

Visualizing Multiscale, Multiphysics Simulation Data: Brain Blood Flow

Joseph A. Insley*
Argonne National Laboratory

Leopold Grinberg†
Brown University

Michael E. Papka‡
Argonne National Laboratory

ABSTRACT

Accurately predicting many physical and biological systems requires modeling interactions of macroscopic and microscopic events. This results in large and heterogeneous data sets on vastly differing scales, both physical and temporal. The ability to use a single integrated tool for the visualization of multiscale simulation data is important to understanding the effects that events at one scale have on events in the other. In the case of blood flow, we examine how the large-scale flow patterns influence blood cell behavior.

In this paper we describe the visualization tools that were developed for data from coupled continuum - atomistic simulations. Specifically, we overview a) a custom ParaView reader plug-in that processes macro-scale continuum data computed by a high-order spectral element solver; and b) an adaptive proper orthogonal decomposition-based technique for the visualization of non-stationary velocity fields from atomistic simulations. We also discuss how the ParaView parallel processing and rendering infrastructure was leveraged in the new tools. We apply our methods to visualize multiscale data from coupled continuum-atomistic simulations of blood flow in a patient-specific cerebrovasculature with a brain aneurysm.

Index Terms: J.2 [Computer Applications]: Physical Sciences and Engineering—Physics; I.6 [Computing Methodologies]: Simulation and Modeling—applications, simulation output analysis; I.3.8 [Computer Applications]: Computer Graphics—Applications

1 INTRODUCTION

Interfacing atomistic-based with continuum-based models is now required to simulate realistic multiscale physical and biological systems [5]. Multiscale modeling is crucial in many disciplines, for example, to tune the properties of smart materials, to probe the function of living cells and organisms, and to predict the dynamics and interactions of rarefied plasmas with dense plasmas [1]. This code-and model-coupling pose a number of challenges, including stable interface conditions, communicating data between various components of the coupled solver, and the analysis of heterogeneous data. The focus of this paper is on visualization of heterogeneous and very large data sets produced by multiscale, multiphysics simulations of unsteady flow. Specifically, we consider the simulation of a blood clot (thrombus) formation at the wall of an aneurysm (see Figure 2), which is part of a larger arterial network. We employ a coupled solver resolving the macroscale flow features by applying a continuum model and microscale features by applying an atomistic approach [5]. The solver is based on coupling two open source parallel codes: DPD-LAMMPS (a dissipative particle dynamics version of the Large-scale Atomic/Molecular Massively Parallel Simulator [2], modified at Brown University) and NekTar - a high-order spectral/*hp* element library. The data produced by each

solver represent either continuum or atomistic fields, and it is saved in separate files with formats specific to each solver. The size of each file depends on the resolution employed by each solver; in the case considered here the number of degrees of freedom required by NekTar was about three billion, while the number of particles in the atomistic domain was about 800 million. Simulation of half of a cardiac cycle took 24 hours on 131,072 cores of a BlueGene/P, and, due to flow unsteadiness, the solution was saved at intermediate states by each solver in over a thousand files. To efficiently handle the visualization of the continuum data we have developed a parallel, coupled ParaView-NekTar code.

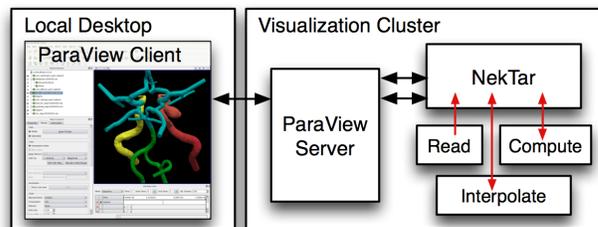


Figure 1: This figure shows the coupling of ParaView with our custom NekTar reader plug-in.

This software is comprised from two major components: 1) ParaView for parallel visualization algorithms and image rendering; and 2) NekTar for parallel data processing. One of the key features of this software is interactive and multi-resolution data visualization. The user accesses NekTar’s utilities via the ParaView GUI, as schematically illustrated in Figure 1. To analyze and visualize the *collective* motion of atoms we implement the adaptive proper orthogonal decomposition, which maps the atomistic data to the continuum. We visualize platelet deposition (the major outcome of the multiscale simulation) by presenting particles instantaneous positions at various times.

