularly attractive when studying incomplete and/or erroneous data streams. However, there is a potential of transmitting too much data and overwhelming the network.

Broadcasting only sparse data from discrete points in any form has pluses and minuses. Almost any Internet protocol can be used. However, do we really want to tie ourselves to one Internet protocol?

Avoiding the attention of network administrators is a serious concern. We must balance adequate data streams with not having any serious impact on a network. This is highly dependent on where we are running the virtual telemetry from and to and cannot easily be determined *a priori*. However, it is easily determined *a posteriori*, which will be part of a well behaved, adaptive broadcast system.

6 Program Construction

Current interactive scientific visualization and computational steering implementations require low latency and high bandwidth computation in the form of model generation, solvers, and visualization. Latency is particularly a problem when analyzing large data sets, constructing and rendering three-dimensional models and meshes, and allowing a scientist to alter the parameters of the computation interactively. However, large-scale computational models often exceed the system resources of a single machine, motivating closer investigation of meeting these same needs with a distributed computational environment.

To achieve execution speeds needed for interactive three-dimensional problem solving and visualization, we have developed the SCIRun problem solving environment and computational steering system [12, 11]. SCIRun allows the interactive construction, debugging, and steering of large scale scientific computations. SCIRun can be conceptualized as a computational workbench, in which a scientist can design via a dataflow programming model and modify simulations interactively. SCIRun enables scientists to interactively modify geometric models, change numerical parameters and boundary conditions, and adaptively modify the meshes, while simultaneously visualizing intermediate and final simulation results.

When the user changes a parameter in any of the module user interfaces, the module is re-executed, and all changes are automatically propagated to all downstream modules. The user is freed from worrying about details of data dependencies and data file formats. The user can make changes without stopping the computation, thus steering the computational process. In a typical batch simulation mode, the scientist manually sets input parameters, computes results, assesses the results with a combination of separate analytical and visualization

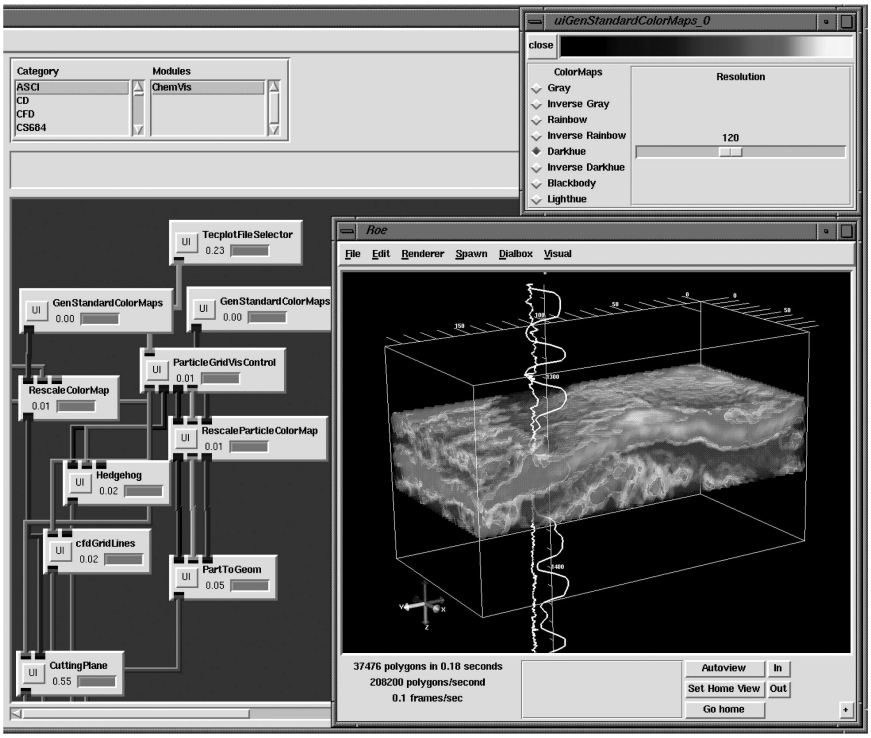

Fig. 1. SCIRun is open source software, and is freely downloadable.

packages, then iterates this procedure. SCIRun closes the loop and allows interactive steering of the design, computation, visualization, and analysis phases of a simulation[17].

