A Note on Dynamic Data Driven Application Simulation (DDDAS) Using Virtual Telemetry[1]

Craig C. Douglas
University of Kentucky, Computer Science Dept., 325 McVey Hall - CCS, Lexington, KY 40506-0045, USA
and Yale University, Computer Science Dept., P.O. 8285, New Haven, CT 06520-8285, USA
craig.douglas@yale.edu

Chad E. Shannon (University of Kentucky)
Yalchin Efendiev, Richard E. Ewing, Victor Ginting, and Raytcho Lazarov (Texas A&M)
Martin J. Cole, Greg Jones, Christopher R. Johnson, and Jennifer Simpson (University of Utah)

1 Introduction

We have immense computing power available at the national 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 continuous flood of information. This is a substantial barrier to achieving the potential benefit computational science can deliver to many application areas including contaminant tracking, which is the target application area of this project.

To address this current state we have identified four relatively diverse areas that have common issues that must be addressed for dynamic data driven application simulation (DDDAS) informational and computational sciences to have the promised impact toward addressing important problems. These issues include:

- Effectively assimilating continuous streams of data into running simulations. These data streams most often will be
  - noisy but with known statistics.
  - received from a large number of scattered remote locations and must therefore be assimilated to a usable computational grid.
  - missing bits or transmission packets, as for example is the case in wireless transmissions.
  - injecting dynamic and unexpected data input into the model.
  - limited to providing information only at specific scales, specific to each sensor type.
- Warm restarting simulations by incorporation of the new data into parallel or distributed computations, which require the data but are sensitive to communication speeds and data quality.
- Tracking and steering of remote distributed simulations to efficiently interact with the computations and to collaborate with other researchers.
- Components to assist researchers in their interpretation and analysis of collections of simulations. This will include designing and creating an application program interface and middleware.

Sensors and data generating devices may take many forms, even, for example, other running computational simulations. The intent of this proposal is to directly address DDDAS issues in the context of a specific application area in order to provide techniques and tools to effectively demonstrate the potential of dynamic data driven simulations for other areas.

The 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. For efficient solution of this class of problems we need: (a) accurate, fast, and locally

[1] This research was supported in part by National Science Foundation grants EIA-0219627, EIA-0218229, and EIA-0218721.

conservative approximation methods and (b) parallel adaptive methods that are dynamic in time. We shall solve these challenge problems using distributed computer systems (discussed above) and the latest developments in Eulerian-Lagrange localized adjoint method, discontinuous Galerkin method and/or streamline diffusion method in concert with domain decomposition and adaptive grid refinement techniques.

Many 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 traditional operating mode for most CFD applications is a static initialization with fixed forcing and boundary conditions followed by a limited exploration of the parameter space. This is clearly inadequate for many long term simulations, particularly when advances in observations capabilities, data assimilation techniques, and computers and networking can be leveraged to determine an optimal enough set of parameters needed for accurate and realistic forecasts.

DDDAS endows applications with dynamic data input capabilities by coupling the model and algorithms to the observations. The ultimate aim is to leverage the current state of the art in computing, networking, and observational instruments to produce a more realistic and accurate depiction of the state of a system than can be derived using either model or observations alone. We stress the fact that continuous data streams from observational instruments and sensors call for a radical change in model philosophy from static to dynamic data input.

Several hurdles stand in the way of achieving such an integrated, dynamically driven modeling system. These hurdles can be classified as data quality and management, computing and networking power, data assimilation and modeling algorithms, visualization, and hardware requirements.

A valid question is why do DDDAS in the first place? Many applications (e.g., 7 day weather forecasting) run fast enough already on parallel supercomputers. Starting a new simulation every time new data is available is then reasonable. Some situations warrant a different approach. Major disaster simulation (e.g., nuclear waste dispersion) is one. In this situation, having access to a large parallel supercomputer is not a given. A set of WiFi connected laptops is much more likely. While current laptops have the computing power of a 1990 vintage Cray processor, this is insufficient for the simulations envisioned in this project.