2 MACROSCALE SIMULATION DATA

For spatial discretization NekTar employs the spectral/*hp* element method (SEM) [6]. The computational domain Ω is decomposed into a set of polymorphic non-overlapping elements $\Omega^{e_i} \subset \Omega$, $i = 1, \dots, Nel$, as illustrated in Figure 3. Within each element the solution $u^e(t, \mathbf{x})$ is approximated in terms of hierarchical, mixed-order, semi-orthogonal Jacobi polynomial expansions [6]

$$u^e(t, \mathbf{x}) = \sum_k \bar{u}_k^e(t) \phi_k^e(\mathbf{x}), \quad (1)$$

where \bar{u}_k^e is an amplitude of the k -th shape function $\phi_k^e(\mathbf{x})$ in the element e . The solution computed by NekTar is saved in the modal space, that is it contains the values of \bar{u}_k^e for each element. To visualize the continuum data at different levels of granularity it is possible to interpolate the solution using formula (1) to different number of points \mathbf{x} .

To achieve high parallel efficiency, NekTar employs a multi-patch approach, where the computational domain is divided into smaller overlapped patches (see Figure 2) [4]. The continuum

*e-mail:insley@mcs.anl.gov

†e-mail:lgrinb@dam.brown.edu

‡e-mail:papka@anl.gov

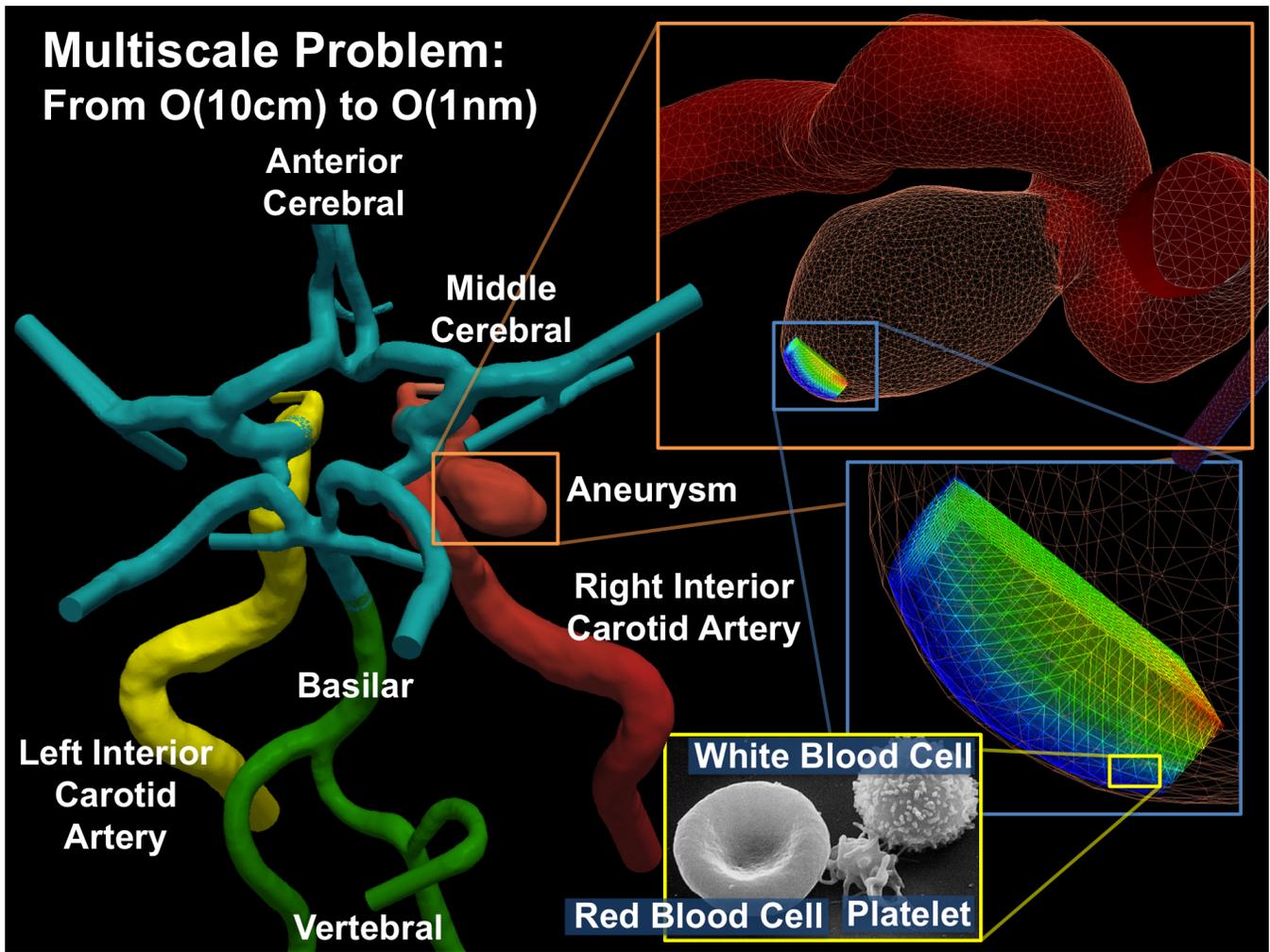


Figure 2: Blood flow in the brain is a multiscale problem. Shown on the left is the macrodomain where the spectral element equations are solved; different colors correspond to different computational patches. Shown in the inset (right), located next to the arterial wall of the aneurysm, is the microdomain of 3.93mm^3 where dissipative particle dynamics is applied. Of interest in the present paper is the deposition of platelets to the aneurysmal wall.

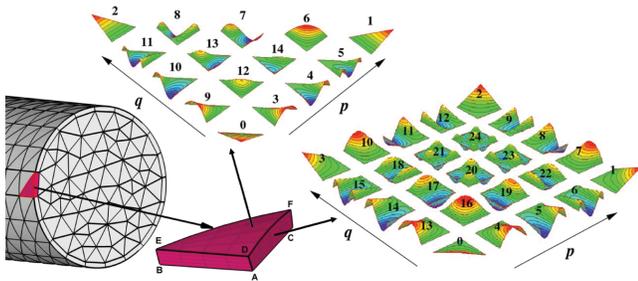


Figure 3: Illustration of the unstructured surface grid and the polynomial basis employed in NekTar. The solution domain (patch) is decomposed into non-overlapping elements. Within each element the solution is approximated by vertex, edge, face and (in 3D) interior modes. The shape functions associated with the vertex, edge and face modes for fourth-order polynomial expansion defined on triangular and quadrilateral elements are shown in color.

