Bioconductor's hopach package

Katherine S. Pollard[1] and Mark J. van der Laan[2]

October 17, 2016

1. Center for Biomolecular Science and Engineering, University of California, Santa Cruz,
`http://lowelab.ucsc.edu/katie/`
2. Department of Statistics and Division of Biostatistics, University of California, Berkeley,
`http://www.stat.berkeley.edu/~laan/`

Contents

1 Overview

This document provides a tutorial for the *hopach* package.

2 Getting started

Installing the package. To install the *hopach* package, first download the appropriate file for your platform from the Bioconductor website `http://www.bioconductor.org/`. For Windows, start R and select the `Packages` menu, then `Install package from local zip file...`. Find and highlight the location of the zip file and click on `open`. For Linux/Unix, use the usual command `R CMD INSTALL` or set the option `CRAN` to your nearest mirror site and use the command

`install.packages` from within an R session.

Loading the package. To load the *hopach* package in your R session, type `library(hopach)`.

Help files. Detailed information on *hopach* package functions can be obtained in the help files. For example, to view the help file for the function `distancematrix` in a browser, use `help.start` followed by `? distancematrix`.

Case study. We demonstrate the functionality of this R package using gene expression data from the AML/ALL microarray gene expression experiments published by Golub *et al.* Golub et al. (1999). These data are included as part of the *hopach* package. To load the `golub` dataset, use `data(golub)`. To view a description of the experiments and data, type `? golub`.

Sweave. This document was generated using the `Sweave` function from the R *tools* package. The source (.Rnw) file is in the `/inst/doc` directory of the *hopach* package.

3   Introduction

An important goal with large-scale gene expression studies is to find biologically important subsets and clusters of genes. We have developed a hybrid clustering method, Hierarchical Ordered Partitioning And Collapsing Hybrid (HOPACH), which builds a hierarchical tree of clusters van der Laan and Pollard (2001, 2003). The methodology combines the strengths of both partitioning and agglomerative clustering methods. At each node, a cluster is split into two or more smaller clusters with an enforced ordering of the clusters. Collapsing steps uniting the two closest clusters into one cluster are used to correct for errors made in the partitioning steps. The `hopach` function uses the median split silhouette (MSS) criteria Pollard and van der Laan (2002a,b) to automatically choose (i) the number of children at each node, (ii) which clusters to collapse, and (iii) the main clusters ("pruning" the tree to produce a partition of homogeneous clusters). An ordered list of genes (or arrays) is obtained by running down the tree completely to the final level. In this tutorial, we illustrate how to use many of functions in the *hopach* package.

4   HOPACH Clustering of Genes

To load the necessary packages and the Golub data set:

```
> library(hopach)
> data(golub)
```

Next, select a subset of interesting genes. Such a subset can be chosen in many ways, for example with the functions in the *genefilter* and *multtest* packages. For this analysis, we will simply take the 200 genes with highest variance across the arrays. In practice, one will typically use a larger data set. The small size was chosen here for demonstration purposes.

```
> vars<-apply(golub,1,var)
> subset<-vars>quantile(vars,(nrow(golub)-200)/nrow(golub))
```

```
> golub.subset<-golub[subset,]
> dim(golub.subset)
```

```
[1] 200   38
```

```
> gnames.subset<-golub.gnames[subset,]
```

It is useful to compute the distance matrix before running `hopach`, because the distance matrix may be needed later in the analysis. Having a copy in hand saves computation time, particularly with larger data sets. The *cosangle* distance metric is often a good choice for clustering genes.

```
> gene.dist<-distancematrix(golub.subset,"cosangle")
> dim(gene.dist)
```

```
[1] 200 200
```

