Grid-Based Numerical Optimisation in a Problem Solving Environment

C.E. Goodyer[1], M. Berzins[1], P.K. Jimack[1] and L.E. Scales[1,2]
[1] School of Computing, The University of Leeds, Leeds, LS2 9JT
[2] Shell Global Solutions, Cheshire Innovation Park, Chester, CH1 3SH

http://www.comp.leeds.ac.uk/ceg/ehlgospel.html

Abstract

In this paper we show how a Problem Solving Environment (PSE) can be used to manage and steer numerical optimisation of a challenging problem from mechanical engineering running in parallel on a remote Grid resource. The industrial code used by Shell Global Solutions in their lubrication work is transformed from a serial code through an interactive PSE providing multivariate visualisations to a parallel application run on a remote Grid resource. Use is made of Globus, MPI, gViz and NAG optimisation libraries for this complete solution environment. The techniques developed, whilst focused on a particular problem, are intended to be generic and extensible to wider engineering applications.

1 Introduction

Industry uses numerical codes for investigation, validation, and testing purposes. All of these processes require highly intensive computational procedures which may vary from large-scale parallel HPC applications to smaller scale individual problems requiring many similar cases to be solved. Access to large computational resources are important for all these types of application. Using the Grid gives an added freedom in terms of where the job may be run, but this must not come at the expense of being able to interact with the simulations.

In previous work [5, 6, 12] it has been shown how an interactive Problem Solving Environment (PSE) for an individual numerical simulation may be constructed and Grid-enabled using IRIS Explorer [13]. In this work the PSE has been re-engineered to be at the level of the industrial user working with Shell Global Solution's optimisation software as applied to elastohydrodynamic lubrication (EHL) modelling. In this work we explain how the Shell software, has been parallelised and enabled to be run on a Grid resource in order to significantly reduce run times. The use of the PSE enables both interactive visualisation of the progress of the optimiser and the ability to interact with the simulations already running on the Grid to help guide the optimiser through the high dimensional parameter space. The complete package is called GOSPEL - Grid Optimisation Software for Problems of Elastohydrodynamic Lubrication.

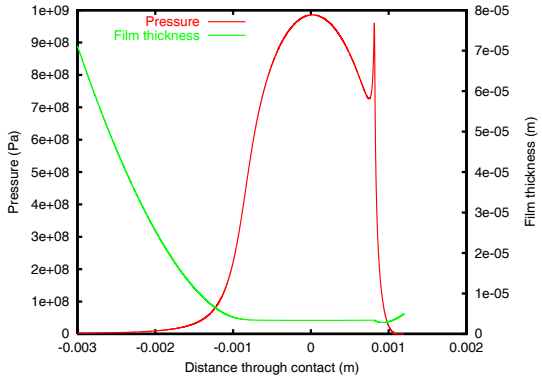
The numerical problem being considered, elastohydrodynamic lubrication (EHL), is briefly described in Section 2, along with a description of the optimisation procedure in terms of the quantities being optimised against and for. Section 3 describes the necessary changes to turn what was previously a serial application into a distributed memory parallel application with fast solution times. Consideration of the appropriate degree of parallelism for this application is given here.

The PSE is a vital tool in effective use of Grid resources; the ability to interact with a simulation to guide the solution or to change the problem being solved are both very important. From a PSE this can be done without recompilation of code or resubmission of the job onto the Grid. As Grid cycle accountancy through utility computing on demand becomes used then it will be important not to have wasted clock cycles, so the ability to know as soon as a simulation has gone awry will actually save money as well as time. These PSE aspects are considered for the EHL optimiser in Section 4. The Grid-enabling aspects of the PSE are also described in this section, along with a description of how the gViz collaborative libraries [14] are used. The output visualisations for such a complex problem are also very important in being able to effectively use the PSE and these are described in Section 5.

