Bayesian Robust Inference of Differential Gene Expression
The bridge package

Raphael Gottardo*

May 3, 2016

*Department Statistics, University of Washington
http://www.rglab.org
raph@stat.washington.edu

Contents

1 Overview

The `bridge` package consists of several functions for testing for differential expression among multiple samples. As for now, the package can only be used with two and three samples. The extension to more than three samples should be straight forward. However, it is good to note that as the number of sample increases, the complexity of the model increases exponentially. In pratice, it would be hard to deal with a large number of samples.

We use a robust Bayesian hierarchical model for testing for differential expression with microarrays. The model is a generalization of a model used by **?**) to estimate the true mean intensities from replicates. Outliers are modeled explicitly using a t-distribution. The model is flexible with an exchangeable prior for the variances which allow different variances for the genes but still shrink extreme variances. Our model can be used for testing for differentially expressed genes among multiple samples. Parameter estimation is carried out using Markov Chain Monte Carlo. The robust estimation is achieved by hierarchical Bayesian modeling (**?**).
Realizations are generated from the posterior distribution via Markov chain Monte Carlo (MCMC) algorithms (**?**). Where the full conditionals are of simple form, Gibbs updates are used; where the full conditionals were not of standard form, slice sampling is used. We use the "stepping out" procedure for the slice sampling as introduced in **?**).

Convergence of the MCMC can be assessed using the `coda` library available from CRAN. Our experience with the software is that 50,000 iterations are usually enough to estimate quantities of interest such as posterior means, credible intervals, *etc.* Because the number of genes I can be quite large, the computing time can be long. We recommend running the functions provided in the pack-

age in batch mode. This can be done via `R CMD BATCH`. An example is presented in the next section.

Help files. As with any R package, detailed information on functions, their arguments and value, can be obtained in the help files. For instance, to view the help file for the function `bridge.2samples` in a browser, use `?bridge.2samples`.

2 Installation

2.1 Windows

You can install the package using the menu *Packages* → *Install Packages from local zip ....* Then browse you local directory to find the package. Make sure you have root privilegies if you want to install the package in the default directory.

2.2 Unix/Linux

Under Unix, go into your favorite shell window and type `R CMD INSTALL bridge_xx.tar.gz` (the package has to be in the directory you are executing the command from). By default R will install the package in `/usr/local/R/lib/R ...` and you will need to have root privilegies in order to do so. You can also install the package locally by using the option -l. If you want to install the package in /home/user/Rlib/ use the command `R CMD INSTALL bridge_xx.tar.gz -l /home/user/Rlib`. If you install the package in a directory different from the R default directory you will need to specify the full path when you load the package, i.e.

```
> library(bridge)
```

3 Data Import

3.1 bridge.2samples

You can read your data into R, using the `read.table` command. It is hard to give a general guideline as different data will have different format. In order to use the package you will need to create two data sets, one for each sample (control and treatment). The data should be on the raw scale, i.e. not transformed. The two datasets should be arranged such that rows correspond to genes and columns to replicates. Table **??** give an example of a data set in the right format.

Table 1: Format of the control and treatment datasets used in the rama package. The two samples must be seperated into two datasets.

sample	1	1	1	1	2	2	2	2
replicate	1	2	3	4	1	2	3	4
	Dataset 1				Dataset 2			
gene 1	...	...	...	...	...	...	...	...
gene 2	...	...	...	...	...	...	...	...
⋮	...	...	...	...	...	...	...	...
gene n	...	...	...	...	...	...	...	...

3.2 bridge.3samples

In this case, one would need to have three datasets, one for each sample. Again the data should be on the raw scale, i.e. not transformed!

4 Testing for differential expression between two samples

Testing for differential between two samples is done via the function `bridge.2samples`.
We demonstrate its functionality using gene expression data from an HIV study of **?**). To load the HIV dataset, use `data(hiv)`, and to view a description of the experiments and data, type `?hiv`. We first load the hiv dataset.

```
> data(hiv)
```

This data set consists of 4 experiments using the same RNA preparation on 4 different slides. The expression levels of about 8000 genes were assessed in CD4-T-cell lines at time $t = 24$ hour after infection with HIV virus type 1. The first 4 columns correspond to the first treatment state (hiv infected). The second four represent the control state. The experiment is a balanced dye swap experiment. Finally, the last two columns contain the row and column positions of each gene on the array (slide).

Our goal is to test for differentially expressed genes between the two samples. To demonstrate the function `bridge.2samples` we will only use a subset of 640 genes.

