Some Grid Enabled Tools for Statistical Research

Rob Crouchley, Dan Grose, John Pritchard and Dave Stott

Centre for e-Science, Lancaster University, C Floor, Bowland Tower South
Lancaster, LA1 4YT, UK

Contact e-Mail: r.crouchley@lancaster.ac.uk

Abstract

Tools for computationally demanding statistical research are becoming available as part of commercial systems, e.g. SAS grid computing and Stata MP. However, these systems can be of limited use on a public grid, e.g. Stata MP can't access multiple data sets and neither system provides access to their source code. Furthermore, there are no plans to install them on the UK National Grid Service (NGS) because of cost/licensing issues.

We use R as the framework to enable statistical modelling on the grid because:

- R is an effective, efficient and easy to use tool for Statistical Modelling
- Many existing tried and tested statistical methods already available for R and can easily be modified to exploit the benefits of grid computing
- Work flows to support the modelling process are simple to create
- R is easy to install on most popular operating systems (Windows, Unix, OSX) and can be used directly from a USB memory stick
- R includes a programming environment, which when used in conjunction with our multiR package, automatically provides a data centric scripting tool for grid computing
- There are no licensing issues

We use 2 examples to illustrate how easy it has become to perform computationally demanding analyses from R on a Windows XP PC or similar.

Enabling Technology

The tools we have developed are: (1) multiR (coarse grained parallel job submission) which can be used in the model exploration/checking stages; (2) sabreR and sabreRGrid (fine grained parallel Sabre job submission) which can be used to estimate computationally demanding event history models. All multiR and sabreRgrid require is:

1. An internet connection
2. The installation of our multiR and sabreRGrid packages for R
3. A certificate to identify the client to the host – typically a grid certificate

Importantly there is no need for users to install or have familiarity of Globus, VDT, gsissh, gsiscp, grid-ftp, grid-proxy tools or any other GRID related software. To the user there is very little difference between using the Sabre library from within R on the desktop, and using Sabre for statistical modelling on the grid from within R.

Example 1

In the 1st example we illustrate how to use our R library, multiR, to bootstrap a model of car ownership rates from the 2001 Census of England and Wales.

The text below illustrates the difference between a serial desktop evaluation and the submission of a High Throughput Distributed Computing (Grid) job of each calculation using multiR from the desktop. The example is adapted from Crawley (2007, p523). In the analysis we regresses car ownership (proportion of households without a car or van in the census output areas) against the `x1` (proportion of persons of working age) `x2,` (proportion households in public housing), `x3` (proportion of Households that are lone parent households), `x4` (proportion of persons 16+ that are single), `x5` (proportion of persons that are "white British").

The data set consists of 159,972 observations. Time to perform regression analysis is approximately 1.4 seconds. Thus to bootstrap using 10,000 replicates = 14,500 seconds or a little over four hours. Time taken to go parallel on the NGS node at Lancaster University (32 processor system) is about 11 minures.