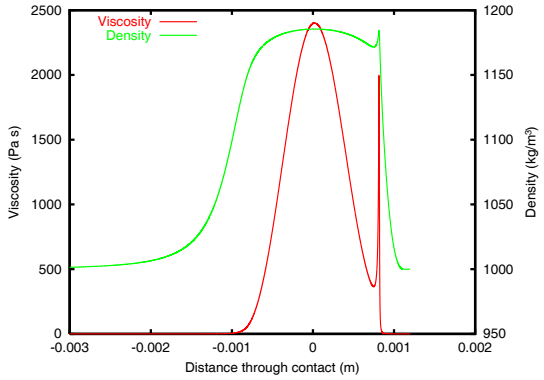
The paper is concluded in Section 6 where further work is proposed. Also described in this section is the next stage of the work to expand the simulations into the computationally more demanding 2-d case which will require hierarchies of parallelism within the solution scheme.

2 EHL and Optimisation

The particularly challenging numerical problem of EHL occurs, for example, in journal bearings and



(a) Pressure (left hand axis) and film thickness



Effective viscosity (Left hand axis) and density

Figure 1: Solution profiles of an EHL line contact

gears where, at the centre of the contact the load exerted over a very small area causes extremely high pressures (up to 3 G Pa) resulting in elastic deformation of the components and significant changes in the lubricant properties in this area. Example solutions are shown in Figure 1 for (a) the pressure and film thickness, and (b) the density and effective viscosity. The numerical solver used by Shell has been developed at Leeds over the past decade. The solution techniques used are described in detail in Fairlie *et al.* [3] and Goodyer [4]. More general information about the techniques used in numerically solving EHL problems may be found in the work of Venner and Lubrecht [11].

At the heart of the EHL problem are the equations describing the operational condition and the rheological model of the lubricant used. The variables defining the cases therefore themselves fall into the same two categories with up to 40 parameters being required to specify a full non-Newtonian thermal EHL simulation. The physical parameters which may be varied include the loading of the contact, the ambient temperature and *slide to roll ratio*, a measure of the amount of slip of one component past the other.

The optimisation work is intended to try to find the set of lubricant properties that best match the total friction through the contact from numerical calculations to the observed friction in experiments performed on a test rig under a sequence of different physical conditions. For the case outlined in the rest of this work, the experiments have been run at three different loadings, two different temperatures and six different slide to roll ratios giving a total of 36 different cases. By using a numerical solver it is possible to run each of these cases for a particular input parameter set. Ten of the lubricant rheology parameters have been optimised to try to find the parameter set that most closely matches the frictional behaviour of the real lubricant. The chosen

Variable	Description
β	Temperature coefficient of viscosity (K^{-1})
z_0	Viscosity parameter
z_1	Viscosity parameter
K_0	Inverse critical shear rate (s)
$\overline{\alpha}$	Pressure coefficient of inverse critical shear rate
$\overline{\beta}$	Temperature coefficient of inverse critical shear rate
$\overline{z_0}$	Inverse critical shear rate parameter
$\overline{z_1}$	Inverse critical shear rate parameter
m	Cross exponent
a	Carreau-Yusada parameter

Table 1: Lubricant parameters used for optimisation

parameters are shown in Table 1. For a given set of lubricant parameters, case $\underline{x}_i$ say, the total frictional residual, $\mathscr{R}_F$ is given by

$$\mathscr{R}_F = \sum_{j=1}^{36} \left(F_j^{num} - F_j^{exp} \right)^2 \qquad (1)$$

where F_j^{num} and F_j^{exp} are the numerical and experimental values of the friction for physical parameters case j. With ten physical parameters to vary the optimiser is thus trying to minimise $\mathscr{R}_F$ in ten dimensional space. Furthermore the evaluation of $\mathscr{R}_F$ requires 36 computationally intensive EHL problems to be solved.

