User's Manual for the gimmR

(Gaussian Infinite Mixture Model in R) package

Support web page: http://eh3.uc.edu/gimm

Outline

1. Introduction

This package serves as the interface between the low level procedures for clustering gene expression data using the Bayesian infinite mixtures clustering procedures described in [1-6]. After initial processing, the package makes system calls to two C++ programs for running the Gibbs sampler (*gimm)* and forming the hierarchical clustering based on the output of the Gibbs sampler (*posthoc)*. It is currently still necessary to **also download (from http://eh3.uc.edu/gimm) and install the basic *gimm* package for Linux or Windows.** Eventually, we are planning to release a full package with all functions being compiled into the package. However, compiling low-level analysis procedures separately makes it possible to utilize OpenMP parallelization on multi-processor systems with appropriate compliers. For example, when installing the *gimm* for linux, the installation script checks whether the OpenMP-capable Intel C++ compiler is installed and how many CPU's exist on the system. The user is then offered the option of using the parallelized code.

2. Using the software

If the *gimm* package is not installed in the directory within your *path*, you will need to specify its location after loading the library. For example if the *gimm* software has been installed in the directory gimm within your home directory under a **Linux** system, you can specify the path variables of *gimmR* package as:

```
library(gimmR)
.GimmPath<-"~/gimm/"
.PosthocPath<-"~/gimm/"
```

Similarly, if you have installed the *gimm* package for **Windows** (*WinGimm)* in the folder "C:\WinGimm", you can specify the path in the same way:

```
library(gimmR)
.GimmPath<-"C:/WinGimm/bin/"
.PosthocPath<-"C:/WinGimm/bin/"
```

When using *gimmR* for Windows, it is important that NONE of the folder names in the path of the *WinGimm* have blanks. Therefore, if you are planning to use *gimmR*, it is important NOT to install *WinGimm* in the default directory (C:\Program Files\WinGimm).

After specifying the two path variables, the following code will cluster the Galactose dataset distributed within R and display the resulting heatmap. This is the dataset we used in several publications to compare the performances of various clustering procedures [2, 7].

```
data(GalData)
galGimm <- runGimmNPosthoc(GalData, M=20, T=820, nIter=100,
           burnIn=50)
heatmap(data.matrix(galGimm$clustData[, -(1:2)]),
   Rowv=as.dendrogram(galGimm$hGClustData), Colv=NA,
   labCol=colnames(galGimm$clustData)[-(1:2)],
   labRow=galGimm$clustData[, 2], scale="none")
```

Please note that the default stack size may be to low when using *gimmR* (or most other hierarchical clustering procedures). To execute the example above, you may need to start your R session with a command like the following:

```
R --max-ppsize=50000
```

If the number of genes in the dataset to be analyzed is extremely large, you may need to increase this number depending on the depth of the resulting hierarchical gene clustering.

The Gibbs sampling can be time-consuming. In order to monitor progress, you can set the variable verbose to TRUE, which will print the current iteration counter and the current number of clusters:

```
galGimm <- runGimmNPosthoc(GalData, M=20, T=820, nIter=100,
           burnIn=50, verbose=TRUE)
```

In addition to creating a list of data frames that can be used to generate heatmaps within R, as a side-effect, `runGimmNPosthoc` produces **Results.cdt** and **Results.gtr** files in the current working directory. The hierarchical clustering defined in these two files can be viewed and analyzed using **treeview** software [8]**.** The java version of **treeview** that runs under Linux can be downloaded from http://sourceforge.net/project/showfiles.php?group_id=84593.

The Windows version can be downloaded from Michael Eisen's web page http://rana.lbl.gov/EisenSoftware.htm. An extended version facilitating the functional annotation of clustering results is available at http://eh3.uc.edu/clean/ and http://eh3.uc.edu/ftreeview/. **The run-time of the Gibbs sampler depends on the size of the data and the number of iterations specified, but also on the number of mixtures underlying the data. Therefore, the Gibbs sampler can be time-consuming!** For this reason, it is recommended to run the sampler in the verbose mode in order to monitor the progress of the sampling procedure.

3. Formatting for simple model

The required format for the input data set can be inferred from the structure of the *GalData* data frame. In the simple model each row in the data frame represents the gene expression profile for a single gene. The first two columns are assumed to contain gene annotations and the remaining columns contain expression levels of genes on different microarrays/experimental conditions. In the *GalData* dataset each gene actually has 4 rows of data obtained from 4 independent microarray hybridizations, but in this simple analysis they are treated as different expression profiles. One simple assessment of the results can be made in terms of the proportion of experimental replicates for the same gene that cluster together. For a thorough description of the dataset see [7]. For the proper treatment of this dataset and experimental replicates it contains using the replicated data IMM model see the next section.