Now, run hopach. The algorithm will take some time to run.

```
> gene.hobj<-hopach(golub.subset,dmat=gene.dist)
> gene.hobj$clust$k
```

```
[1] 63
```

```
> table(gene.hobj$clust$labels)
```

```
1110 1120 1210 1220 1310 1320 1330 1340 1350 1360 1410 1420 1430 1440 1500 2100
   1    4    1    5    1    1    1    1    1    3    1    1    3    1    3    1
2210 2221 2222 2223 2224 2300 2510 2520 2530 2540 2550 2560 2570 2580 2611 2620
   1    3    2   12    6    4    1    2    1    1    3    1    1    1    4    1
2630 2650 2700 3100 3200 3300 3400 3511 3600 3710 3720 3730 3740 3810 3820 3830
   1    1   10    1    3    1    2    4    4    1    1    6    1    1    2    2
3840 3900 4100 4200 5100 5200 5310 5320 5330 5340 5500 5600 5711 5720 5750
   3    5   37   19    1    1    1    1    8    1    1    2    4    1    1
```

```
> gene.hobj$clust$sizes
```

```
 [1]  1  4  1  5  1  1  1  1  1  3  1  1  3  1  3  1  3  2 12  6  4  1  2  1
[26]  1  3  1  1  1  4  1  1  1 10  1  3  1  2  4  4  1  1  6  1  1  2  2  3  5
[51] 37 19  1  1  1  1  8  1  1  2  4  1  1
```

The rows and columns of the distance matrix can be ordered according to the order of the genes in the final level of the hierarchical tree. Genes close to each other in the tree are similarly expressed, so the ordered distance matrix shows the clustering structure (See Figure 1). Clusters of similarly expressed genes appear as blocks on the diagonal of the matrix. With the default colors, red represents small distance and white large distance.

```
> dplot(gene.dist,gene.hobj,ord="final",main="Golub AML/ALL Data (1999): Gene Distance Matrix",
```

5 Bootstrap Resampling

In order to better understand the variability of the `hopach` clusters, we use the non-parametric bootstrap. The proportion of bootstrap resampled data sets that each gene falls into each of the clusters (fixed from the `hopach` clustering result) is an estimate of the membership of that gene in each cluster. This is a form of "fuzzy clustering".

```
> bobj<-boothopach(golub.subset,gene.hobj,B=100)
```

The argument B controls the number of bootstrap resmapled data sets used. The default value is `B=1000`, which represents a balance between precision and speed. For this example, we use only `B=100` so that it will not run too long. The `boothopach` function still takes some time to run. The bootstrap is a powerful, but computationally intensive, method. The `bootplot` function makes a barplot of the bootstrap reappearance proportions (See Figure 2).

```
> bootplot(bobj,gene.hobj,ord="bootp",main="Golub AML/ALL Data (1999)",showclusters=FALSE)
```

6 HOPACH Clustering of Arrays

The HOPACH algorithm can also be applied to cluster arrays, based on their expresion profiles across genes. In this AML/ALL data set, we actually know that the arrays come from two classes of patients.

```
> table(golub.cl)
```

We will ignore the labels, and perform unsupervised clustering of the arrays using `hopach`. This analysis method differs from classification, because the class labels are not used and the subset of genes has not been chosen to best discriminate between the two classes of arrays. Hence, we do not necessarily expect the clustering to reproduce the class labels. This exploratory approach allows us to see what patterns there are in the data without using any information besides the subset of gene expression measurements. Euclidean distance is often a good choice for clustering arrays.

```
> array.hobj<-hopach(t(golub.subset),d="euclid")
> array.hobj$clust$k
```

```
[1] 17
```

7 Output files

7.1 Gene clustering and bootstrap results table

The `makeoutput` function is used to write a tab delimited text file that can be opened in a spreadsheet application or text editor. The file will contain the `hopach` clustering results, plus possibly the corresponding bootstrap results, if these are provided. The argument *gene.names* can be used to insert additional gene annotation.

```
> makeoutput(golub.subset,gene.hobj,bobj,file="Golub.out",gene.names=gnames.subset[,3])
```

4

7.2 Bootstrap fuzzy clustering in MapleTree

The MapleTree software `http://mapletree.sourceforge.net/` is an open source, cross-platform, visualization tool to graphically browse results of cluster analyses. The `boot2fuzzy` function takes the gene expression data, plus corresponding `hopach` clustering output and bootstrap resampling output, and writes the (.cdt, .fct, and .mb) files needed to view these "fuzzy clustering" results in MapleTree.

```
> boot2fuzzy(golub.subset,bobj,gene.hobj,array.hobj,file="GolubFuzzy",gene.names=gnames.subset
```

The three generated files can be opened in MapleTree by going to the `Load` menu and then `Fuzzy Clustering Data`. The heat map contains only the medoids genes (cluster profiles). Double clicking on a medoid opens a zoom window for that cluster, with a heat map of all genes ordered by their bootstrap estimated memberships in that cluster (highest memebership first).

