Grid Enabling Geographically Weighted Regression

Daniel J Grose[1], Richard Harris[2], Chris Brundson[3], and Dave Kilham[2]

[1]Centre for e-Science , Lancaster University , United Kingdom
[2]School of Geographical Sciences , University of Bristol , United Kingdom
[3]Department of Geography , University of Leicester , United Kingdom

April 25, 2007

1 Introduction

The application of high performance computing to spatial analysis has long been of interest to geographical scientists and has spearheaded research in computations. Of particular note is the pioneering work undertaken by Stan Openshaw at the University of Newcastle and at the Centre for Computational Geography at Leeds University, of which an exemplar is the Geographical Analysis Machine [11]. More recently, Martin [10] has identified the potential for geocomputation to develop under the rubric of high performance computer (grid) networks and e-(electronic) social science. He identifies four essential research issues for e-social science: automated data mining; visualization of spatial data uncertainty; incorporation of an explicitly spatial dimension into simulation modelling; and neighbourhood classification from multi-source distributed datasets.

Missing, perhaps, from Martin's list is the use of computational grid to 'speed up' the repetitious processes of many spatial statistics. What GAM, GWR (see below) and other methods of spatially localized analysis have in common is a general sequence of

1. calibrating the size of the kernel or search window to the amount of spatial autocorrelation found in the attributes of the data being examined

2. creating spatially overlapping subsets of the data to reflect this
3. allowing the kernel to pass from one subset to the next, applying a statistical test in each
4. simulating confidence intervals for the statistical result by detaching the data attributes from the geographical coordinates at which they were captured, then repeatedly reattaching the attributes to randomly selected locations and applying the test again.

If a study region is divided into n overlapping grid squares then the first and third stages of the sequence are completed by allowing the kernel to expand from a minimum to a maximum width through z increments and determining an optimal result. This requires $n \times z$ processes. For the fourth stage (and assuming the kernel size is now fixed), the data are redistributed m times, requiring a further $n \times m$ tests. In total, then, the method invokes approximately $n(z + m)$ processes (plus others to subset the data, run the statistical tests and so forth).

The attraction of high performance computing and, in particular, the use of parallelisation, arises from the course granularity of the overall sequence of events (granularity being the size of computation that can be performed between communication or synchronization points [9] [12]. For many spatial statistical procedures, each of the stages of calibration, fitting and assessing significance can be parallelised with processes that will operate without communication to the others (since, for example, the outcome of a model fitted to one spatial subset of the data does not affect or modify the outcome of a model fitted to another). In principle, each of the $n(z + m)$ processes can be sent to separate computational nodes; their outputs need only be pooled and assessed once the results have been established.

2 Geographically Weighted Regression (GWR)

Geographically Weighted Regression (GWR) builds on traditional linear regression methods by permitting the relationships between variables to vary spatially. This is achieved by allowing the linear predictor to be a function of the spatial coordinates (u, v) as follows.

$$\eta = \sum_{j=1}^{p+1} \beta_j(u, v) x_j \tag{1}$$

Here the x_j are the $p+1$ explanatory variables and the $\beta_1(u, v)$,$\beta_2(u, v)$,...,$\beta_{p+1}(u, v)$ are functions (assumed continuous) in u and v. Thus, given a set of observations $\{y_i\}$ corresponding to the set of realisations of the dependent variables

$\{(x_{i,1}, x_{i,2}, \ldots, x_{i,p+1})\}$, it is necessary to determine the $\beta_j(u, v)$. To this end GWR assumes that the $\beta_j(u, v)$ exhibit little variation close to a given y_i that is to be estimated using standard (but weighted - see below) regression methodology. This allows the $\beta_j(u, v)$ to be approximated using the $i \times j$ constants $\beta_{i,j}(u_i, v_i)$ that are evaluated in the vicinity of the y_i using standard regression methodology.

As noted in [6], the assumption of constancy in the parameters in the proximity of y_i provides a straightforward means of estimating the $\beta_j(u, v)$ but results in a bias in the local regression coefficients. After all, if the relationship between y and the x_j varies continuously across space then this must be true even within the vicinity of the point of interpolation. To reduce the effect of bias, the contribution of sample points in the local regression model are weighted according to their proximity to y_i. Typically, the weighting function is parameterised, and the GWR methodology employs a calibration process which sets out to calculate the parameters of the weight functions so as to form an appropriate trade off between bias and standard error in the prediction of the overall model (it clearly is necessary to use some data around the point of interpolation to estimate its value: too few points and the estimate will lack precision; too many points will smooth out the local relationship).