In the parallel code example below, a different seed is specified for each processor as it is possible that the same random sequence might be used on some of the processors when parallel jobs are started almost simultaneously. There are more robust ways of ensuring independent random number sequences for each processor, but for clarity this is not demonstrated here. We break the job up using a function wrapper (`bootstrap.coeffs`) so we can simplify the task of obtaining 10,000 replicates. This function allows you to group the individual replicates together so as to optimise the use of distributed resources (in this case 100 groups of 100 replicates).

```
*******************   serial code  ***********************

england.wales<-read.table("england.wales.tab")
attach(england.wales)
model<-lm(y~x1+x2+x3+x4+x5)
yhat<-fitted(model)
residuals<-y-yhat
coeffs<-matrix(nrow=10000,ncol=6)
for(i in 1:10000)
{
  shuffled<-yhat+sample(residuals)
  boot.model<-lm(shuffled~x1+x2+x3+x4+x5)
  coeffs[i,]<-coef(boot.model)
}
apply(coeffs,2,mean)
apply(coeffs,2,sd)


*******************   parallel code  **********************

library(multiR) # load the multiR library
session<-multiR.session.dlg() #see example 2 for the dialogues
```

```
bootstrap.coeffs<-function(data,yhat,residuals,sample.size,seed)
{
   set.seed(seed)
   coeffs<-matrix(nrow=sample.size,ncol=6)
   for(i in 1:sample.size)
    {
      shuffled<-yhat+sample(residuals)
      boot.model<-lm(shuffled~x1+x2+x3+x4+x5,data=data)
      coeffs[i,]<-coef(boot.model)
    }
   return(coeffs)
}

england.wales<-read.table("england.wales.tab")
attach(england.wales)
model<-lm(y~x1+x2+x3+x4+x5)
yhat<-fitted(model)
residuals<-y-yhat
arguments<-lapply(1:100,as.list)
common.arguments<-
list(data=england.wales,yhat=yhat,residuals=residuals,sample.size=100)
bootstrap.job<-
multiR(session,bootstrap.coeffs,arguments,common.arguments=common.arguments
)
results<-multiR.results(session,bootsrap.job)
coeffs<-results[[1]]
for(i in 2:length(results)) coeffs<-rbind(coeffs,results[[i]]) # this is
where we reassemble the results
apply(coeffs,2,mean)
apply(coeffs,2,sd)
```

MultiR provides a general tool for HTDC, other applications (besides bootstrapping) are MCMC and Geographically Weighted Regression.

Example 2

Longitudinal data make it possible to disentangle the complexities of state dependence, i.e. the dependence of current behaviour on earlier or related outcomes (endogeneity), from the confounding effects of unobserved heterogeneity (omitted variables and cluster effects) and non-stationarity, i.e. changes in the scale and relative importance of the systematic relationships over time. Large sample sizes reduce the problems created by local maxima in the computationally demanding estimation of models for disentangling these complexities in observational data.

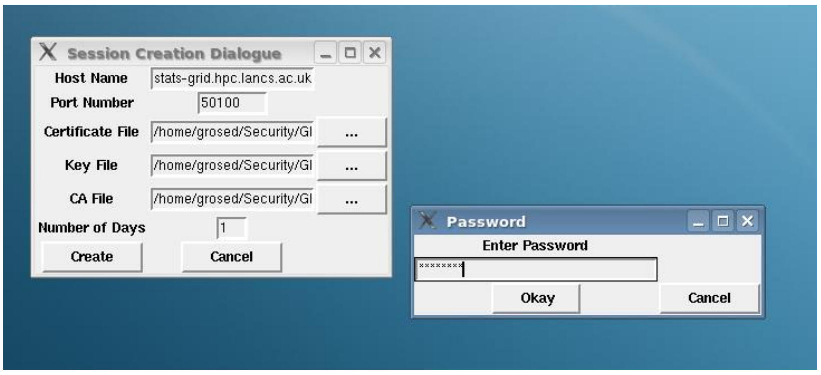
Sabre is a program for the statistical analysis of multi-process random effect response data. These responses can take the form of binary, ordinal, count and linear recurrent events. The response sequences can be of different types. Such multi-process data is common in many research areas, e.g. the analysis of work and life histories. In tests, our multilevel software for event history data (sabreR, sabre in R) outperforms the commercial alternatives. It also out performs the main random effect modelling alternatives for R, i.e. lme4 (http://cran.r-project.org/web/packages/lme4/index.html) and npmlreg (http://cran.r-project.org/web/packages/npmlreg/index.html).

In the 2nd example we use the National Pupil Data (NPD) data. The NPD contains information for all pupils attending publicly-funded secondary schools in England, covering the period from 2001-2 through until 2005-6, i.e. for 5 cohorts. This gives 5 cohorts on attainment at KS2 (aged 11), KS3 (14) and KS4 (16) for each pupil in 3000 schools in 150 LEAs. State Dependence comes from the impact of a pupil's results in KS2 on KS3, and KS3 on KS4. An initial conditions problem arises as we do not observe any attainment prior to KS2. The level-2 heterogeneity effects come from unobserved variation between pupils in latent ability in KS exams. We estimate a 1st order model on attainment using the R library sabreRgrid in order to determine the impact of becoming a specialist school on attainment over the study period.