The physical requirements and the challenging nature of the numerical EHL solver mean that a very robust optimisation method is needed. Shell have found that the simplex method [2, 9] proves to be the most reliable. The implementation used here is from the NAG C library [1]. In brief this method

[1] `http://www.nag.co.uk`

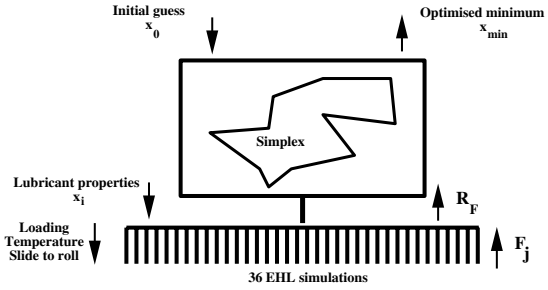


Figure 2: Optimiser schematic

takes an initial supplied estimate of the solution $\underline{x}_0$ and, in n-dimensional space generates a simplex with $n+1$ vertices, $\underline{x}_0$, $\underline{x}_1$,..., $\underline{x}_n$ according to the method of Parkinson and Hutchinson [10]. At each point, $\underline{x}_i$ the value of $\mathscr{R}_F$ is calculated and the vertex with the maximum $\mathscr{R}_F$ value is reflected through the centre of gravity of the remaining vertices, and $\mathscr{R}_F$ at this new point evaluated. This result may lead to further moves of this vertex along this direction.

Each evaluation of $\mathscr{R}_F$ incurs the cost of performing 36 EHL solutions, and the typical number of $\mathscr{R}_F$ evaluations required in a run is of the order of 1×10^3, hence being able to speed-up the evaluation of these functions is very important. The overall schematic of the optimiser is shown in Figure 2. This shows the dataflow with the 36 EHL cases at the bottom with varying $\underline{x}_i$ lubricant parameter sets being supplied by the optimiser from potential points in the simplex. Each case returns an F_j contribution to the $\mathscr{R}_F$ value. Finally the optimiser returns a minimum solution from the search space of $\underline{x}_{min}$.

3 Parallelism

The parallelism in this optimisation problem comes from the fact that for each evaluation of the function value $\mathscr{R}_F$ we need to perform 36 EHL calculations. Each of these calculations will have the same lubricant characteristics just different operating conditions. Theoretically this should mean that all 36 processes can be run independently since the result of one does not influence any of the others. However there are great time savings to be made for EHL problems by using continuation. That is, the result to one problem is a very good guess for the solution to a similar problem. For example, by having the six cases with the same loading and ambient temperature solved in turn only the first calculation (for the lowest slide to roll ratio, say) is relatively expensive compared to the others (which are undertaken with sequentially larger slide to roll rations, say). The particular choice of which directions to use continuation in will be returned to below.

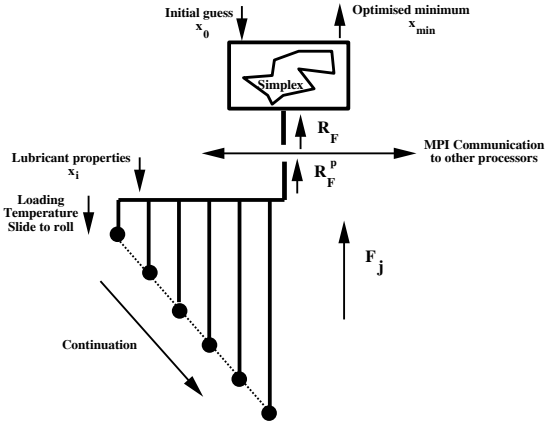


Figure 3: Parallel optimiser schematic

The parallel software from this project is designed for computational grids such as the White Rose Grid[2] with its mixture of shared and distributed memory machines, including a 256-processor Beowulf style cluster. For reasons of portability the parallelism is undertaken using MPI [8].

The EHL case being solved in this work is a one dimensional line contact problem. Each of the individual cases fits easily in memory and has little need for parallelism itself. This means that the parallelisation may be focused at the level of the optimiser. The work per processor is sketched in Figure 3. Since each processor can perform one set of continued runs then the only communication necessary is each individual processor's contribution $\mathscr{R}_F^p$ to the global $\mathscr{R}_F$. Once the combined total has been accrued then the optimiser can function as for the serial case.

