EMPIRE optimization and interfacing

Peter Jan van Leeuwen, Phil Browne, and David Scott

Data assimilation can be defined as Bayesian Inference for the geosciences. It is an essential part of any prediction system as it is a mechanism to generate the best starting point of the forecast based on both the numerical model and all observations available. As such it is used in all branches of the geosciences, with varying sophistication. Most advanced are weather prediction applications, where the system dimension is very high, currently over 10^9 dimensional state vectors and very short run times (a new 5 day forecast has to be ready every 6 hours) so extremely efficient algorithms are needed. These codes grow over time and are very hard for academics to use. To this end we developed the extremely efficient and easy to use EMPIRE software system for academic use. Interestingly, the weather forecast centres like Met Office and the European Centre for Medium Range Weather Forecasting ECMWF have shown considerable interest in this development. This project is about making EMPIRE more efficient in terms of random number generation and linear solves, which are the current performance bottlenecks.

Introduction

Data assimilation is the science of combining observations of a system with a numerical model of that system in order to improve the model forecast, to obtain a better description of the system, or to directly improve the model. It has a firm mathematical basis in Bayes theorem, and explores methods like the Kalman filter and particle filters, and methods from optimal control, inverse problems, and machine learning. Quite often the applications are very high dimensional, up to 10^9 at present for numerical weather prediction, asking for very efficient methods and codes. The most efficient methods that provide best estimates including uncertainty are based on Monte-Carlo integrations. It turns out that the data-assimilation algorithms can be coded almost as black boxes for the different models, from weather forecasting and other geophysical applications to biology and neuroscience. This motivated the formulation of the data-assimilation software package EMPIRE that encodes several state-of-the-art data-assimilation algorithms in a very efficient way. This development allows researchers in all the application fields to concentrate on their science without having to reinvent the wheel. Through the National Centre for Earth Observation the EMPIRE system is maintained and developed further into a national tool for the geosciences and beyond.

This project consists of two parts, the first on more efficient normal random number generation, and the second on coupling EMPIRE to PETSc for more efficient linear solves.

Efficient random number generation
As part of it's sequential methods for DA, a huge amount of stochastic perturbations are required. As an example, running a 32 member ensemble of climate models required the generation of around 1trillion normally distributed random numbers. [32*2314430*180*72=959840409600]

The system used to rely on a random number generator that was downloaded from the internet without regard for it's efficiency. Until now, other operations were substantially more costly than the random number generations. Following on from other algorithmic developments, the optimization of the generation of random numbers is now a priority for large-scale use on ARCHER.

The first part of the project was tasked with ensuring the optimality of the random number generation that is used. Specifically, vectorisation of these would like to be achieved as they are currently sequentially generated (within an MPI thread). There are complications with the parallelisation inherited by the code structure as the code is a hybrid MPI/OpenMP implementation. Hence, ensuring independence across MPI threads was crucial.

An existing implementation of the Ziggurat pseudo random number generator was modified to exploit the vector registers present in modern chips such as the Intel Sandy Bridge processor. As the length of a vector register varies between processor types the code has to compile time parameters that may be used to tune the code for different register lengths.