7.3 HOPACH hierarchical clustering in MapleTree

The MapleTree software can also be used to view HOPACH hierarchical clustering results. The `hopach2tree` function takes the gene expression data, plus corresponding `hopach` clustering output for genes and/or arrays, and writes the (.cdt, .gtr, and .atr) files needed to view these hierarchical clustering results in MapleTree. These files can also be opened in other viewers such as TreeView `http://rana.lbl.gov/EisenSoftware.htm`, jtreeview `http://sourceforge.net/projects/jtreeview/`, and GeneXPress `http://genexpress.stanford.edu/`.

```
> hopach2tree(golub.subset,file="GolubTree",hopach.genes=gene.hobj,hopach.arrays=NULL,dist.gene
```

The `hopach2tree` writes up to three text files to the current working directory (or path given by the *file* argument). A .cdt file is always produced. When *hopach.genes*! =NULL, a .gtr is produced, and gene clustering results can be viewed, including ordering the genes in the heat map according to the final level of the `hopach` tree and drawing the dendogram for hierarchical gene clustering. Similarly, when *hopach.arrays*! =NULL, an .atr file is produced and array clustering results can be viewed. These files can be opened in MapleTree by going to the `Load` menu and then `HOPACH Clustering Data`.

References

T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:321–531, October 15 1999. URL `http://www-genome.wi.mit.edu/MPR/`.

K. Pollard and M. van der Laan. New methods for identifying significant clusters in gene expression data. In *Proceedings of the American Statistical Association*, volume Biometrics Section [CD-ROM]. American Statistical Association, 2002a. URL `http://www.stat.berkeley.edu/~laan/Research/Research_subpages/Papers/jsmpaper.pdf`.

Golub AML/ALL Data (1999): Gene Distance Matrix
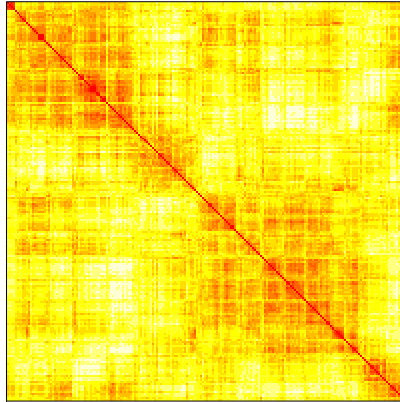**Ordered Distance Matrix**

Figure 1: The `dplot` function orders the rows and columns of the distance matrix according to the final level of the `hopach` hierarchical tree. Red represents small distance and white large distance.

K. Pollard and M. van der Laan. A method to identify significant clusters in gene expression data. In *SCI2002 Proceedings*, volume II, pages 318–325. International Institute of Informatics and Systemics, 2002b. URL `http://www.bepress.com/ucbbiostat/paper107/`.

M. van der Laan and K. Pollard. Hybrid clustering of gene expression data with visualization and the bootstrap. Technical Report 93, Group in Biostatistics, University of California, may 2001. URL `http://www.stat.berkeley.edu/~laan/Research/Research_subpages/Papers/hopach.pdf`. To appear in JSPI.

M. van der Laan and K. Pollard. A new algorithm for hybrid hierarchical clustering with visualization and the bootstrap. *Journal of Statistical Planning and Inference*, 117:275–303, 2003.

Golub AML/ALL Data (1999)
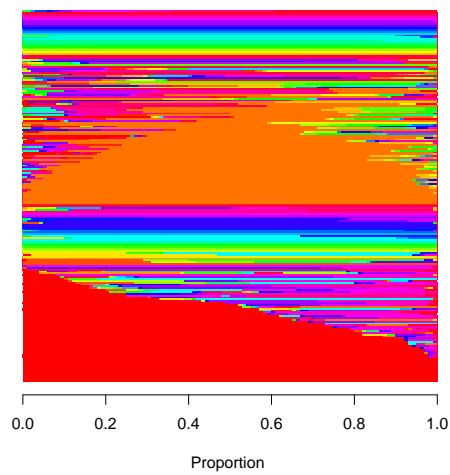**Barplot of Bootstrap Reappearance Proportions**

Figure 2: The `bootplot` function makes a barplot of the bootstrap reappearance proportions for each gene and each cluster. These proportions can be viewed as fuzzy cluster memberships. Every cluster is represented by a different color. The genes are ordered by `hopach` cluster, and then by bootstrap estimated membership within cluster on the vertical axis so that each gene is represented by a horizontal bar.