For example, sensors can usually be placed quickly above ground quickly. Underground tracking is much harder and more time consuming since wells usually have to be installed. Data will come in at significant rate and have to be processed. In the laptop environment, calculations will probably need to be interrupted and have new data inserted as it becomes available. Even how data can be assimilated and how much is needed or usable are issues that must be addressed.

2 Contaminant Tracking Using SCIRun

Our main research to date has concentrated on the following:

- Development of software within SCIRun simulator [13, 15].
- Development of general virtual telemetry broadcasting with new SCIRun modules for receiving using streaming data techniques and a new data format similar to MP3 streaming.
- Advanced computational as well as multiscale techniques related to multi-component porous media flows.

For the software development we use new or improved modules and interfaces of SCIRun to implement various numerical methods for porous media flows. We now have several simulators that work both for

rectangular as well as on general three dimensional unstructured grids. A finite volume element framework is utilized since this method maintains numerical conservation of flux. Eventually we will use the mesh template mechanism from SCIRun, but additional basis function types and finite elements must be written for SCIRun.

Given a domain, the mesh generator NETGEN [17] is used to discretize the domain into a collection of tetrahedral. We wrote a C++ code that serves as an interface between NETGEN and SCIRun so that all mesh information required by the finite volume element algorithm can be accessed conveniently through this interface.

We also wrote a code to solve a time-dependent transport equation on the same grid setting. The flux computed from a much older, static data driven code is used as one of the inputs through virtual telemetry generated at another site. An upwind scheme is applied to resolve the transport part, which is then combined with an explicit time integration to obtain the transport quantity at the next time level.

Our first application is 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 over the course of the project.

The mathematical formulation of the problem is given by coupled equations that describe pressure distribution and the transport equations. The pressure field is described by the elliptic (or parabolic, in the presence of compressibility) equation and the transport of components is described by a convection-diffusion equation that is dominated by convection effects.

The point of this example is to capture the effects of the heterogeneities. We have used various heterogeneous permeability fields in our simulations. These fields are generated using GSLIB libraries [3]. The heterogeneities are typically chosen to have large correlation features in the horizontal direction. Due to the presence of these heterogeneous features we expect irregular flow behavior, e.g., the contaminant can be transported faster in some regions while slower in others.

We now present some representative simulation results. In all cases the global domain is a $5\times1\times1$ box. The fine scale models are of dimension $40 \times 40 \times 40$ and are geostatistical realizations of unconditioned, log-normally distributed permeability fields with prescribed variance σ^2 (σ^2 here refers to the variance of $\log k$) and correlation structure. The correlation structure is specified in terms of dimensionless correlation lengths in the x- and z-directions, l_x and l_z, nondimensionalized by the system length. The realizations were generated using GSLIB algorithms [3] using a spherical variogram model. Simulation results are presented for the contaminant concentration at certain dimensionless time defined by pore volume injected (PVI). Let V_p be the total pore volume of the system. Then the PVI is defined by

$$\frac{1}{V_p}\int_0^t q_t(\tau)d\tau,$$

provides the dimensionless time for the displacement.

To illustrate the three dimensional scalar fields, we depict some cross sections of the field. The cross sections for three values of y as well as the cross sections for two values of x and z are depicted. The slices along the y direction show the anisotropic effects induced by the long range features of the permeability fields along the x direction, while the cross sections for fixed values of x and z are designed to complement a 3-D view of the scalar field.

Advanced computational tools, methods, and algorithms for porous media flows, directly related to this project, have also been of major research interest and efforts. We have worked in two main directions, namely, computational techniques and multiscale methods related to multi-component porous media flows and adaptive methods for general transport and diffusion equations.

3 Virtual Telemetry

Data that is transmitted through a telecommunications system is commonly referred to as telemetry. The transmission media can be one or more of land lines, underwater lines, microwave, or satellite based. There

is both latency and broadcast time based on distance and resistance in the physical media that determines how long the data takes to get to the receiver.

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 trivially vary the amount of telemetry that we sample and its frequency.

Real telemetry based on high resolution photographs from space on a slow space to land transmission system can be quite a challenge, however, but we are not dealing with this situation presently.

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 and can easily be transmitted using modified forms of 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.