The use of continuation adds an extra level of robustness to the solver. Since the optimiser may occasionally select values for parameters outside the normal physically expected ranges then having a good initial guess may make the difference between convergence and failure. Whilst failure is itself a valid conclusion which is handled by setting the calculated friction to have a 100% error these solutions may be very expensive computationally to calculate. Table 2 compares various different continuation schemes and shows the results for the maximum number of processors (36) no continuation possible, continuation with increasing temperature (2 runs per processor), with increasing loading (3 runs per processor) and increasing slide to roll ratio (6 runs per processor). It can be clearly seen that maximising the amount of continuation used is very important for reducing the overall run-

[2] `http://www.wrgrid.org.uk`

Continuation scheme	Processors	Solution time (s)	Number of $\mathscr{R}_F$ evaluations	Convergence failures	Average time per $\mathscr{R}_F$ evaluation (s)
No continuation	36	5916	1001	315	5.91
Temperature	18	6133	1011	309	6.07
Loading	12	1413	210	21	6.72
Slide to roll	6	678	210	7	3.23

Table 2: Optimiser solution times for varying continuation schemes

time. The number of convergence failures is the number of individual EHL simulations which failed to converge. There are clearly more of these for cases where continuation has not been used as much.

4 Grid-enabling the PSE

A PSE generally has both numerical simulation and output visualisation as core components. The addition of computational steering abilities means that visual feedback to the user can then be used to modify the simulation already running. Such a system for individual EHL problems has already been described by Goodyer *et al.* [5, 6]. In this work we have chosen to build the PSE in NAG's IRIS Explorer package.

Much of this part of the work has used the gViz libraries which are described elsewhere in this volume [14]. In brief gViz provides a communication interface for a process running on a (typically) Grid resource to enable other users to connect to the simulation and either visualise the results or steer the calculation.

An example of a typical map for the PSE is shown in Figure 4 where the dataflow pipeline, generally from left to right, is clearly visible. The majority of the modules are used in the visualisation process and hence only the three modules on the left are described here.

The first module in the map, *GlobusSearch* interrogates a GIIS server to analyse the available resources and their current statuses [1]. The user can then select a resource and choose a suitable launch method, including launching the job onto the Grid using Globus[3]. For this work we have extended the gViz library to include parallel launch mechanisms including writing a parallel job submission script or Globus RSL script which then gets submitted to Sun Grid Engine for scheduling onto a suitable node. When the job is spawned a socket connection back to the PSE is made telling the launching application which node of the Grid resource the simulation will be communicating from. Information about this node and port is then passed to the next two modules in the map, *SteerGOSPEL* and *VisualiseGOSPEL*. Knowledge of where the simulation is running also allows any other user access to the simulation through the gViz libraries. This means that one person, with Grid certification, say, can start the simulation and other collaborators around the world can then all see the results of that simulation and help to steer the computation [1, 12]. In fact, the person who originally launched the Grid job need not actually be involved from that point on.

Computational steering is the ability to change a simulation that is already running. One example of this could be choosing to use a lower quality mesh in the early stages of the solve, but as the solution gets near to a minimum using a higher resolution mesh to improve the accuracy of the solution obtained.

The module *SteerGOSPEL* has several uses. Firstly it shows the current best set of values found by the simplex, along with $\mathscr{R}_F$. This allows a user access to individual numbers from the simulation rather than much larger datasets for visualisation purposes. These numbers can also be used for steering. For example it is possible to resubmit this current best set to the optimiser once a minimum has been found. The NAG library will then build a new simplex around this previous minimum potentially allowing it to escape from local minima. Similarly, a different point in the search space can be specified away from where the optimiser has previously searched. Finally, as mentioned, the accuracy can be changed. A method we have implemented here is the ability to turn on (or off) the thermal components of the solution. The thermal solve is much more expensive but adds greater accuracy to the friction results obtained, especially for cases where more heat is generated [3].