It is important to have the ability to develop and manipulate the telemetry simulation interactively. An example of this need is with sensors. It can be imagined that the typical telemetry collection will combine many types of sensors reporting different data types and at different rates. The ability to quickly and seamlessly swap sensor types and dataflow types is critical to a DDDAS application. SCIRun provides the ability to develop rapid prototypes. The modular design of SCIRun allows the simulation to be built from a set of components, in such a system components or modules may be assigned to each sensor type making it easy to combine or change the sets of sensors used in a simulation. This modularity also enables a change in the scale or type of a problem being simulated to be done very quickly and directly.

Testing of the system and its robustness with simulated data will require the ability to manipulate the data stream either by adding data losses or abnormalities. Using SCIRun, the running telemetry stream can be manipulated by hand essentially as a data steering problem.

As the testbed is expanded it will almost certainly be a requirement that the application perform in a distributed environment. It will also be likely that the application be required to work, seamlessly, with other software packages or libraries, not directly incorporated in the application. There are numerous examples of the ease of bridging third party software with SCIRun, and of SCIRun operating in a distributed environment [14, 13] . This will enable the utilization of existing free codes for streaming the dynamic data needed to drive this simulation.

7 Aknowledgements

This work was supported in part by a National Science Foundation collaborative research grant (EIA-0219627, EIA-0218721, and EIA-0218229).

8 Conclusions

We have described issues in constructing a virtual telemetry system. Our target is enabling DDDAS components in a predictive contaminant model. We expect to have useful open source middleware readily available on the Internet soon. Once the middleware is completed, we can explore many interesting features of DDDAS.

References

1. Chu, J., and Journel, A. In *Geostatistics for the next Century*, R. Dimitrakopoulos, Ed. Kluwer Academic Publishers, 1994, pp. 407–412.
2. Durlofsky, L. J. Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media. *Water Resour. Res. 27* (1991), 699–708.
3. Efendiev, Y., and Durlofsky, L. Accurate subgrid models for two-phase flow in heterogeneous reservoirs. paper SPE 79680 presented at the 2003 SPE Symposium on Reservoir Simulation, Houston, TX.
4. Efendiev, Y., and Durlofsky, L. A generalized convection diffusion model for subgrid transport in porous media. submitted to SIAM on Multiscale Modeling and Simulation.
5. Efendiev, Y. R., Durlofsky, L. J., and Lee, S. H. Modeling of subgrid effects in coarse scale simulations of transport in heterogeneous porous media. *Water Resour. Res. 36* (2000), 2031–2041.
6. Efendiev, Y. R., and Popov, B. On homogenization of nonlinear hyperbolic equations. submitted to SIAM J. Appl. Math. (available at http://www.math.tamu.edu/∼yalchin.efendiev/submit.html).
7. Falconer, K. *Fractal Geometry: Mathematical Foundations and Applications*. John Wiley & Sons, 1990.
8. FastPath Research. Palantir. http://www.fastpath.it/products/palantir.
9. Gini. Gini. http://gini.sourceforge.net.
10. GStreamer. Gstreamer. http://www.gstreamer.net.

11. Johnson, C., and Parker, S. The scirun parallel scientific computing problem solving environment. In *Ninth SIAM Conference on Parallel Processing forScientific Computing* (1999).
12. Johnson, C., Parker, S., and Weinstein, D. Large-scale computational science applications using the scirun problem solving environment. In *Supercomputer 2000* (2000).
13. Miller, M., Hansen, C., and Johnson, C. Simulation steering with scirun in a distributed environment. In *Applied Parallel Computing, 4th InternationalWorkshop, PARA'98*, E. E. B. Kagstrom, J. Dongarra and J. Wasniewski, Eds., vol. 1541 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1998, pp. 366–376.
14. Miller, M., Hansen, C., Parker, S., and Johnson, C. Simulation steering with scirun in a distributed memory environment. In *Seventh IEEE International Symposium on HighPerformance Distributed Computing (HPDC-7)* (jul 1998).
15. Oh, W. *Random field simulation and an application of kriging to image thresholding*. PhD thesis, State Univeristy of New York at Stony Brook, 1998.
16. Schoeberl, J. NETGEN-An advancing front 2d/3d-mesh generator based on abstract rules. *Computing and Visualization in Science 1* (1997), 41–52. http://www.sfb013.uni-linz.ac.at/ joachim/netgen/.
17. SCIRun: A Scientific Computing Problem Solving Environment. Scientific Computing and Imaging Institute (SCI), http://software.sci.utah.edu/scirun.html, 2002.
18. VideoLAN. Videolan. http://www.videolan.org.
19. Wu, X. H., Efendiev, Y. R., and Hou, T. Y. Analysis of upscaling absolute permeability. *Discrete and Continuous Dynamical Systems, Series B 2* (2002), 185–204.