The basic technique that we are using to take an old, robust code that uses only static input dat and makes long term predictions with the capability of getting the transient data throughout the simulation. Instead of trying to run the old code very, very fast, we want to run it in real time using small time steps for all components. We can run the old code on a fast, cheap PC or small parallel computer depending on how much data we generate per time step. The code sleeps most of the time while waiting for the wall clock to catch up. Our sample time steps are one minute long up to a few hours.

We are using a 3D tensor product mesh with finite differences. The code is conservative, which is essential in the situation we are interested in.

We use data from a small subset of computed values and assume that a sensor is placed there. The location, time, and a few pieces of floating point data are all that have to be transmitted on a per sensor time dependent basis.

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, in either case, there is a potential of transmitting too much data and overwhelming the network. Worse yet, there is the real possibility that the recording or motion picture industries may erroneously believe that the data is pirated music or film material and cause serious legal problems.

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 is be part of a well behaved, adaptive broadcast system.

Initially, we assumed that we could use one of the audio formats that claims to be data lossless (e.g., the Ogg audio format with the FLAC encoder). Unfortunately, we discovered through a comprehensive search that floating point data was never considered by the designers of audio formats nor the authors of encoders and decoders for audio streaming. The video streaming field is much more primitive than audio and it turned out to be an even less useful area to investigate.

Eventually, we decided to place MP3 data headers around the telemetry data. A small program was written that takes sample data and produces an output file containing the same data only with mp3 headers placed where they need to be. This is the basis of the CH3 encoding we developed. Nothing is really encoded in the sense of compression but this aspect is discussed later in this section.

The actual MP3 is 4 bytes in size. The first 12 bits compromise the syncword used to synchronize data transfer. The 14th and 15th bit are always set to 01 to represent Layer 3 data. In the third byte, the bit

rate and sampling rate are stored by a code that is an index into a table. The fourth byte specifies whether the data is stereo or mono.

We use mono and do not interlace the data, as is an option. Many of the bits in the MP3 header can be used for other things of interest to us, but saving a bit here and there makes no sense and offers a possible serious bug if the MP3 header format changes in the future.

We decided that we had to use Open Source software since we determined during the project that we would have to modify both the streaming code and the receiving client in order to eliminate unwanted data compression techniques aimed at integer data.

MP3 normally uses Huffman encoding. Many encoders also implement filters based on knowing which audio or visual frequencies humans cannot hear or see, respectively. Since the data is integer based, the values out of range are zeroed out. Imagine a floating point number whose exponent has been zeroed (e.g., 1.2×10^{31} might become just 1.2, a small, but noticeable error in data transmission).

We are now using a modified version of the Gini streaming server [16]. Gini must be modified since it checks the data to see if it corresponds to legitimate MP3 audio data. Our Ch3 format data fails this criteria. The fix is to comment out a few lines of code in one file (mp3.c) of the streamer.

Besides having good enough headers, as already described, we have to generate playlists on disk that Gini uses to stream the data as an audio stream. Part of the overall process is to generate new playlists and getting Gini to "warm restart" so that it rereads the playlist.

Receiving the data as audio is a function of modifying an Open Source MP3 client like XMMS [1]. We added the CH3 format to produce a new version of XMMS which we call xccs. We still see junk data coming through the Internet, which is eliminated by using a socket with the BLOCKING property, which unfortunately offers the possibility of deadlocking the client eventually. Since the junk occurs only on initialization, we are investigating other ways of initially flushing the socket in order to have truly asynchronous transmission, which will make the connection more robust.

4 Summary of Results to Date

In [12] we have developed a new multiscale computational approach for multi-phase flow that is applicable for multi-component single phase flow. The latter is currently under investigation, which will be further extended to a multi-phase multi-component scenario. In [12] multiscale finite volume techniques along with a perturbation argument are used to upscale porous media flows, such as single-phase and two-phase immiscible flows, Richards' equations. In [11] analysis of our multiscale finite volume element method is presented.

The papers [7, 8, 9, 10] are dedicated to the construction and analysis of novel multiscale methods for nonlinear elliptic and parabolic equations. In these papers we systematically investigate our new approach showing that it can handle various nonlinearities in the homogenization process. The applications of this work to single and two-phase flows are presented in [5, 6], where we have proposed a generalized convection-diffusion approach for up-scaling porous media flows. The methods are tested successfully for both single and two phase flows in heterogeneous porous media.