solver is applied in each patch separately, the solution continuity is achieved by imposing proper interface conditions. The multi-patch approach allows the solution data for each patch to be saved and visualized separately; moreover, when visualizing the entire solution, different resolutions can be applied to each patch. Such an approach results in greater flexibility and computational savings, since high resolution data is generated only in the region of greatest interest.

2.1 Data description

In NekTar the solution data and geometry are stored separately in MESH and SOLUTION files. The MESH file contains coordinates of elements, connectivity, boundary condition information, as well as information on the curved boundaries. The SOLUTION files contain the calculated values, which can be used both for restart and also for data analysis. The size of the MESH and SOLUTION files depends on the number of elements in the patch, spectral resolution and number of variables considered in the computation, i.e., velocity field, pressure, temperature, etc.

2.2 Processing high-order spectral element data from NekTar

Processing of NekTar’s data is accomplished in two steps. In the first step the MESH data is read and the computational domain is partitioned across processes (with METIS [8]). In the second step each process reads and stores the SOLUTION corresponding to the elements included in its partition. The modal values of the solution are cached in memory, and are used to transform the solution from modal to physical space, using Equation (1). Using the GUI the user specifies what data should be made available and at what resolution. These two steps are applied to process the data for each patch. The second step is repeated for each time step. After these calculations required to transform data from modal to physical space have been completed, the tasks of computing derived properties (such as derivatives) and converting data to *vtkUnstructuredGrid* of tetrahedra is performed in each partition independently, resulting in linear scaling. Processing of NekTar’s data is performed with the custom reader using the functionality provided by the NekTar library. To reduce data redundancy at the interfaces of elements the *vtkClean-ToGrid* filter is applied. Finally, the cleaned *vtkUnstructuredGrid* is passed on to the ParaView pipeline where all of the standard visualization algorithms can be applied to it.