Communication from the PSE to the simulation is done through the gViz libraries. At suitable points the simulation will check if any new input data has been received. If a steering request is for additional accuracy, say, then these changes can be introduced without changing the points of the current simplex and would therefore only apply to future calculations. If, on the other hand, a new simplex was re-

[3] `http://www.globus.org`

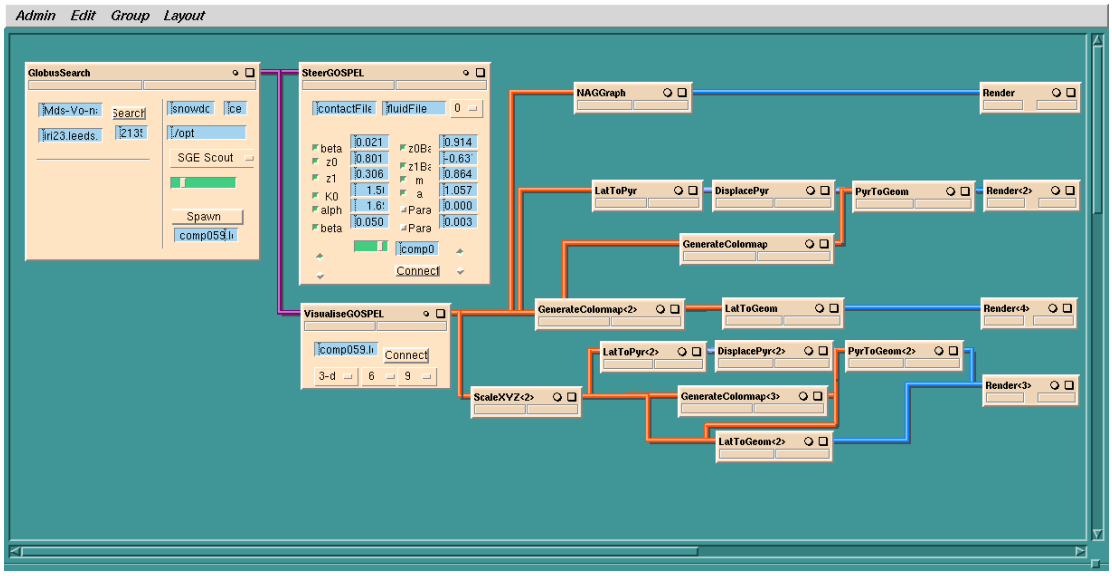


Figure 4: IRIS Explorer map of GOSPEL. Dataflow represented by wires between modules.

quested then the NAG libraries do not allow movement of the current simplex points and hence use of the communication flag inside the routine will cause the optimisation routine to drop out of the NAG routines and then the new simplex is submitted.

The *VisualiseGOSPEL* module communicates with the simulation to receive all the datasets for visualisation. These are then packaged up into standard IRIS Explorer datatypes and sent down the rest of the map for visualisation. When the full datasets are being shown then more information needs to be returned from the parallel nodes than is necessary for just the optimisation process. The root process which is communicating with any attached users also needs to retain full copies of all output data previously generated so that any listeners joining the simulation later get the full set of results rather than just those generated from that stage. The descriptions of the output datasets are explained in the following section.

