Mathematical Modelling of Chemical Diffusion through Skin using Grid-based PSEs

Christopher Goodyer[1], Jason Wood[1], and Martin Berzins[2]

[1] School of Computing, University of Leeds, Leeds, UK
ceg@comp.leeds.ac.uk, jason@comp.leeds.ac.uk
[2] SCI Institute, School of Computing, University of Utah, Salt Lake City, USA
mb@sci.utah.edu

Abstract A Problem Solving Environment (PSE) with connections to remote distributed Grid processes is developed. The Grid simulation is itself a parallel process and allows steering of individual or multiple runs of the core computation of chemical diffusion through the stratum corneum, the outer layer of the skin. The effectiveness of this Grid-based approach in improving the quality of the simulation is assessed.

1 Introduction

The use of Grid technologies [8] has enabled the automated use of remote distributed computers for many new applications. Many of the larger Grid projects around the world are not simply using high performance computers (HPC) but are making better use of the distributed networks of machines within their own companies and institutions. In many of these cases, the problems being tackled on these systems are not so large that parallel computing is being used for single cases, but instead, multiple production runs, sometimes within the context of a design optimization process, are run on a multiprocessor architecture, e.g. [11].

The key elements of any computation are that the results obtained are useful, accurate, and have been obtained as efficiently as possible. Accuracy of the results is an issue that is resolved by an appropriate choice of mathematical model and computational method. The usefulness of the results is usually governed by choice of appropriate input parameters, ranging from those representing the solution case to those concerning the methods used during the computation. A common method for getting better control of the relationship between input parameters and output results has been through the use of Problem Solving Environments (PSEs). These typically combine the inputs, the computation, and visualisation of the outputs into one workflow, where the user is said to "close the loop" by feeding back from the visualisation of the output, to changes to the inputs. A key element of PSEs is that they

must be accessible for non-expert computer users, since the users are typically scientists more focused on their own fields rather than on the intricacies of using the Grid. PSEs are discussed in general in Sect. 3, together with the PSE developed in this work.

The application considered here is that of chemical diffusion through the skin. In this work we have simulated how the application of a chemical on the outside of the body gets through the outer layer into the body. The use of multiple simulations is very important for situations such as this, since each calculation represents only one particular case. For transient cases using the true full 3-d heterogeneous skin structure it can take hours or even days for solutions to fully converge. Through the use of multiple instances of the solver it is possible to reduce this to the maximum individual runtime, provided enough resources are available. Brief details of the solver, and the range of cases being considered are given in Sect. 2.

The interaction of the PSE with the remote processes is the most important part of this work. The software described here has allowed transparent use of the Grid. For example making the process of running a large parallel job on a remote resource as easy as running a small one locally is very important for non-expert users. We have used the gViz library [2, 22] to provide the Grid middleware to handle the communication between the user's desktop and the remote Grid processes. How these components are joined together is discussed in Sect. 4 where the use of a directory service of running jobs is also described.

The main advantages of using a PSE are the ability to visualise the output datasets and to steer the calculations; these are discussed in Sect. 5. The paper is summarised in Sect. 6 along with a summary of the advantages of the Grid-based approach described, and suggested necessary extensions for future work.

2 Chemical Diffusion Through Skin

The motivating problem in this work is that of numerical modelling of chemical diffusion through the skin. Such a situation might arise either purposefully or accidentally. For example a person being accidentally exposed to a chemical at work may hope for minimal adsorption into the body, but application of a drug transdermally could have great therapeutic benefits.

The barrier function of the skin comes almost entirely from the outermost layer, the *stratum corneum*. This is a highly lipophilic layer only about 10μm thick. Once a chemical has got through the stratum corneum it is into the *viable epidermis*, which is hydrophilic, and hence effectively into the blood steam. The stratum corneum itself is made up of between six and 40 layers of corneocytes. Each corneocytes is hexagonal in shape, and is typically about 40μm across, 1μm high. They are surrounded by a lipid layer about 0.1μm wide, both between the tessellating corneocytes and between the individual layers. It is through this lipid that almost all of the chemical diffuses since the corneocytes themselves are almost impermeable. The diffusion path through

the lipid from surface to body is very tortuous due to the aspect ratio of the corneocytes.