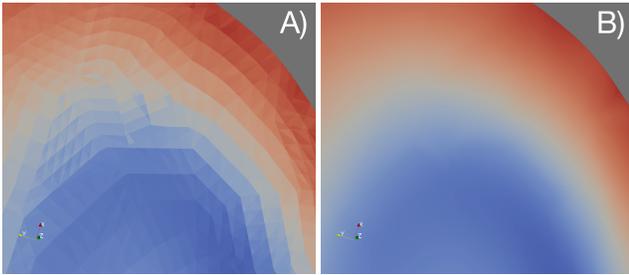


Figure 4: Solution of a flow problem showing vorticity. A) data computed using first order derivatives operator (ParaView). B) data computed using high-order derivatives operator (NekTar). Although the computation times are similar, and both cases use the same final grid, the results of B) are much more accurate.

The data computed by NekTar are processed with high-order spectral accuracy, i.e., the interpolation, integration and differentiation are performed on the quadrature grid consistent with the simulation resolution. In computing derived quantities it is important to take into account that high spatial resolution achieved by P-refinement (i.e. using higher polynomial order) allows for the use of relatively large elements, thus using high-order operators for computing derived quantities is more appropriate. In Figure 4 we plot results of the solution in the right interior carotid artery from our blood flow problem. The vorticity field computed with the two approaches is compared: in Figure 4 (A) the spectral element data is first projected onto a relatively coarse grid and then vorticity is computed (with ParaView) using a low-order derivative operator; in Figure 4 (B) derivatives are computed (using NekTar) with the spectral accuracy consistent with the simulation resolution and the results are displayed on the same grid as in the first case. Clearly a high-order derivative operator provides more accurate results, even for relatively smooth data.

At any time the user can choose to change the resolution of the spectral elements of the mesh by entering the desired resolution through the GUI. Changes in resolution parameters automatically propagate to NekTar’s functions, which use the solution data already cached in memory in its original resolution, to perform interpolation onto a different set of points and update the *vtkUnstructuredGrid*. Similarly, when the user animates through time steps of the data set the geometry of the elements and information on

their connectivity are reused, and only the data from the new SOLUTION file needs to be read from disk. As new time steps are read in the cleaned *vtkUnstructuredGrids* for previous time steps are cached in memory so that they don’t need to be reread from disk each time they are to be displayed. Performing advanced memory management, including pre-fetching future time steps to decrease latency while animating, is an area of future work.

Figure 5 depicts the strong scaling performance of the ParaView-NekTar reader plugin-in, considering the average time to process a single time step of NekTar data. The data used for these tests comprise the four patches that make up the cerebrovasculature shown in Figure 2, which totals about 576 MB per time step on disk. Once read from disk, and transformed to physical space with derived quantities, each time step consumes about 8 GB in memory when calculated using an element resolution of eight. This time includes reading the SOLUTION data from disk and performing interpolation on it, calculating vorticity, and rendering. From the figure we see linear scaling of the time required both when calculating the vorticity using ParaView and when using NekTar, for element resolution of eight. While the performance is nearly identical, the image quality of the vorticity field, as illustrated previously in Figure 4, is superior when calculated with NekTar.

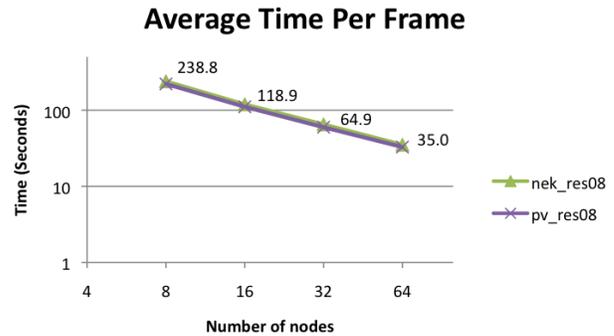


Figure 5: Strong scaling performance of the ParaView-NekTar reader plugin-in. The *nek_res08* line illustrates time when vorticity was calculated using NekTar, while the *pv_res08* line represents overall time when vorticity was calculated using ParaView. In both cases an element resolution of eight was used.

3 MICROSCALE SIMULATION DATA

We consider two types of data produced by our atomistic solver: 1) quantities associated with each particle, such as particle type, coordinates and velocity vectors, activation level for particles representing platelets, etc., and 2) an ensemble average velocity or density computed by projecting atomistic data onto the continuum using window proper orthogonal decomposition. In the following subsections we describe this particle and continuum data in more detail.