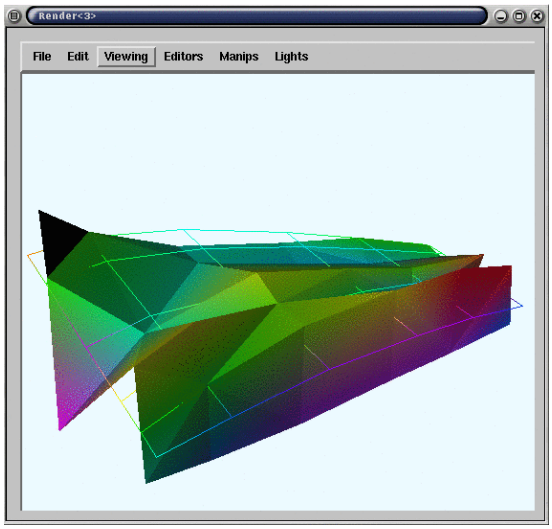
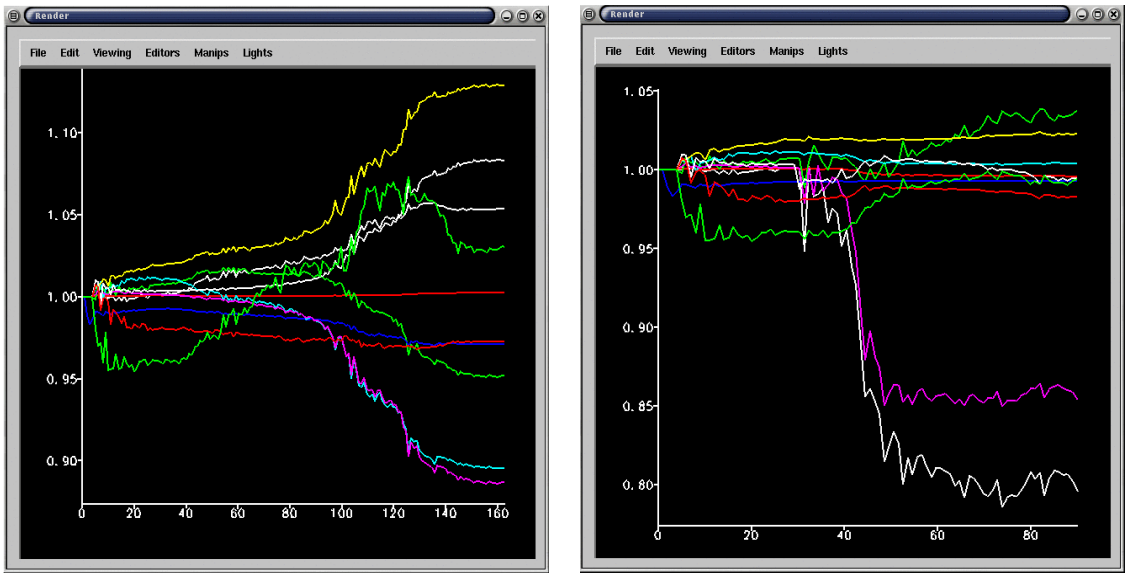


Figure 5: Friction errors for all cases considered. The 2-d mesh shows the experimental friction values against the slide-to-roll ratio with the displacement of the surface in the third dimension representing the error in the numerically calculated friction for the best simplex point.

5 Visualisation

The full optimisation run generates hierarchies of multivariate data. Each EHL simulation is itself reduced to just one number, F_j^{num} from Equation (1). The distance each of these calculated values is away from F_j^{exp} is interesting to the users at Shell in assessing the convergence of the solver. This kind of information is one obvious area in which user steering may help, for example if the results were all good except at, say, very high ambient temperatures then previous knowledge could accelerate the optimisation process. These results are shown in Figure 5 which shows a 2-d plane with increasing slide to roll ratios plotted against experimental friction for each of the loadings and ambient temperatures. The 3-d surface represents the errors in each of the calculated friction values. If a perfect solution was found this would collapse to be co-planar with the six lines of experimental results.

The progress of the optimiser itself is shown in the other outputs available. The useful information is



(a) No steering (b) New simplex after 30 iterations

Figure 6: Progression of optimiser showing relative change of best solution found to initial guess. Each line represents a different variable from Table 1.

mainly focused on the best data set found thus far. This multidimensional data cannot be represented as easily as tracking the movement of a point in 2-d or 3-d space, and instead other techniques are required to sensibly assess the progress of the optimiser. Two alternative visualisations of this data are currently provided to the user. The first is shown in Figure 6 where the y-axis represents the relative change from the initial estimate for each of the ten variables, with progression along the x-axis being the incremented for improvements in the $\mathscr{R}_F$ value. In Figure 6 two different graphs are shown. The first has the optimiser progressing without any steering, the second has a new simplex formed after the 30^{th} improvement to the best point in the simplex. It can be clearly seen how this has encouraged the optimiser to a very different point in the search space.

The second visualisation of the best simplex values found uses parallel coordinates [7]. The choice is available to the user of whether to use conventional 2-d parallel coordinates, or to stretch the convergence of each variable out into the third dimension as shown in Figure 7. Parallel coordinates can help to see dependencies between variables.