The mathematical model of this Fickian diffusion process has been well understood for some time, although the modelling is often only done on 1-d homogeneous membranes. There has been some 2-d work on "brick and mortar" shaped domains, notably that of Heisig et al. [13] and Frasch [9] but this work is the first to tackle the true three dimensional nature of the skin.

The model of Fickian diffusion is given in non-dimensional form by:

$$\frac{\partial\theta}{\partial\tau} = \frac{\partial}{\partial X}\left(\gamma\frac{\partial\theta}{\partial X}\right) + \frac{\partial}{\partial Y}\left(\gamma\frac{\partial\theta}{\partial Y}\right) + \frac{\partial}{\partial Z}\left(\gamma\frac{\partial\theta}{\partial Z}\right) , \tag{1}$$

where X, Y and Z are the coordinates, τ is the time, θ is the concentration, and γ the diffusion coefficient, with boundary conditions:

$$\theta = 1, \text{ on } Z = L \quad \theta = 0, \text{ on } \Gamma : Z = 0 ,$$

and periodic boundary conditions between opposing faces perpendicular to the X-Y plane. In this work we are assuming that the corneocytes themselves are impermeable and hence on these boundaries we are assuming symmetry conditions perpendicular to the boundary. The domains have been meshed using unstructured tetrahedra, and discretised and solved using a Galerkin linear finite element solver, based on that originally developed by Walkley [19].

The quantities of interest are the concentration profile throughout the domain; the total flux out through the bottom face and the *lag time* a measure of the relative time taken to reach steady state. These are defined precisely as

$$\text{Flux out, } F_{out} = \frac{1}{A_\Gamma}\int_\Gamma \frac{\partial\theta}{\partial Z}\mathrm{d}\Gamma \tag{2}$$

$$\text{Cumulative mass out} = \int_{\tau=0}^{T} F_{out}\mathrm{d}\tau , \tag{3}$$

where the lag time is the X-intercept of the steady state rate of cumulative mass out, extrapolated backwards.

The key questions being considered with the solver thus far have been concerned with the physical geometric model of the skin. The differences in the alignment of 2-d corneocytes was considered in [12] and these differences are further complicated by the move to 3-d. We have been considering the effects of alignment, and how these are affected by the aspect ratio of the corneocytes and the thickness of the lipids. Part of this work is to verify the previously published geometric estimates for the effect of "slits, wiggles and necking" around the corneocytes [5, 14]. The effect of the number of layers of corneocytes is also of great importance. The idea of layer independence of quantities needs rigorous proof, as the relative difference between having two or three layers is much greater than the difference between ten and eleven layers. Another case of great importance concerns the effect of the size of

application patches to the skin. Since any patch spreads out sideways as well as down, even through homogeneous membranes, then the separation of patches compared to the depth of stratum corneum makes significant changes to both the mass of chemical getting into the body and the lag time.

3 Problem Solving Environments

PSEs allow a user to break away from the traditional batch mode of processing. Traditionally, the user of a scientific research code would set all the parameters inside the program, compile the software, run it with the results saved to disk. These results could then be loaded into a visualisation package, processed and rendered as a screen output. Only once this sequence had been completed would the user be able to decide if the original set of parameters needed alteration, and the whole process would begin again.

PSEs are typically built in or around visualisation packages. Some commercial visualisation products have added PSE capabilities during their development, such as IRIS Explorer [21] and AVS [18]. Other systems that were built around particular application areas have expanded into more general products, such as SCIRun [16]. These all have the same key elements of modular environments where graphical blocks represent the tasks (modules), with wires joing the blocks representing dataflow from one task to another. The arrangements of modules and wires are variously referred to as workflow, dataflow networks and maps. With a PSE the simulation is presented as a pre-compiled module in the workflow, and all the variables that may be changed are made available to the user through a graphical user interface (GUI).

An example of the interface of the standard PSE for chemical diffusion through the skin, with the simulation running completely on the local machine, is shown with IRIS Explorer in Fig. 1. The key parts of the environment are, clockwise from top left, the librarian (a repository of all available modules), the map editor (a workspace for visually constructing maps), the output visualisation, and the input modules. IRIS Explorer has a visual programming interface where modules are dragged from the librarian into the map editor and then input and output ports are wired together between different modules to represent the flow of data through the pipeline, typically ending up with a rendered 3-d image. The inputs to the modules are all given through GUIs enabling parameter modification without the need to recompile.