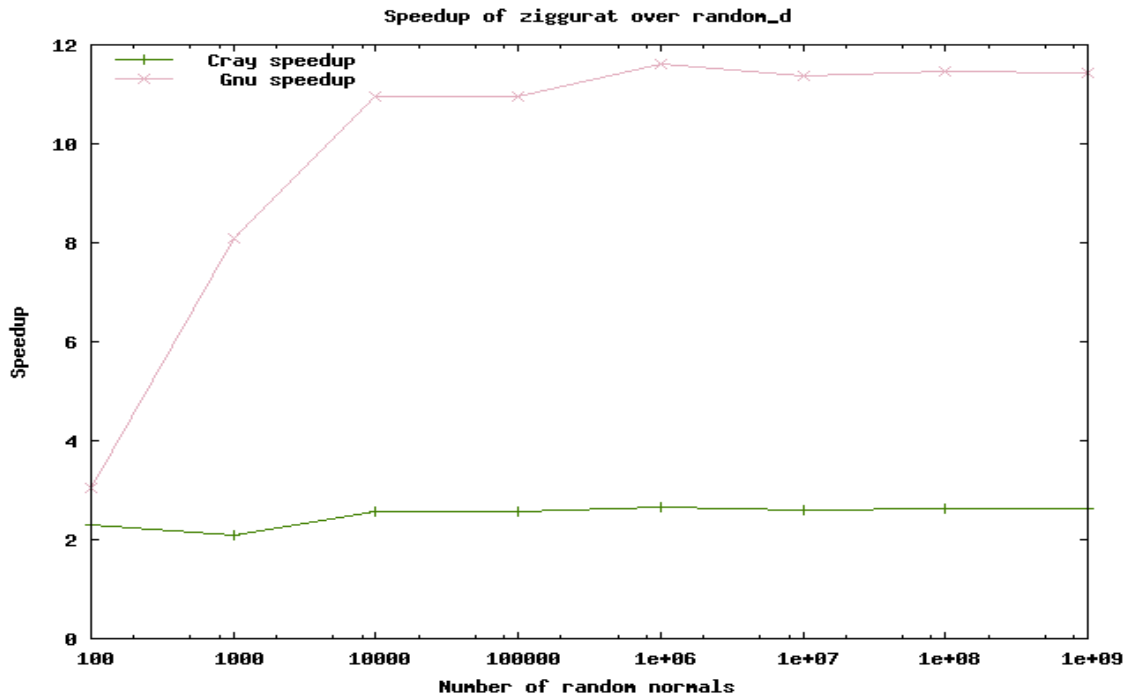

Figure 1: Speed up in normal random number generation compared to the old algorithm for the Cray and the Gnu compiler as function of the number of random numbers generated.

The scaling of the random-number generation with the number of processors was also tested, and perfect scaling has been achieved, as depicted in figure 2
.

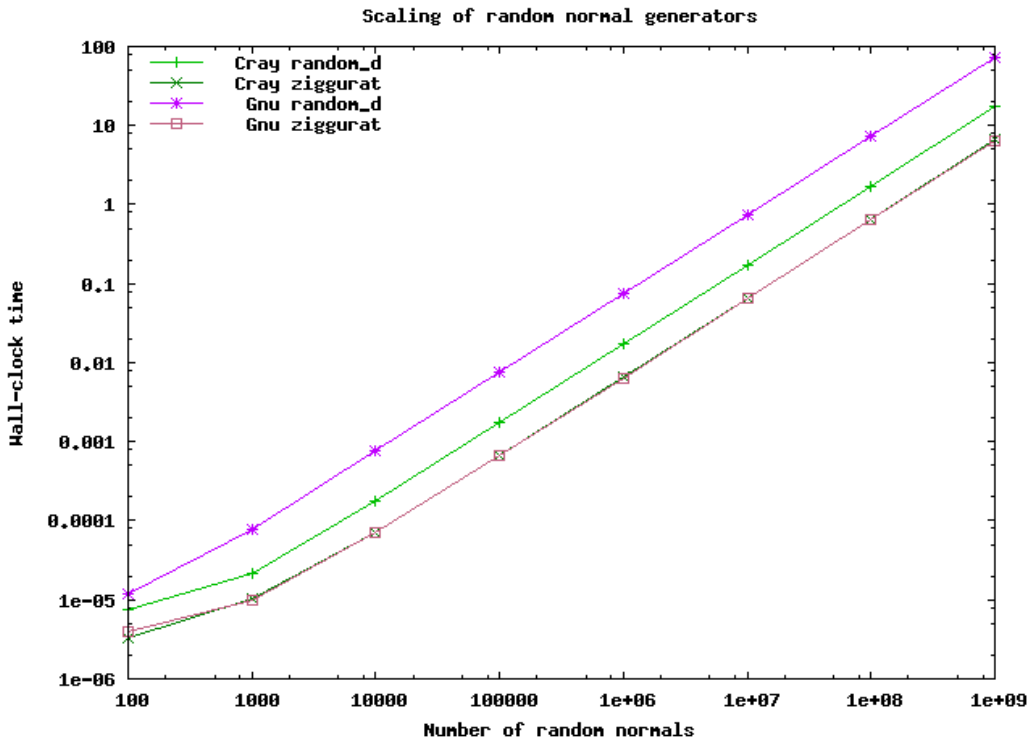

Figure 2: Scaling of random number generation algorithms.