Given the method of Nelder and Mead [9] moves the worst point in the simplex, this may mean that the best point is not constantly updated, it is also of interest to know how good the other points being tested are. With so many calculations taking place it would not be sensible or useful to try to visualise all of the quantities evaluated. Instead we have used 2-d plots of all the $\underline{x}$ points tested for user-selected

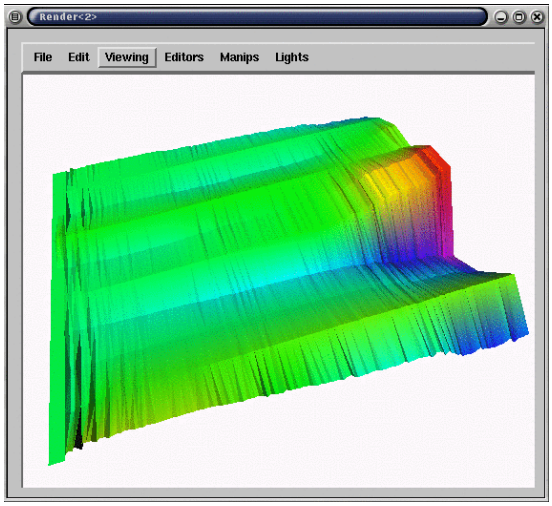


Figure 7: 3-d Parallel coordinates showing progression of the optimal solution found with increasing quality of solution from left to right. Each component of the solution is a separate 2-d graph with unit spacing into the picture. The connecting surface is designed to help visualise dependencies between variables

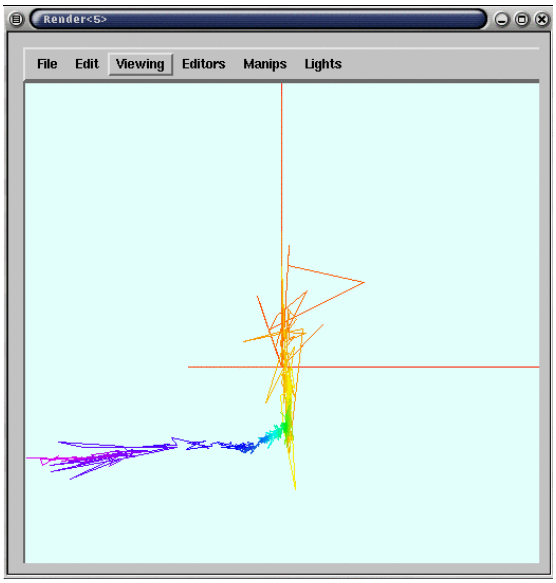


Figure 8: Plot of how two user selected components of $\underline{x}$ have been varied in search space. Red indicates the first directions tested with blue and magenta representing the latest calculations. The initial tests perpendicular to each other are clearly visible.

pairs of axes. These can either be coloured by the quality of the solution, i.e. the $\mathscr{R}_F$ value, as shown in Figure 8, or by how recently they were visited. These visualisations enable better understanding of the search space.

6 Conclusions and Future Work

In this work we have shown how an industrial serial optimisation code requiring many individual numerical calculations at every stage can be parallelised, Grid-enabled and embedded within a PSE. It has been seen that the gViz libraries have handled all the communication between the PSE and the simulation effectively. Use of MPI within the NAG libraries has been incorporated and consideration has been given to the most efficient methods of continuation for these problems. It was found that combining the maximum levels of continuation gave solutions significantly faster. They also found better solutions in terms of minimising the friction residual.

The visualisation output has been driven by the needs of the users and is used to help steer the simulation, for example by manually moving away from local minima or adding additional layers of detail or computational accuracy when near to a local minimum.

There are many future directions for this work. Potentially the most important is expanding the numerical solver used. Much work into individual EHL problems is in 2-d point contact cases but up to now they have always been far too expensive to even consider solving thousands of times. With the advent of faster computers and the evolution of parallel implementations of the 2-d EHL codes [6] it is now feasible for the individual EHL simulations to be undertaken in parallel, leading to a hierarchy of parallelism.