In the map shown the first module is the interface to the mesh generator, Netgen [17]. This module takes inputs concerning the alignment of the corneocytes, their sizes and separation, and how much their alignment is staggered. It then passes the appropriately formatted mesh structure onto the second module which performs the simulation. This module takes user inputs such as the location and size of chemical patch on the skin, whether the solve is to be steady state or transient, and computational parameters such as the level of initial grid adaptation and the frequency of generation of output datasets.

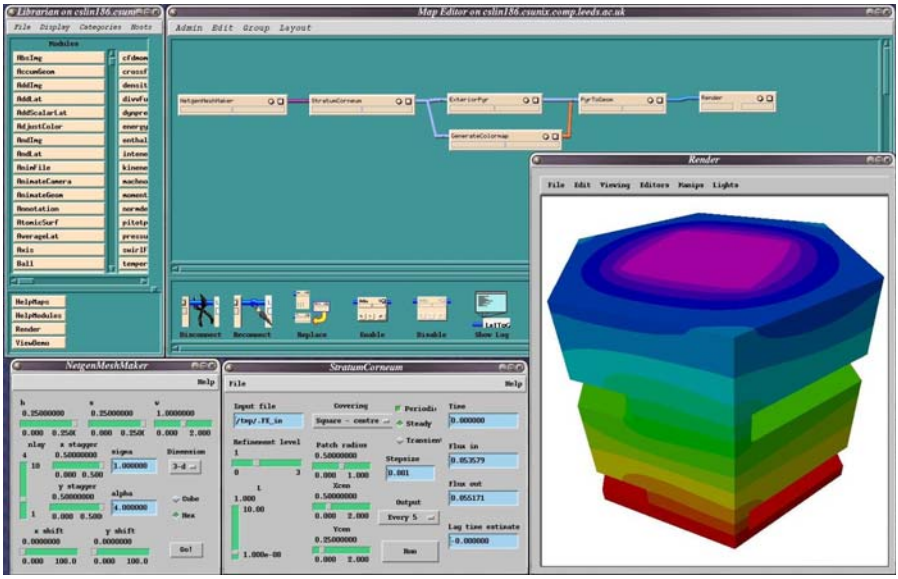

Figure 1. IRIS Explorer PSE for skin

4 Remote Grid Simulation

One of the drawbacks to using a PSE used to be that the simulation was part of the environment and therefore had to be run locally within the PSE. This assumption, however, is not necessarily true as was shown by the 'Pollution Demonstrator' of Walkley et al. [3,20] which extended the approach to that of a remote simulation connected to a local (IRIS Explorer) PSE which still handled the steering and visualisation needs. This bespoke approach has been extended into a generic thread-based library, called *gViz* [2] providing an interface to the remote simulation. The local PSE environment is now independent and hence no longer needs to be IRIS Explorer.

The gViz library itself operates as a collection of threads attached to the simulation code which manage the connections between the simulation and any connected users. Thus, once the code is running, the simulation initialises the library making itself available for connections from multiple users. When a user connects to the simulation new threads are started to handle steering and output datastreams. The library does not limit the number of connected users making it possible for multiple users to be collaborating by visualising the output results and modifying input parameters of the same simulation.

To launch the simulation onto a remote resource it is necessary to both select the desired machine and an appropriate job launching mechanism. The launching module can interrogate a GIIS service (part of the Globus toolkit [7]) to discover information about available resources. Once the simulation starts, the gViz thread opens a port to which connections can be made.

The gViz library supports several communications methods between simulation and other processes. Here we are using a web service style *directory service* communication built around SOAP using the gSOAP library [6]. In this method, the simulation contacts the directory service at startup and deposits its connection details. These can then be retrieved by the one or more PSEs when convenient to the user.

The ability to undertake collaborative work is provided at no extra cost through the use of the gViz library. Since the directory service can be advertised to collaborators as running in a consistent location they will all be able to connect to this to discover the location of all the running simulations. This also assists users who may wish only to visualise the output rather than steer.