4. Analyzing data with replicates

The following code will cluster *GalData* using the replicated data IMM model as described in [2]. The only difference from the code in Section 2 is that nReplicates parameter is set to 4 and the T parameter specifying the number of profiles is set to 205 in the call to `runGimmNPosthoc function.`

```
data(GalData)
galGimm<-runGimmNPosthoc(GalData, M=20, T=205, nReplicates=4,
          nIter=100, burnIn=50, verbose=TRUE)
heatmap(data.matrix(galGimm$clustData[, -(1:2)]),
   Rowv=as.dendrogram(galGimm$hGClustData), Colv=NA,
   labCol=colnames(galGimm$clustData)[-(1:2)],
   labRow=galGimm$clustData[, 2], scale="none")
```

The generic structure of a data frame to be used with the replicated data model for T genes across M conditions with two replicates per condition looks as follows.

GeneID	GeneName	Exp_1	Exp_2	Exp3	Exp_4	…	Exp_M
$Gene_1$	$GeneName_1$	G_1-C_1-R_1	G_1-C_2-R_1	G_1-C_3-R_1	G_1-C_4-R_1	...	G_1-C_M-R_1
$Gene_1$	$GeneName_1$	G_1-C_1-R_2	G_1-C_2-R_2	G_1-C_3-R_2	G_1-C_4-R_2	...	G_1-C_M-R_2
$Gene_2$	$GeneName_2$	G_2-C_1-R_1	G_2-C_2-R_1	G_2-C_3-R_1	G_2-C_4-R_1	...	G_2-C_M-R_1
$Gene_2$	$GeneName_2$	G_2-C_1-R_2	G_2-C_2-R_2	G_2-C_3-R_2	G_2-C_4-R_2	...	G_2-C_M-R_2
....	....	....	....	....	....	....	....
$Gene_T$	$GeneName_T$	G_T-C_1-R_1	G_T-C_2-R_1	G_T-C_3-R_1	G_T-C_4-R_1	...	G_T-C_M-R_1
$Gene_T$	$GeneName_T$	G_T-C_1-R_2	G_T-C_2-R_2	G_T-C_3-R_2	G_T-C_4-R_2	...	G_T-C_M-R_2

where G_i represents the i^{th} gene, C_j represents the j^{th} experimental condition and R_k represents the k^{th} replicate.

5. Using Context-Specific Infinite Mixture Model (CSIMM)

In the context-specific model, the experimental conditions are further organized into contexts (http://eh3.uc.edu/gimm/csimm). The following code will re-create the analysis of the sporulation and cell cycle data described in the paper [4]. **It should be noted that this code takes about three hours to run on the OpenMP-enabled system with dual 3.6GHz Xeons. On a single CPU Windows machine it can take 24 hours to complete. To run just a few test-iterations of the algorithm reduce the nIter and burnIn numbers.**

```
data(PrimigCho)
pcGimm<-runGimmNPosthoc (PrimigCho, M=31, T=5865, nIter=10,
         burnIn=5, contextSpecific="y", nContexts=4,
         contextLengths=c(8,7,9,7), verbose=TRUE, intFiles=TRUE)
heatmap(data.matrix(pcGimm$clustData[,-(1:2)]),
         Rowv=as.dendrogram(pcGimm$hGClustData), Colv=NA,
         labCol=colnames(pcGimm$clustData)[-(1:2)],
         labRow=pcGimm$clustData[,2], scale="none")
```

Successfully running this code may require starting R with

```
R --max-ppsize=50000
```

and set the *expressions* option with

```
options(expressions=100000)
```

As it can be seen from the code, in order to run the context-specific clustering, the *contextSpecific* parameter is set to “y”, *nContexts* is set to 4, and *contextLengths* is set to the vector c(8,7,9,7) defining number experiments in each of the 4 contexts. Note that numbers in *contextLengths* need to add up to the total number of experiments *M*. Furthermore, the context-specific model at this point does not support replicated observations.