From a Grid perspective other work that ought to be undertaken concerns security. The friction and lubricant data used in these simulations is commercially sensitive and so secure methods of communicating this to and from remote Grid resources must be considered.

The other necessity for future expansion concerns more general book-keeping. When multiple simulations are launched or a new user wants to join in, they must know to which part of which resource to be pointing their PSE. One possible method of accomplishing this is by new simulations registering with a *web service* when they start up. A directory of running simulations can then be kept from which potential users can select which simulation on which resource they wish to connect.

Another issue concerning collaborative PSEs is that any user joining a simulation which has previously been steered will not know what changes were made and when, since only the present input dataset and the entire output data are currently provided. To be able to repeat the experiment such information would also need to be accessible.

Acknowledgements

The authors wish to thank the DTI and EPSRC for funding this work with Shell Global Solutions through Core Programme e-Science grant number GR/S19486/01.

References

[1] 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.

[2] Dantzig G.B. *Linear Programming and Extensions*. Princeton University Press, 1963.

[3] Fairlie R., Goodyer C.E., Berzins M. and Scales L.E. 'Numerical modelling of thermal effects in elastohydrodynamic lubrication

solvers.' In: D. Dowson *et al.* (editor), *Tribological Research and Design for Engineering Systems, Proceedings of the 29th Leeds-Lyon Symposium on Tribology*. Elsevier, 2003.

[4] Goodyer C.E. *Adaptive Numerical Methods for Elastohydrodynamic Lubrication*. Ph.D. Thesis, University of Leeds, Leeds, England, 2001.

[5] Goodyer C.E. and Berzins M. 'Eclipse and Ellipse: PSEs for EHL solutions using IRIS Explorer and SCIRun.' In: P.M.A. Sloot, C.J.K. Tan, J.J. Dongarra and A.G. Hoekstra (editors), *Computational Science, ICCS 2002 Part I, Lecture Notes in Computer Science*, Vol 2329, pp. 521–530. Springer, 2002.

[6] Goodyer C.E., Wood J. and Berzins M. 'A parallel Grid based PSE for EHL problems.' In: J. Fagerholm, J. Haataja, J. Järvinen, M. Lyly, R. P. and V. Savolainen (editors), *Applied Parallel Computing, Proceedings of PARA '02, Lecture Notes in Computer Science*, Vol 2367, pp. 523–532. Springer, 2002.

[7] Inselberg A., Chomut T. and Reif M. 'Convexity algorithms in parallel coordinates.' *Journal of the ACM*, Vol 34(4), pp. 765–801, 1987.

[8] Message Passing Interface Forum. 'MPI: A message-passing interface standard.' *International Journal of Supercomputer Applications*, Vol 8(3/4), 1994.

[9] Nelder J.A. and Mead R. 'A simplex method for function minimization.' *Computing Journal*, Vol 7, pp. 308–313, 1965.

[10] Parkinson J.M. and Hutchinson D. 'An investigation into the efficiency of variants on the simplex method.' In: F.A. Lootsma (editor), *Numerical Methods for Non-linear Optimization*, pp. 115–135. Academic Press, 1972.

[11] Venner C.H. and Lubrecht A.A. *Multilevel Methods in Lubrication*. Elsevier, 2000. ISBN 0-444-50503-2.

[12] Walkley M.A., Wood J. and Brodlie K.W. 'A distributed collaborative problem solving environment.' In: P.M.A. Sloot, C.J.K. Tan, J.J. Dongarra and A.G. Hoekstra (editors), *Computational Science, ICCS 2002 Part I, Lecture Notes in Computer Science*, Vol 2329, pp. 853–861. Springer, 2002.

[13] Walton J.P.R.B. 'Now you see it – interactive visualisation of large datasets.' In: C.A. Brebbia and H. Power (editors), *Applications of Supercomputers in Engineering III*. Computatational Mechanics Publications / Elsevier Applied Science, 1993.

[14] Wood J.W., Brodlie K.W. and Walton J.P.R. 'gviz: Visualization and computational steering for e-Science.' In: *Proceedings of the All Hands Meeting 2003*. 2003.