When multiple simulations are involved then there are additional job submission considerations. Many providers of Grid resources are actively limiting the number of submissions per person able to be executed simultaneously, and hence running large numbers concurrently is often not possible on the chosen resource. In the work of Goodyer et al. [11] it was seen how the use of a parallel environment was beneficial for solution time of a previously serial optimisation application, through concurrent simulation of independent cases with similar parameter sets. Here we extend this idea to take the multiple independent simulations inside one large MPI job. This means that co-allocation of all runs is handled by the resource's own job scheduler and reduces the number of submitted jobs to just one, thus avoiding any resource limits set. When the MPI job starts all simulations register separately with the directory service, with unique identifiers allowing each to communicate back to the PSE separately if required.

5 Visualization and Steering

With multiple simulations being controlled by the same PSE there are issues concerning effective management of both input and output data, [4,22]. In this section we address these in turn, relating to how they have been used for the skin cases solved to date. The cases discussed here are all concerned with assessing how the geometric makeup of the skin affects the calculated values of flux out and lag time. To that end we have already generated a large selection of 3-d meshes for the solver to use On start-up each parallel process uses standard MPI [15] commands to discover its unique rank in the simulation, and load the appropriate mesh.

The steering control panel has a collection of input variables, known as *steerables* and output values, known as *viewables*. The steerables are the inputs as described in Sect. 2, and the viewables are output variables calculated by the simulation to which the module is currently attached, including the quantities of interest and the current time of the solver through a transient simulation.

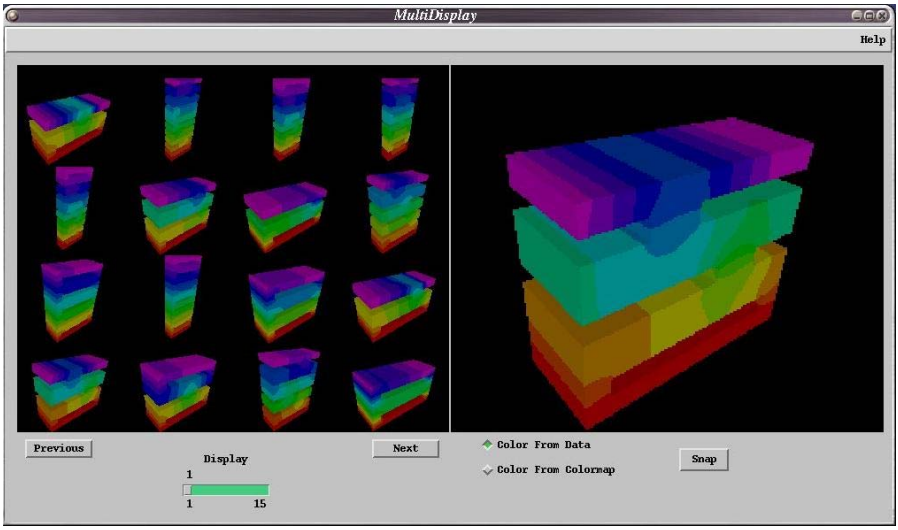

Figure 2. MultiDisplay visualising many output streams

Another difference between the PSE shown in Fig. 1 and the Grid enabled version is that this steering module now has two location bars along the bottom, and the buttons 'Connect' and 'Steer all'. The two location bars contain the connection information retrieved from the directory service: the first being an individual location, the second being a multiline list of simulations. The connections for steering updates can therefore be actually given to anywhere from one to all of the simulations simultaneously. In this manner it is possible to extend the experimentation approach, discussed in Sect. 3 to many simulations, rather than one specific case at a time. In addition to the ability to 'Steer all' it is similarly possible to retrieve all the current viewable parameters from the simulations currently running, and hence the ability to produce summary visualisations of all the cases.

The visualisation of remote gViz processes for both 2-d and 3-d datasets has been done in IRIS Explorer, SCIRun, VTK and Matlab [1, 2, 10, 20]. The data sent is generic to enable the client PSE end to convert it into a native format.