We model $y^* = \log_2(y)$ where y are the raw intensities. Testing of differential expression is done using the function `bridge.2samples`. This function output an object of type `bridge2` containing the sampled values from the posterior distribution. In our case the inference will be based on the posterior probability of differential expression, `post.p`.

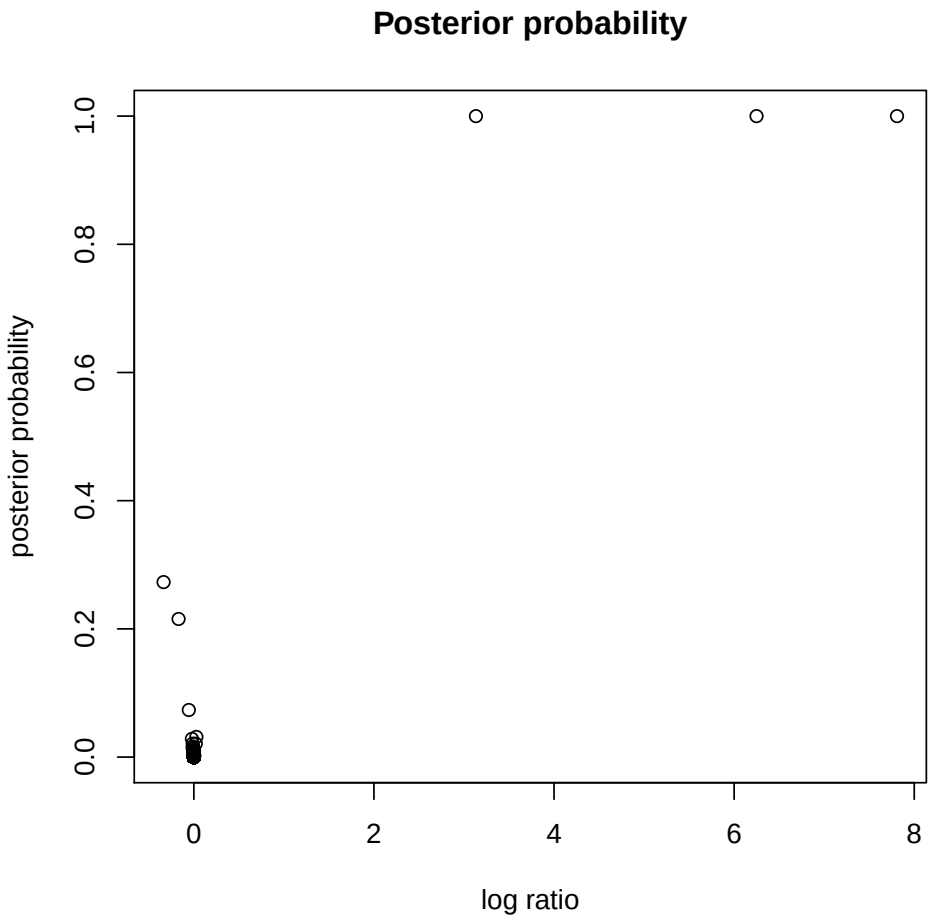
```
> bridge.hiv<-bridge.2samples(hiv[1:640,c(1:4)],hiv[1:640,c(5:8)],B=2000,min.iter=0,batch=1,mcm
```

Once you have fitted the model you may view or plot the sampled parameters from the posterior distribution. We can also obtain point estimates of the paramaters of interest such as the gene effects

```
> gamma1<-mat.mean(bridge.hiv$gamma1)[,1]
> gamma2<-mat.mean(bridge.hiv$gamma2)[,1]
```