Each MPI process requiring random numbers has its own copy of the random number generator which is seeded with a number derived from the process's ID. The code as used in EMPIRE generates four 32bit pseudo random integers at a time. This is possible as the Sandy Bridge processors which make up ARCHER's compute nodes have 128bit registers. The random numbers are stored in an array and supplied to the calling MPI process as required. When all four integers have been used the next request for a random number triggers the generation of four more.

The new algorithm was found to achieve significant speedup compared to the existing algorithm within the EMPIRE software framework, as demonstrated in figure 1. This shows that a huge speed up has been achieved, as much as 11x faster when using the GNU compiler, and a factor of 2.5 using the Gray compiler.

This algorithm and hence speedup will be adopted by many users of ARCHER and other HPC systems in areas including meteorology, climate science, space weather, oceanography and neuroscience.

We also investigated the quality of the normal random numbers. Figures 3 and 4 show that the random numbers follow the Gaussian distribution which we require, the classic "bell-shaped" curve. Figure 5 shows the difference in the

distributions with different compilers and algorithms. These are precisely within the bounds we would expect with the finite sample size we have used to generate the distributions.

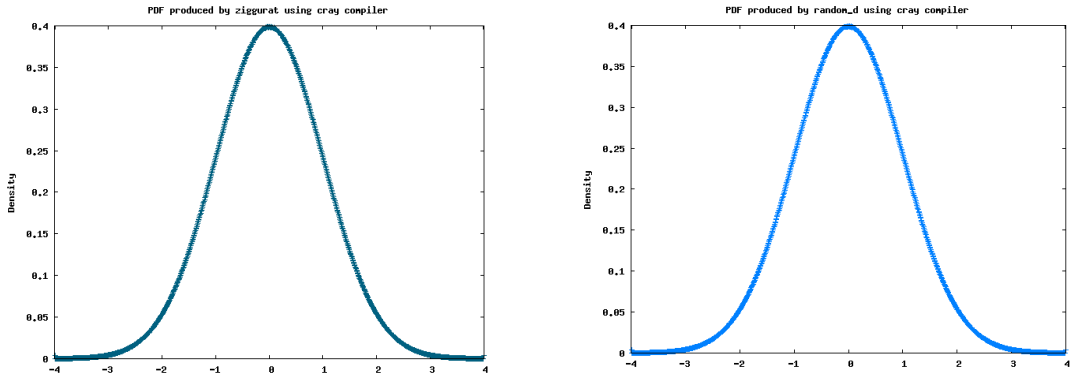

Figure 3: Gaussian distributions generated by the ziggurat and random_d algorithms using the cray compiler.

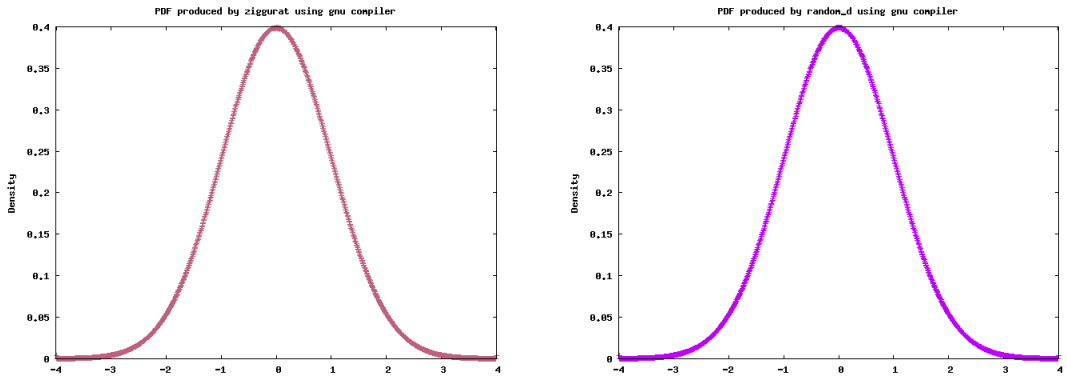

Figure 4: Gaussian distributions generated by the ziggurat and random_d algorithms using the gnu compiler.