3 Scaling GWR to the Grid

The calibration of the weight function employed by GWR determines the number of neighbouring points of y_i to incorporate in the local regression model. This can be defined by the number of neighbours that have to be considered or by defining a radius about y_i from within which points should be included (i.e. either the number of neighbours or the geographical space around y_i may be fixed). In either case, it is necessary to determine (and rank) the inter point distances either for the complete data set being considered or to use a method of spatial indexing. Though more computationally demanding, calculating the distances for the complete data set retains the greatest flexibility when applying GWR. For n data sample points, this operation is $O(n^2)$. Even ignoring the computational cost associated with determining the parameters of the weight functions and performing the local regression for each data point, the method does not scale well with n. Figure 1 shows the CPU time for calculating the inter point distances vs the number of points in a data set. The results were obtained by using the **spDistsN1** function from the R sp package which is employed by the GWR package for

R, spgwr[1]. Clearly, if large data sets are to be considered, either a differ-

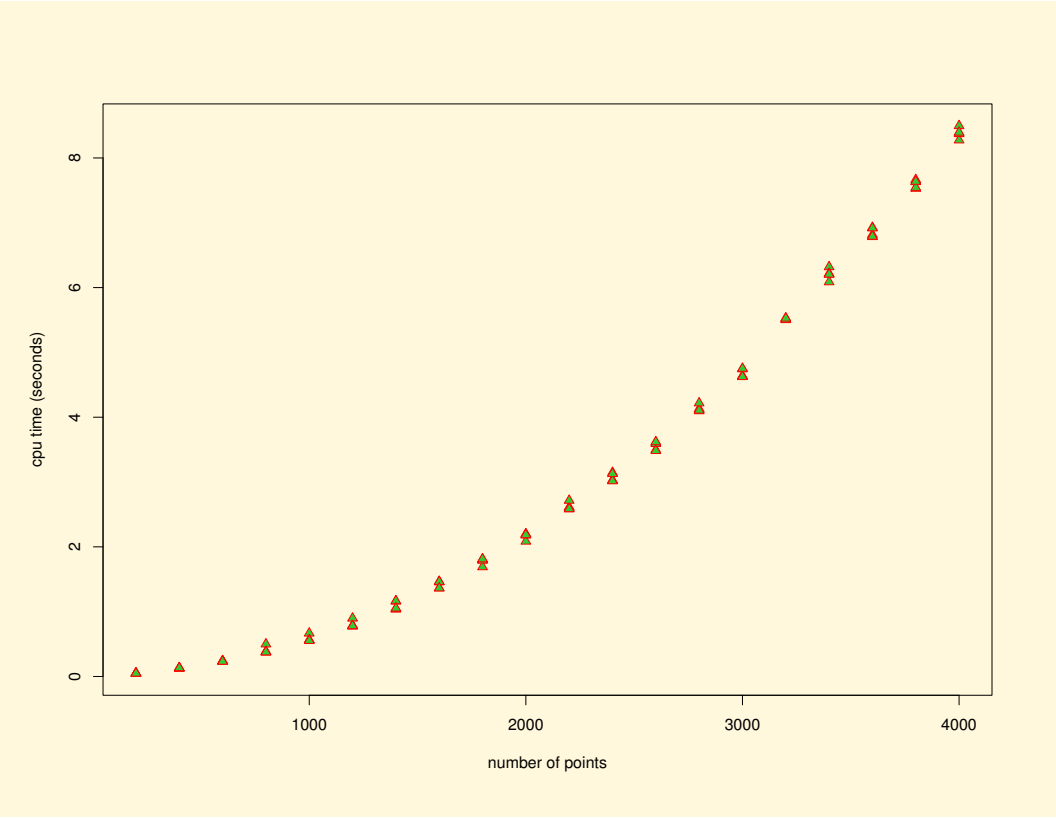

Figure 1: $O(n^2)$ nature of nearest neighbours calculation.

ent modelling approach has to be adopted or alternatively, the use of High Performance Computing (HPC) facilities are required [5].