3.1 Particle data

Typical DPD simulation of blood flow includes particles representing the plasma, red blood cells (RBCs), platelets, glycocalyx, etc. These particles may have different properties; moreover, some blood cells (i.e. RBCs or glycocalyx) must be represented by a collection of particles with fixed connectivity. For visualization purposes the particles are distinguished by their *ID* and *type*. In simulations with non-periodic domains particles may exit the computational domain over time and be removed, while others are inserted. In such cases the same ID can be associated with different particles at different time steps, which poses obvious challenges in tracing individual particles in time and space.

The number of particles in blood flow simulations can be extremely large; this affects not only the IO complexity and the size of data files, but also the visualization. Fortunately, the majority of the DPD particles (at least in simulations considered here) represent the blood plasma, and their visualization can be substituted by presenting the flow field as a continuum instead of as discrete particles. Using such a technique leads to substantial computational savings. In typical simulations data for every 100 M particles require about 6.5 GB of disk space; avoiding output of data for the plasma particles typically reduces the disk space requirements by 80 to 95 percent. For example, in Figure 6 we visualize data from a blood flow simulation using the DPD method. The large-scale flow features are presented using continuum data computed by a projection of atomistic data onto a specified grid. The small scale features, such as RBC membrane folding, are presented by showing a surface representing each RBC membrane, constructed from a collection of particles (typically 250 to 600).

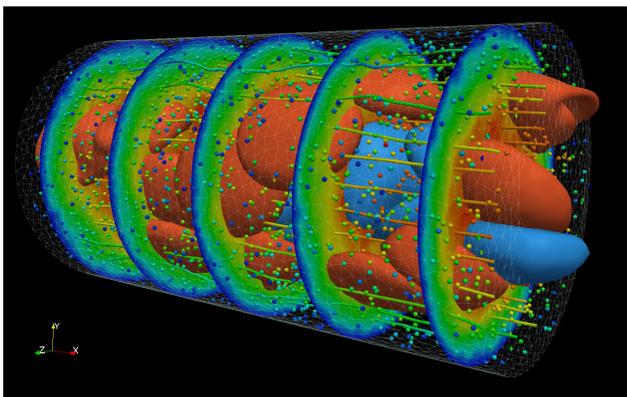


Figure 6: This figure shows healthy (red) and diseased (blue) red blood cells and a fraction of solvent particles (small spheres). Using only a subset of the solvent particles prevents them from occluding the rest of the data. Also shown are streamlines and cutting planes of the velocity field of the continuum data.

In general we have faced two major challenges in the visualization (using ParaView) of atomistic data computed with LAMMPS. The first challenge is due to the lack of a specialized reader to acquire the atomistic data, and in particular the RBC data. The second challenge is computing an ensemble average of a non-stationary solution and projecting the atomistic data onto the continuum.

To overcome the first challenge we have created a utility to process the atomistic data and output it to *vtk* files appropriate for ParaView. The data for particles representing plasma and platelet particles are transformed into *vtkUnstructuredGrids* of vertices. Particles representing RBCs and connected according to the fixed connectivity maps form *vtkUnstructuredGrids* of triangles. Each of the two *vtkUnstructuredGrids* is saved into a separate *vtk* file for each time step. Our current efforts are on transforming this utility into a another specialized ParaView reader plug-in, and to make it available to other users. To overcome the second challenge and accurately compute the ensemble average solution of non-stationary data we employ window proper orthogonal decomposition (WPOD). This method is briefly described in the next subsection.

3.2 Projection of atomistic data onto continuum

WPOD is a spectral analysis tool based in transforming a velocity field into orthogonal temporal and spatial modes: $\bar{u}(t, \mathbf{x}) = \sum_{N_{pod}} a_k(t) \phi_k(\mathbf{x})$. The temporal modes are computed as the eigenvectors of the correlation matrix constructed from the inner product

of velocity fields (snapshots) computed at different times. The velocity field snapshots are computed by sampling (averaging) data over short time intervals, typically $N_{rs} = [50 - 500]$ time steps. The data are sampled over spatial bins centered at the vertices of the finite element mesh. The radius of each bin is comparable to the cut-off radius of the DPD particles and also to the distance between the vertices. The POD eigenspectrum can be subdivided into two regions. The first region contains the fast converging eigenvalues λ_k , corresponding to the low-order modes, while the second region contains the slow converging λ_k , corresponding to the high-order modes. In Figure 7 we plot an eigenspectra and first three POD modes computed in DPD simulation of pulsatile flow in a pipe. The fast converging modes correspond to the motion, correlated in time and space, while the slow converging modes correspond to the thermal fluctuations. The ensemble average solution is computed from the fast converging modes. The continuum data is computed on a