In [2, 14] we have developed, theoretically studied, implemented, and tested adaptive finite volume methods for transport equations in general domains covered by unstructured tetrahedral meshes. We have established the reliability and the efficiency of the schemes and tested them on computing the concentration of passive chemicals in heterogeneous aquifers.

In cite [4] we describe our initial views of how to take an application that uses static, initial data and use it to generate data in a manner similar to real data generated by sensors in the field, so-called virtual telemetry, for new DDDAS enabled codes. This provided a framework for all three groups to produce the new codes that use telemetry-like data for injection into the data streams.

References

[1] P. Alm, T. Nilsson, O. Hallnas, and H. Kvalen. XMMS - X multimedia system: A cross platform multimedia player. http://www.xmms.org, 2003.

[2] C. Carstensen, R.D. Lazarov, and S.Z. Tomov. Explicit and averaging a posteriori error estimates for adaptive finite volume methods. *Preprint, Isaac Newton Inst. Math. Sci.*, 2003. (available at http://www.newton.cam.ac.uk/preprints/NI03010.pdf).

[3] C. V. Deutsch and A. G. Journel. *GSLIB: Geostatistical software library and user's guide, 2nd edition.* Oxford University Press, New York, 1998.

[4] C. C. Douglas, Y. Efendiev, R. Ewing, R. Lazarov, M. R. Cole, C. R. Johnson, and G. Jones. Virtual telemetry middleware for DDDAS. In *Computational Sciences - ICCS 2003*, volume 4, pages 279–288. 2003.

[5] Y. Efendiev and L. Durlofsky. Accurate subgrid models for two-phase flow in heterogeneous reservoirs. SPE 79680, presented at the SPE Reservoir Simulation Symposium held in Houston, Texas, February 3-5, 2003.

[6] Y. Efendiev and L. Durlofsky. Generalized convection-diffusion model for subgrid transport in porous media. *SIAM Multiscale Modeling and Simulation*, 1(3):504–526, 2003.

[7] Y. Efendiev and A. Pankov. Homogenization of nonlinear random parabolic operators. submitted to EJDE (available at http://www.math.tamu.edu/∼yalchin.efendiev/ep-hom-parab.ps).

[8] Y. Efendiev and A. Pankov. Meyers type estimates for approximate solutions of nonlinear elliptic equations and their applications. submitted to Num.Math.

[9] Y. Efendiev and A. Pankov. Numerical homogenization of nonlinear random elliptic operators. *SIAM Multiscale Modeling and Simulation.* submitted.

[10] Y. Efendiev and A. Pankov. Numerical homogenization of nonlinear random parabolic operators. submitted to SIAM MMS, (available at http://www.math.tamu.edu/∼yalchin.efendiev/ep-num-hom-parab.ps).

[11] V. Ginting. Analysis of two-scale finite volume element for elliptic problem. to be submitted.

[12] V. Ginting, R. Ewing, Y. Efendiev, and R. Lazarov. Upscaled modeling for multiphase flow. submitted to Journal of Computational and Applied Mathematics.

[13] C. R. Johnson, S. Parker, D. Weinstein, and S. Heffernan. Component-based problem solving environments for large-scale scientific computing. *J. Concurrency and Computation: Practice and Experience*, 14:1337–1349, 2002.

[14] R.D. Lazarov and S.Z. Tomov. A posteriori error estimates for finite volume element approximations of convection-diffusion-reaction equations. *Comput. Geosciences*, 6:483–503, 2002.

[15] S. G. Parker, D. Beazley, and C. R. Johnson. Computational steering software systems and strategies. *IEEE Computational Science and Engineering*, 4:50–59, 1997.

[16] K. Pifko, B. Dakay, and T. Szerb. Gini. http://gini.sourceforge.net, 2003.

[17] J. Schöberl. Netgen - an advancing front 2d/3d-mesh generator based on abstract rules. *Comput.Visual.Sci*, 1:41–52, 1997.