Fortunately, the methods employed by GWR are relatively simple to adapt to large HPC systems on a computational grid. For example, in the case of calculating the inter point separations, an existing k Nearest Neighbours algorithm can be employed which readily divides across multiple compute nodes [7]. For the algorithm employed in [7], given n data points and m processors, the compute time τ reduces as

$$\frac{\tau(m)}{\tau(1)} = \frac{n(2 - \frac{1}{m}) - 1}{m(n-1)} \tag{2}$$

When the number of data points is much greater than the number of processors i.e. when $n >> m$, (which is the normal case), then

$$\frac{\tau(m)}{\tau(1)} \simeq \frac{2}{m} \tag{3}$$

[1]Tests were run using R on a single processor Intel processor PC

In a test case of $n = 200000$ and $m = 100$ the average user time per process was 885 seconds and required 1.5Gb of memory. These requirements are well with the scope of most modern HPC facilities, including the UK's developing computational grid infrastructure (The National Grid Service).

4 Deploying GWR on the Grid

The nature of the GWR methodology makes implementing it on parallel HPC architectures reasonably straightforward. However, although research communities are adopting the use of Grid based HPC resources, the barriers of entry still remain high [1]. This is primarily due to the need for most users to interact directly with the middleware (for example, the Globus toolkit, OpenSSH and myproxy) that are used to access the Grid. Although there are a number of interfaces specifically designed to simplify the process of job submission and monitoring (GridSAM, AHE [2] etc), none of these are application domain specific. One approach to make methodologies available on the Grid more accessible is a to provide a Service Oriented Architecture (SOA) for the methodologies. This enables the services to be accessed directly from client systems via interfaces which reflect the context of the application domain. Furthermore, since much software used by researchers is extensible in some way, provision of a SOA allows the client to access the Grid from within a familiar environment.

Steps toward providing an architecture to host SOA's have been taken as part of several projects, for example the GROWL (Grid Resources on a Work Station) library [8] which has been used to host applications within several client environments such as R [4] [3].

4.1 Providing SOA's using the GROWL client server architecture

Amongst other things, GROWL provides for a three tier client/server architecture which can be used to support the development of SOA's
(figure 2). The first tier consists of the client interfaces, specialised to specific client application requirements, and integrated into the client server architecture using modules created from service interface definitions published in an Interface Definition Language (IDL) file. The IDL files are generated and published by the developers of the services represented in the third tier of the architecture.

The second tier consists of the GROWL server. The main functions of the server are threefold:

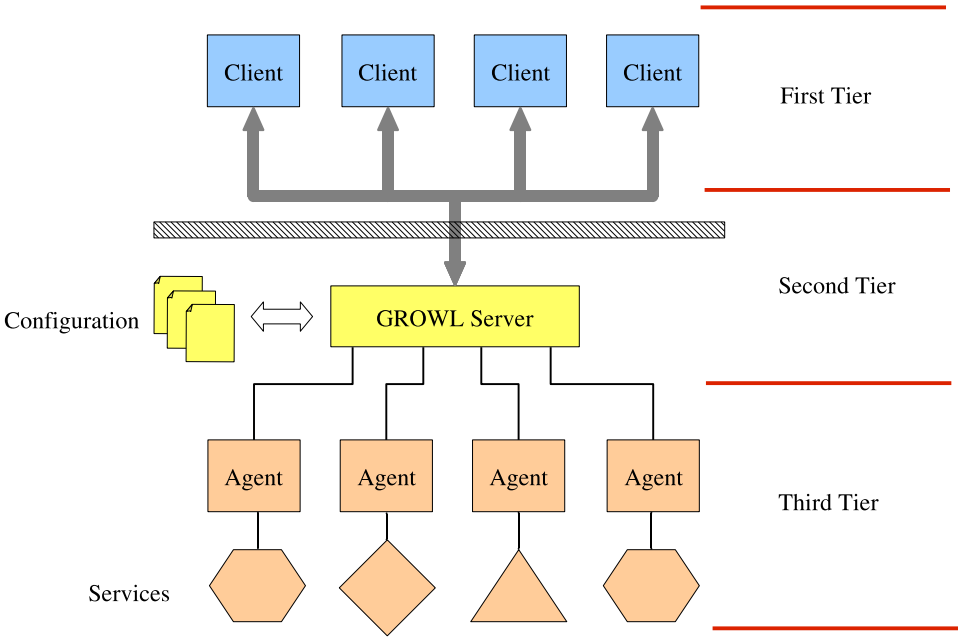

Figure 2: GROWL Client - Server Architecture

1. Authentication of clients (using Distinguished Names obtained from the clients certificates).

2. Hosting services by acting as a proxy for the service interface.

3. Mapping of client requests to specific service instances.

The third tier consists of the services themselves. A service is defined by its interface which is published using the IDL. Importantly, an individual service may be implemented in a number of different ways, the particular implementation varying according to the requirements of the system(s) hosting and the client(s) accessing it. It may, of course, also vary in time as the requirements of a service implementation change. Services are created by service developers and the interface definitions for a service can be created automatically from existing code using GROWL utilities.