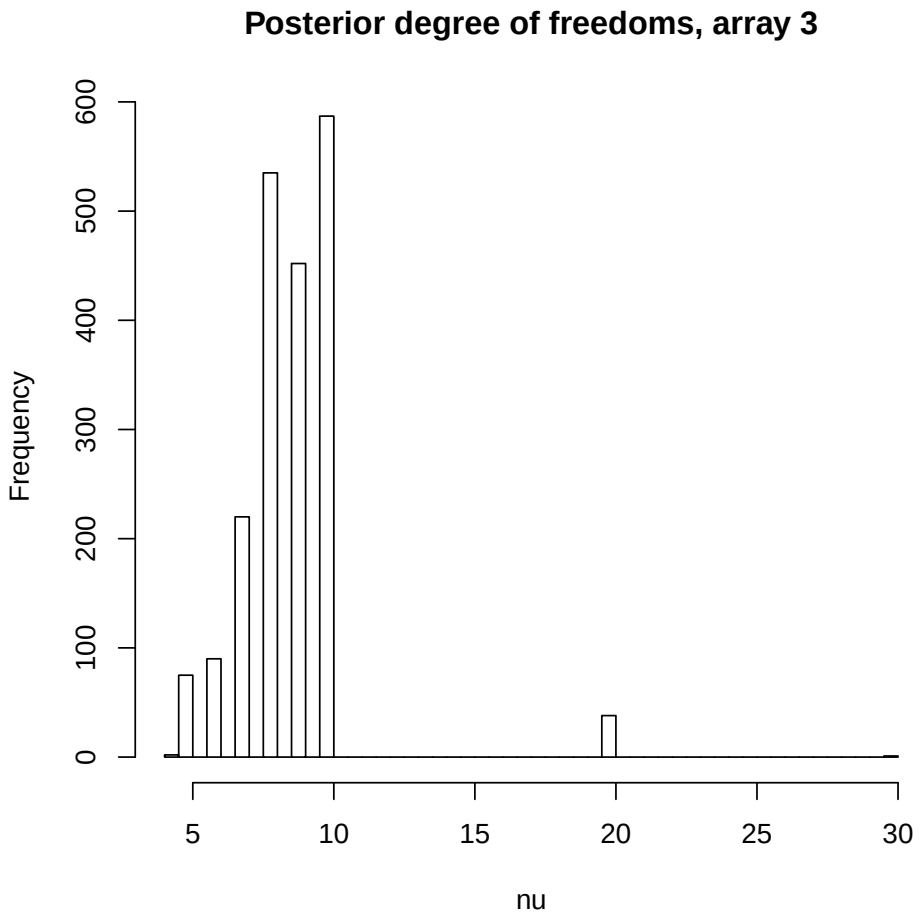
and/or plot the resulting posterior probabilities as a function of the posterior log ratios of $\gamma_1 - \gamma_2$.

```
> plot(gamma1-gamma2,bridge.hiv$post.p,col=1,pch=1,main="Posterior probability",xlab="log ratio
```



Robustness is achieved by using a t-distribution for the errors with a different degree of freedoms for each replicate array. This is formally equivalent to giving each replicate a different weight in the estimation process for each gene. A low number of degrees of freedom for one array will indicate that the corresponding array contains numerous outliers. One could use the function `hist` to look at the posterior mode of the degrees of freedom for each array.

```
> hist(bridge.hiv$nu1[3,],main="Posterior degree of freedoms, array 3",xlab="nu",50)
```



Fitting a t-distribution can be seen as a weighted least square problem with a special hierarchical structure for the variances. Similarly to a weighted least squares function, our function computes weights associated with each replicate. The weights can be useful as a tool for diagnosing the quality of the replicate measurements. If one knows the exact positions of the genes on the array, it is possible to look at the spatial variation of the weights. This can be done via the function `weight.plot` of the `rama` package. We refer the reader to the rama package vignette for further details.

5 Testing for differential expression among three samples

Because (most) microarray technologies allow the direct comparison of at most two samples, the model used in `bridge.2samples` needs to be slightly modified as well to accommodate for changes in the experimental design. In `bridge.2samples`, the log measurements were modeled as a bivariate t-distribution for the log measurements. Depending on the technology used, with three samples, the data will either be ratios (cDNA) or raw measurements (Affymetrix) and therefore a natural model is a univariate t-distribution.

The estimation of the parameters is done similarly to the function `bridge.2samples`. The function output an object of type `bridge3` containing the sampled values from the posterior distribution. In our case the inference will be based on the posterior probability of differential expression.

```
> sample1<-matrix(exp(rnorm(150)),50,3)
> sample2<-matrix(exp(rnorm(200)),50,4)
> sample3<-matrix(exp(rnorm(150)),50,3)
> mcmc.bridge3<-bridge.3samples(sample1,sample2,sample3,B=10,min.iter=0,batch=1,mcmc.obj=NULL,a
```

6 Computing time and batch mode

The computing time can be quite long depending on the number of genes and replicates. It takes about 5 hours to run the MCMC on the full HIV data set using a Linux machine with AMD Athlon at 2GHz. However we feel that this additional computational cost is worth the effort and can lead to good results. The code can be run in batch mode without any user intervention; this optimizes computing resources and allows the user to perform other tasks at the same time. R code can be run in batch mode by typing `R CMD BATCH myfile.R` at the shell prompt. The file `myfile.R` contains the command to execute. An example file could be the following, Then, when the execution is done one can load the resulting output into R using the `load` command. For further details about the `BATCH` command type `?BATCH` in R.

7 Acknowledgment

This research was supported by NIH Grant 8 R01 EB002137-02.