6. Using Differential Co-expression Infinite Mixtures (DCIM)

Like in the context-specific model, the experimental conditions are further organized into contexts but DCIM is designed to discern these contexts from the data rather than require their *a priori* specification (http://eh3.uc.edu/gimm/dcim/). The following code will run the DCIM analysis for a subset of the Schmidt *et al.* dataset [9]. The subset is comprised of the top 500 DCS gene signature [6]. To invoke the DCIS algorithm set the `estimate_contexts` parameter to “y”. By default, each sample is assigned to its own “basic” context, and basic contexts are then grouped into “meta”-contexts. It is possible to specify basic contexts by using the `nContexts` and `contextLengths` parameters as in the CSIMM setting. **It should be noted that this code may take an extended period of time to complete. To run just a few test-iterations of the algorithm reduce the nIter and burnIn numbers.**

```
data(DCE500)
dceGimm <- runGimmNPosthoc(DCE500, M=200, T=500, nIter=100,
         burnIn=50, estimate_contexts="y", verbose=TRUE,
         intFiles=TRUE)
drawHeatmap(dceGimm, scale="none")
```

Given two contexts (usually estimated from the data), the following code will determine the gene-specific differential co-expression (DCE) score for these two contexts and plot their distribution.

```
data(GalData)
gimmOut <- runGimmNPosthoc(GalData, dataFile="galData", M=20,
         T=820, estimate_contexts="y", intFiles=TRUE,
         verbose=TRUE)
res <- computeDCEscore(which(cutree(gimmOut$hSClustData,
         k=2)==1), which(cutree(gimmOut$hSClustData, k=2)==2),
         paramsList=list(dataFile="galData", M=20, T=820,
         burnIn=5000))
res <- res[gimmOut$hGClustData$order]
plot(1:length(res), res, type="n")
polygon(c(1:length(res), length(res), 1), c(res, 0, 0), col=4))
```

Appendix: Using custom-compiled *gimm* executables

If the executables (*gimm* and *posthoc*) are not working on your Linux/Unix system, or you want to take an advantage of OpenMP parallelization, please download and install **(from http://eh3.uc.edu/gimm)** the *gimm* package**.** During installation of the *gimm* for linux, the installation script checks whether the OpenMP-capable Intel C++ compiler is installed and how many CPU's exist on the system. User is then offered the option of making use of parallelized code. If the *gimm* software has been installed in the directory gimm within your home directory under a **Linux** system, you will need to set the path variables of *gimmR* package as:

```
library(gimmR)
.GimmPath<-"~/gimm/"
.PosthocPath<-"~/gimm/"
```

This will have to be done every time you use *gimmR*. Alternatively, you can replace old binaries (*gimm* and *posthoc*) in R/library/gimmR/doc (assuming that this is where gimmR is installed and you have appropriate privileges to do that) with newly created executables in the ~/gimm directory.

Reference List

1. Medvedovic M, Sivaganesan S: **Bayesian infinite mixture model based clustering of gene expression profiles**. *Bioinformatics* 2002, **18:**1194-1206.

2. Medvedovic M, Yeung KY, Bumgarner RE: **Bayesian mixture model based clustering of replicated microarray data**. *Bioinformatics* 2004, **20:**1222-1232.

3. Medvedovic M, Guo J: **Bayesian Model-Averaging in Unsupervised Learing From Microarray Data**. *BIOKDD 2004* 2004.

4. Liu X, Sivaganesan S, Yeung KY, Guo J, Bumgarner RE, Medvedovic M: **Context-specific infinite mixtures for clustering gene expression profiles across diverse microarray dataset**. *Bioinformatics* 2006, **22:**1737-1744.

5. Liu X, Jessen WJ, Sivaganesan S, Aronow BJ, Medvedovic M: **Bayesian hierarchical model for transcriptional module discovery by jointly modeling gene expression and ChIP-chip data**. *BMC Bioinformatics* 2007, **8:**283.

6. Freudenberg JM, Sivaganesan S, Wagner M, Medvedovic M: **A semi-parametric Bayesian model for unsupervised differential co-expression analysis**. *BMC Bioinformatics* 2010.

7. Yeung KY, Medvedovic M, Bumgarner RE: **Clustering Gene Expression Data with Repeated Measurements**. *Genome Biology* 2003, **4:**R34.

8. Eisen MB, Spellman PT, Brown PO, Botstein D: **Cluster analysis and display of genome-wide expression patterns**. *Proc Natl Acad Sci U S A* 1998, **95:**14863-14868.

9. Schmidt M, Bohm D, von TC, Steiner E, Puhl A, Pilch H, Lehr HA, Hengstler JG, Kolbl H, Gehrmann M: **The humoral immune system has a key prognostic impact in node-negative breast cancer**. *Cancer Res* 2008, **68:**5405-5413.