For concurrent simulations visualisation is conceptually no harder than for a single simulation, however the realities are slightly different. In the work of Handley et al. [1] this has been tackled for the first time. Here multiple datastreams from multiple simulations are combined, and the 2-d output is tiled across the display. In 3-d this is a significantly harder problem, because the quantities of data being returned by the simulation are potentially orders of magnitude more. To address this issue we have developed a 'MultiDisplay' module, as shown in Fig. 2. This module receives images rather than the usual 3-d data. The individual datastreams from the simulations are rendered off

screen, with an appropriate colourmap and camera position to produce a 2-d image, and these images are fed into the MultiDisplay module. The image pane on the left shows tiles of 16 simulations at a time, and these may be enlarged into the right hand pane for closer inspection.

With multiple simulations connected to the PSE the quantity of data being received is the same, regardless of how it is visualised. It is just the quality of the rendering process to get the final images which can make a difference to the time and memory used.

6 Conclusions

In this work we have demonstrated how an intensive finite element solution code can be run through a Problem Solving Environment. It has been seen that this simulation can be launched remotely onto a Grid resource, and still remain interactive for both steering and output visualisations. Through the use of running multiple simulations run through one large MPI job it is possible to have concurrent calculation of different cases, with the same set of steerable inputs, for example. Steering has been shown to be possible on both individual cases and sets of cases up to a 'steer all' capacity.

The output visualisations for these cases have been shown to be possible in a highly detailed manner, through the use of the visualisation environments processing power on individual cases, or in a group processing fashion to render the simultaneous cases all in the same window.

The use of the PSE has enabled two forms of steering to the skin scientist user. Firstly, the ability of the scientist to ask "What if...?" questions on individual cases has been seen to be very beneficial in getting quick answers to individual questions. Through the use of multiple concurrent simulations it has been possible to get more general answers over a wider range of cases than had been possible with just the single run.

An important part of running the simulations through the PSE and on the Grid has been the ability to detach the local processes from the remote simulations, hence enabling monitoring on demand rather than being constantly connected. It is true that when potentially hundreds of cases are being run for hours and days that the contact time with the simulations is probably only a small percentage of that, however the added flexibility enables the user to see instantly when a parameter has been set up incorrectly. This means that erroneous computations can be minimised and all cases adjusted accordingly.

There are several issues that have arisen out of this work needing further consideration. The main ones are concerned with how to efficiently handle the visualisation of large numbers of large datastreams. For the skin geometries we have considered here the obvious first step would be to give the user the option of only retrieving the surface mesh. This would reduce the necessary transmission time from the simulation to the PSE, and also reduce the amount of work necessary locally to render the final image. By retaining the

full dataset at the simulation end it would be possible to examine the data of an individual simulation in greater depth than would normally be done for multiple simulations. Another potential idea would be to actually do far more of the visualisation work remotely, hence reducing the local load even further. This could be done by using products such as the Grid-enabled version of IRIS Explorer discussed by Brodlie et al. [2] to put the visualisation process closer to the simulation than to the desktop, hence removing the need to the data to reach the local machine before the final rendering.

Acknowledgments

This work is funded through the EPSRC grant GR/S04871 and builds on the "gViz Grid Middleware" UK e-Science project. We also acknowledge Annette Bunge of the Colorado School of Mines for her collaboration and expertise.