To use sabreRgrid, the user types `library(sabreRgrid)` in R, and creates a sabre "session" object on their desktop using dialogue boxes, by typing

```
session.1<-sabre.session.dlg()
```

This generates some dialogues which guide the user through the process of selecting a server (host), service (port number) and creation of their proxy credentials required to access this service. The screen snapshot below demonstrates this process (in the sequence as named). In this example, the result is stored in session 1.



Host Name stats-grid.hpc.lancs.ac.uk
Port Number 50100
Certificate File /home/grosed/Security/Gl
Key File /home/grosed/Security/Gl
CA File /home/grosed/Security/Gl
Number of Days 1

Enter Password ********

Submitting a Sabre analysis is now identical to using the R package sabreR on the desktop except that the session object is provided to the sabre function e.g.

```
sabre.model.1<-sabre(y~x,case=id,ordered=TRUE)
```

(which for illustrative purposes is used to estimate a random effects ordered response model with normal quadrature (default) rather than adaptive quadrature) becomes

```
sabre.model.1<-sabre(y~x,case=id,ordered=TRUE,session=session.1)
```

Note that if a session object is not provided, serial sabre is employed for the analysis rather than the service associated with the session object. To recover results or determine the status of an analysis at a later time,

```
print(sabre.model.1,session=session.1)
```

This will show the iterations so far and the coefficients and likelihood if the analysis is complete.

Our R libraries (sabreR, multiR, sabreRGrid) will be obtainable from [1, 2] and http://cran.r-project.org/ [3].

Demonstrations

It is intended to demonstrate several examples of statistical computing using R in conjunction with the multiR and sabreRgrid packages and distributed computing resources provided by the National Grid Service (NGS). The demonstrations are:

1) The basics
A purely pedagogical example which introduces the basic features of the multiR package which demonstrates how to evaluate R functions concurrently on multiple distributed computing resources.

2) Bootstrapping a regression model
This example is run both with and without multiR so as to demonstrate the significant reduction in compute time that can be achieved by using distributed resources. The example highlights some important considerations that need to be made when using distributed resources for statistical computing which include optimising the use of the resources available and ensuring the reliable generation of independent random sequences.

3) Integration of a multivariate distribution
This example demonstrates how functionality provided by existing R packages can be immediately deployed using multiR. In addition, the example is used to demonstrate how, in a distributed environment, traditional calculus based optimisation techniques, (which can be difficult to parallelise), may be replaced by less efficient but (counter-intuitively) quicker "shotgun" methods.

4) Submission/monitoring of a fine grained parallel sabreR job
This example demonstrates how to submit and monitor using sabreRgrid the estimation (using adaptive quadrature) of a bivariate random effects competing risk model.

Obviously, time constraints may mean that all four of the demonstrations may not be possible.

URL Links

[1] Sabre, http://sabre.lancs.ac.uk/

[2] multiR, http://e-science.lancs.ac.uk/multiR

[3] R, http://cran.r-project.org/

References

Bates, G.E., and Neyman, J., (1952), Contributions to the theory of accident proneness, I, An optimistic model of the correlation between light and severe accidents, II, True or false contagion, *Univ Calif, Pub Stat*, 26, 705-720.

Crawley, M,J., (2007), *The R Book,* Wiley

Davies, R.B. and Crouchley, R., (1985), The determinants of party loyalty: a disaggregate analysis of panel data from the 1974 and 1979 General Elections in England, *Political Geography Quarterly*, 4, 307-320.

Heckman J.J., (2001), “Micro data, heterogeneity and the evaluation of public policy: Nobel lecture”, *Journal of Political Economy*, 109, 673—748.

Massy, W.F., Montgomery, D.B., and Morrison, D.G., (1970), *Stochastic models of buying behaviour*, MIT Press, Cambridge, Mass.