The key advantages of this three tier client/server architecture are:

- Clients, server and services may be upgraded or replaced independently.

- A single interface may correspond to many service implementations.
- All services are accessed via a single (secure) port.
- Services have persistence. This is important since the services are used asynchronously.
- Developers of client applications can program against an interface in a language and platform independent manner and need no understanding of the service logic.
- Developers of services need not be aware of the client application logic and do not require an understanding of web services.

5 Summary

The full paper demonstrates how the existing spgwr package for R has been adapted for use on HPC resources on the Grid and the performance of the package is analysed in detail. The manner in which the GWR methodology is hosted as a service on the Grid using a SOA is detailed. Furthermore, the SOA is employed to provide an R package for accessing Grid based GWR from a remote client system .

6 Acknowledgments

This research is funded by the National Centre for e-Social Science, Small Grants Projects, RES-149-25-1041. The authors of the spgwr package are Roger Bivand and Danlin Yu.

References

[1] Johnathon Chin and Peter Coveny. Towards tractable toolkits for the grid: a plea for lightweight, usable middleware. Reality Grid, June 2004. http://www.realitygrid.org.

[2] P.V. Coveney, R.S. Saksena, S.J. Zasad, M. McKeown, and S. Pickles. The application hosting environment: Lightweight middleware for grid-based computational science. *Computational Physics Communications*, 2007. In press.

[3] Rob Crouchley, Ties van Ark, John Pritchard, John Kewley, Rob Allan, Mark Hayes, and Lorna Morris. Putting social science applications on the grid. In *First International Conference on e-Social Science*. National Centre for e-Social Science, June 2005. http://www.ncess.ac.uk/events/conference/2005/.

[4] Daniel J Grose et al. sabreR : Grid-enabling the analysis of multi-process random effect response data in R. In *Second International Conference on e-Social Science*. National Centre for e-Social Science, June 2006. http://www.ncess.ac.uk/events/conference/2006/.

[5] Richard Harris et al. Developing Grid enabled spatial regression models. In *Second International Conference on e-Social Science*. National Centre for e-Social Science, June 2006. http://www.ncess.ac.uk/events/conference/2006/.

[6] A. Stewart Fotheringham, Chris Brundson, and Martin Charlton. *Geographically Weighted Regression - the analysis of spatially varying relationships*. Wiley, 2002. ISBN 0-471-49616-2.

[7] D.J. Grose. Testing the north west grid using a k nearest neighbours algorithm. Technical report, North West Grid, November 2006. http://www.nw-grid.ac.uk/?q=nodes/uola/testbed/.

[8] Mark Hayes, Lorna Morris, Rob Crouchley, Daniel Grose, Ties van Ark, Rob Allan, and John Kewley. Growl: A lightweight grid services toolkit and applications. In Simon Cox and David W. Walker, editors, *Proceedings of the UK e-Science All Hands Meeting*. EPSRC, September 2005. epubs.cclrc.ac.uk/bitstream/920/460.pdf.

[9] I. Lumb. Hpc grids. In A. Abbas, editor, *Grid Computing: a Practical Guide to Technology and Applications*, pages 119–33. Charles River Media, Hingham, MA, 2004.

[10] D. Martin. Socioeconomic geocomputation and e-social science. *Transactions in GIS*, (9):1–3, 2005.

[11] S. Openshaw, M. Charlton, C. Wymer, and A.W. Craft. A mark i geographical analysis machine for the automated analysis of point datasets. *International Journal of Geographic Information Systems*, (1):335–58, 1987.

[12] B. Wilkinson and M. Allen. *Parallel Programming: techniques and applications using networked workstations and parallel computers*. Prentice Hall, Upper Saddle River, NJ, 1999. ISBN = 1-314-0563-2.