References

1. Aslanidi, O. V., Brodlie, K. W., Clayton, R. H., Handley, J. W., Holden, A. V. and Wood, J. D. Remote visualization and computational steering of cardiac virtual tissues using gViz In: Cox, S., and Walker D. W., eds.: Proceedings of the 4th UK e-Science All Hands Meeting (AHM'05) EPSRC (2005)
2. Brodlie, K., Duce, D., Gallop, J., Sagar, M., Walton, J. and Wood, J. Visualization in Grid Computing Environments. IEEE Visualization (2004)
3. Brodlie, K. W., Mason, S., Thompson, M., Walkley, M. A., and Wood, J. W. Reacting to a crisis: benefits of collaborative visualization and computational steering in a Grid environment. In: Proceedings of the All Hands Meeting 2002 (2002)
4. Brooke, J. M., Coveney, P. V., Harting, J., Jha, S., Pickles, S. M., Pinning, R. L., and Porter, A. R. Computational steering in RealityGrid. In: Cox, S., ed.: Proceedings of the All Hands Meeting 2003, EPSRC, 885–888 (2003)
5. Cussler, E. L., Hughes, S. E., Ward III, W. J. and Rutherford, A. Barrier Membranes. Journal of Membrane Science, 38:161-174 (1988)
6. van Engelen, R. A. and Gallivan, K. A. The gSOAP Toolkit for Web Services and Peer-To-Peer Computing Networks. In: Proceedings of IEEE CCGrid. (2002)
7. Foster, I. and Kesselman, C. Globus: A metacomputing infrastructure toolkit. International Journal of Supercomputer Applications, **11**, 115–128 (1997)
8. Foster, I. and Kesselman, C. The Grid 2 : The Blueprint for a new computing infrastructure. Elsevier (2004)
9. Frasch H. F. and Barbero A. M. Steady-state flux and lag time in the stratum corneum lipid pathway: results from finite element models. *Journal of Pharmaceutical Sciences*, Vol 92(11), 2196–2207 (2003)
10. Goodyer, C. E. and Berzins, M. Solving Computationally Intensive Engineering Problems on the Grid using Problem Solving Environments. In: Cuhna, J. C. and Rana, O. F., eds., *Grid Computing: Software Environments and Tools*. Springer Verlag (2006)

11. Goodyer, C. E., Berzins, M., Jimack, P. K., and Scales, L. E. A Grid-enabled Problem Solving Environment for Parallel Computational Engineering Design *Advances in Engineering Software*, 37(7):439–449 (2006)
12. Goodyer C. E. and Bunge A. What If...? Mathematical Experiments on Skin, In: *Proceedings of the Perspectives in Percutaneous Penetration*, La Grande Motte, France (2004)
13. Heisig M., Lieckfeldt R., Wittum G., Mazurkevich G. and Lee G. Non steady-state descriptions of drug permeation through stratum corneum. I. the biphasic brick-and-mortar model. *Pharmaceutical Research*, Vol 13(3), 421–426 (1996)
14. Johnson, M. E., Blankschtein, D. and Langer, R. Evaluation of solute permeation through the stratum corneum: Lateral bilayer diffusion as the primary transport mechanism. *Journal of Pharmaceutical Sciences*, 86:1162-1172 (1997)
15. Message Passing Interface Forum: MPI: A message-passing interface standard. International Journal of Supercomputer Applications **8** (1994)
16. Parker, S. G., and Johnson, C. R. SCIRun: A scientific programming environment for computational steering. In: Meuer, H. W., eds.: Proceedings of Supercomputer '95, New York, Springer-Verlag (1995)
17. Schöberl, J. Netgen mesh generation package, version 4.4, (2004) `http://www.hpfem.jku.at/netgen/`
18. Upson, C., Faulhaber, T., Kamins, D., Laidlaw, D., Schlegel, D., Vroom, J., Gurwitz, R. and van Dam, A. The application visualization system: A computational environment for scientific visualization. IEEE Computer Graphics and Applications, 9(4):30–42 (1989)
19. Walkley, M. A., Jimack, P. K. and Berzins, M. Anisotropic adaptivity for finite element solutions of 3-d convection-dominated problems. *International Journal of Numerical Methods in Fluids*, Vol 40, 551–559, (2002)
20. Walkley, M. A., Wood, J., Brodlie, K. W. A distributed collaborative problem solving environment. In: Sloot, P. M. A., Tan, C. J. K., Dongarra, J. J., Hoekstra, A. G., eds.: Computational Science, ICCS 2002 Part I, Lecture Notes in Computer Science. Volume 2329, 853–861, Springer (2002)
21. Walton, J. P. R. B. Now you see it – interactive visualisation of large datasets. In: Brebbia, C. A., Power, H., eds.: Applications of Supercomputers in Engineering III. Computational Mechanics Publications/Elsevier Applied Science (1993)
22. Wood, J. W., Brodlie, K. W., Walton, J. P. R. gViz: Visualization and computational steering for e-Science. In: Cox, S., ed.: Proceedings of the All Hands Meeting 2003, EPSRC, 164–171 (2003) ISBN: 1-904425-11-9