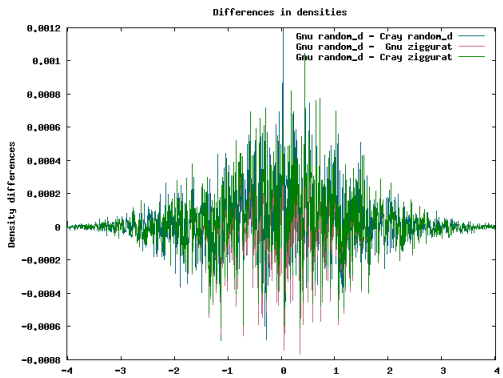

Figure 5: Differences in the randomly generated distributions with the ziggurat algorithm and the cray compiler, compared with the original random_d algorithm using the gnu compiler.

Efficient linear solves

EMPIRE is designed as a general-purpose data assimilation software package. Many of these data-assimilation algorithms are dependant on linear solves, and the efficiency of these can be the bottleneck in the code.

Therefore, part 2 of the project investigated a usable way to interface with numerical linear algebra software such as PETSc. Linking EMPIRE to PETSc to leverage the highly optimized numerical linear algebra such as Krylov methods and algebraic multigrid methods will allow new data assimilation algorithms to be implemented.

For example, EMPIRE has included within it an Ensemble Transform Kalman Filter (ETKF) which calls LAPACK to perform an SVD to do a linear solve. The more ubiquitous Ensemble Kalman Filter (EnKF) has an update step of the form:

$$X_{n+1} = X_n + K\,b$$

where $K = P_f H^T (H P_f H^T + R)^{-1}$

and hence could be implemented should an efficient method for the linear algebra be found.

This structure of this problem is pervasive in data assimilation, and hence such linear algebra could be used for implementing Kalman filters, Ensemble Kalman smoothers, optimal interpolation etc., including fully nonlinear methods, and thus giving the whole user community access to new methods which have the potential to improve their science.

Here, the data structure of EMPIRE has proved to be a challenge to the integration of the PETSc library. PETSc defines the manner in which it expects its matrices (or matrix operators) to be stored in distributed memory. EMPIRE by necessity has partial information (either matrix entries or vectors in a low rank decomposition) on each MPI process. Further, each MPI process may be required to do the linear algebra operation with its own right hand side data.

PETSc (the Portable, Extensible Toolkit forScientific Computation) facilitates the solution of sets of linear (and non-linear) equations. Usually the linear operators are represented by matrices but it is possible to represent them by functions which operate on vectors. This is referred to as a matrix free formulation.

If matrices are used then PETSc can construct various preconditioners from the matrices. Using the matrix free approach this is not possible and the user has to supply the preconditioner. Before the work being described here was carried out EMPIRE already used its own matrix free approach. The user was required to supply a number of operators represented as functions on vectors. Amongst these were an operator called R and its inverse InvR. Fortunately InvR is usually a good preconditioner for the problem to be solved so it is possible to use PETSc to carry out the linear solves. This involves writing versions of the operators that act on PETSc data structures (Vecs) and wrapping them up so that they can be used by PETSc pretty much as matrices would be. Finally these wrapped

operators are registered with the linear solver in place of the usual matrix representations.

Specifically, the linear algebra equation

$$(HQH^T + R)\, x = b$$

was solved to test the PETSc integration. To this end we use $R = 0.1I$, with I the identity matrix, H is a projection operator into the first half of the state and matrix $Q = 0.2I$. PETSc was employed in matrix-free mode. The solver was GMRES, preconditioned by R^{-1}.

With random right-hand sides, out of 10 tests, only a single test had a residual > 10^{-15}. Preconditioning by R^{-1} is typically what we will use in most applications as, in the case of uncorrelated observation errors, R is diagonal hence very easily inverted. These results show that the coupling has been very successful and future EMPIRE users will be able to exploit this efficiency in full.

Conclusions

This project has been highly successful and all its objectives have been achieved within the allocated time. Specifically, the new normal random number generation has been sped up by a factor 2.5-11, resulting in a total speed-up of the code for high-dimensional applications by a factor 2-9. Furthermore, the coupling of PETSc to EMPIRE will allow for much more efficient linear solves. No benchmark figures are available for the latter as this will be highly problem dependent, but it is allowing us to encode a much larger class of data-assimilation methods efficiently within the EMPIRE software system. Both improvements will be felt immediately by the fastly growing user group of the system.

Acknowledgements

This work was funded under the embedded CSE programme of the ARCHER UK National Supercomputing Service (http://www.archer.ac.uk)