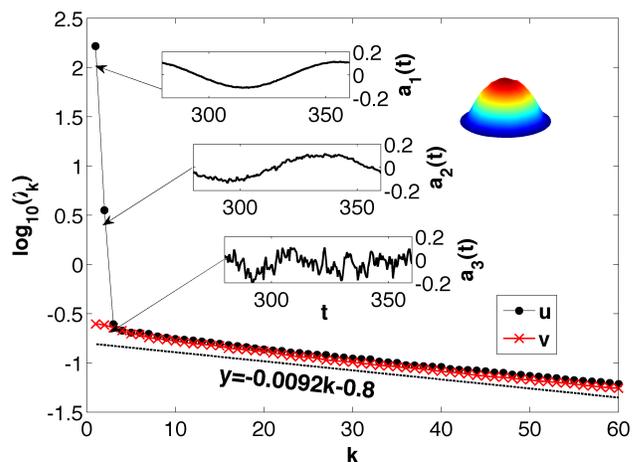


Figure 7: DPD simulations of pulsatile flow in a pipe: POD eigenspectra.

set of grid points, vertices of the finite element mesh, located inside the atomistic domain and having fixed coordinates. These data typically consists of the ensemble average solution representing the velocity field, density and pressure.

Since the grid used for the field data stays consistent over the course of the simulation, only the data values need to be written to disk for each time step. In a post-processing step the static grid and connectivity information are read from disk and used to create a *vtkUnstructuredGrid*. For each time step the data values are read and then applied to this static grid. Again this results in a *vtk* file per time step, including all field variables from the calculation. These can be read into ParaView along with the particle data and standard visualization filters, such as streamlines and cutting planes applied to them, as seen in Figure 6.

4 VISUALIZATION OF MULTISCALE DATA

The methods and software described in the previous sections have been applied to visualize multiscale data from coupled atomistic-continuum simulations of platelet deposition in an aneurysm. The continuum data computed by NekTar reveal the large scale flow features such as flow direction, recirculation regions, swirling flow, etc. Visualization of the atomistic data computed by DPD-LAMMPS, particularly that of activated platelets, illustrate the platelet deposition, along with the detachment of small platelet clusters due to high shear flow, which could not be detected without visualization. In Figure 8 streamlines show the complex flow patterns in the aneurysm, while platelet deposition onto the arterial

wall is depicted by plotting the location of active platelets. Ensemble average velocity fields computed with the WPOD method help to verify the correctness of the coupling between NekTar and DPD-LAMMPS. In Figure 9 the data computed by NekTar and DPD-LAMMPS are compared by plotting the velocity vector fields extracted along the slice at the boundary between the continuum data and the WPOD data. In Figure 9 the region inside the black rectangle was calculated by the DPD-LAMMPS code, while the outside region was calculated by NekTar.

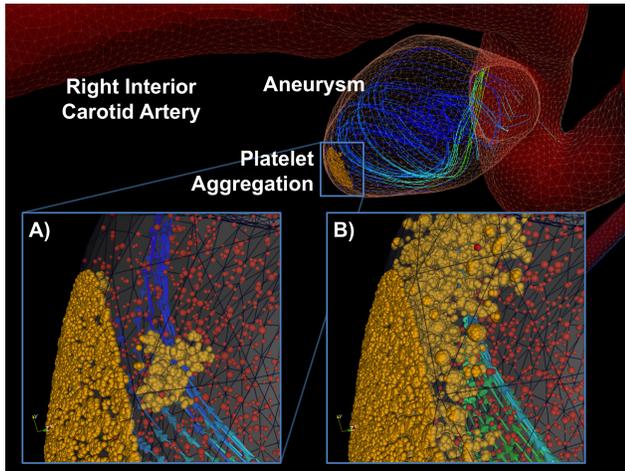


Figure 8: This figure reflects the overall flow through the artery and aneurysm as calculated by NekTar, as well as that within the subdomain calculated by DPD-LAMMPS, seen in greater detail in A and B, together with platelet aggregation along the aneurysm wall.

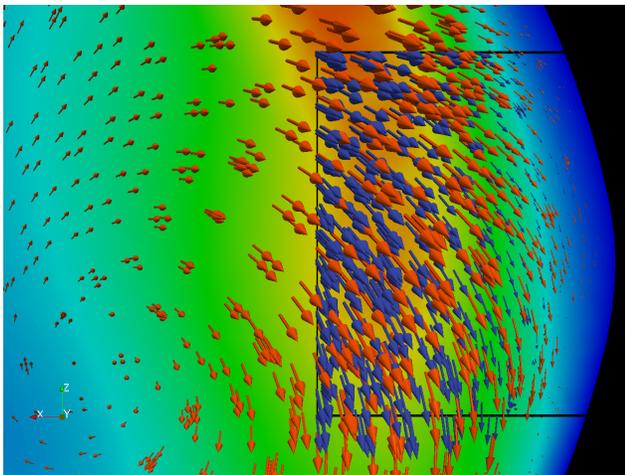


Figure 9: This figure helps to validate that the flow fields calculated by the two different scale codes (NekTar for macroscale, DPD-LAMMPS for microscale) are consistent.

5 DISCUSSION

As our compute capabilities continue to grow, an increasing number of scientific domains will likely be turning to multiscale simulations to more accurately model both physical and biological systems. Having appropriate tools for scientists to visually explore and quantitatively analyze these large heterogeneous data sets will be critical to extracting knowledge from them [7, 3].

In this paper, we described a new NekTar reader plug-in designed to process data computed by a high-order spectral/*hp* element solver, and how it was used to explore macroscale simulation data. While the example used here was for blood flow simulation, it is a generic solution that can be applied to a large number of problems where the solution is obtained using a high-order spectral method. The NekTar reader plug-in will be made available to a wide community of NekTar users, as it will soon be included in the standard distribution of ParaView.

We also discussed tools and a new technique for the visualization of data computed by the atomistic solver LAMMPS. Finally, we have applied these new visualizations tools to present data from multiscale simulations computed by our coupled solver.

Determining the appropriate DPD-LAMMPS data to visualize has been an evolving process. As the capabilities of the integrated solvers are extended and better understood, the data formats are expected to stabilize. As we move closer to that stability, we envision moving toward additional custom ParaView reader plug-ins, which will enable this data to also be read in its native format, without the need to create and store additional data specifically for visualization.

ACKNOWLEDGEMENTS

This work was supported in part by the TeraGrid, the National Science Foundation Grants OCI-0504086 and OCI-0904190, and by the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. We also thank Professor G. E. Karniadakis, Dr. D. Fedosov, and the Visualization group in the Mathematics and Computer Science division at Argonne National Laboratory for their contribution to multiscale simulations. In addition we acknowledge Kitware Inc. for providing technical support in our development of the ParaView plug-in.

REFERENCES

- [1] International assessment of research and development in simulation-based engineering and science, WTEC report. Technical report. Editors: S. Glotzer and S. Kim.
- [2] *LAMMPS Molecular Dynamics Simulator*. Sandia National Laboratories, <http://lammps.sandia.gov>.
- [3] R. Fuchs and H. Hauser. Visualization of multi-variate scientific data. *Computer Graphics Forum*, 28(6), 2009.
- [4] L. Grinberg and G. E. Karniadakis. A new domain decomposition method with overlapping patches for ultrascale simulations: Application to biological flows. *Journal of Computational Physics*, 229(15):5541 – 5563, 2010.
- [5] L. Grinberg, V. Morozov, D. Fedosov, J. A. Insley, M. E. Papka, K. Kumar, and G. E. Karniadakis. A new computational paradigm in multiscale simulations: Application to brain blood flow. In *Proceedings of the 2011 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, Seattle, WA, 2011.
- [6] G. Karniadakis and S. J. Sherwin. *Spectral/hp Element Methods for CFD, second edition*. Oxford University Press, 2005.
- [7] K.-L. Ma, C. Wang, H. Yu, K. Moreland, J. Huang, and R. Ross. Next-generation visualization technologies: Enabling discoveries at extreme scale. *SciDAC Review*, 12:12 – 21, 2009.
- [8] K. Schloegel, G. Karypis, and V. Kumar. Parallel static and dynamic multi-constraint graph partitioning. *Concurrency and Computation: Practice and Experience*, 14(3):219–240, 2002.