Pigengene: Computing and using eigengenes

Habil Zare

Modified: 26 April, 2016. Compiled: October 17, 2016

Contents

1 Introduction

Gene expression profiling technologies such as microarray or RNA Seq provide valuable datasets, however, inferring biological information from these data remains cumbersome. *Pigengene* address two challenges:

1. **Curse of dimensionality:** The number of features in an expression profile is usually very high. For instance, there are about 20,000 genes in human. In contrast, the number of samples (patients) is often very limited in practice, and may not exceed a few hundreds. Yeung et al. have shown that standard data reduction methods such as principal component analysis (PCA) are not appropriate to directly apply on gene expression data [1]. Instead, *Pigengene* addresses this challenge by applying PCA on gene modules.
2. **Normalization:** Data produced using different technologies, or in different labs, are not easily comparable. *Pigengene* identifies *eigengenes*, informative biological signatures that are robust with respect to the profiling platform. For instance, it can identify the signatures (compute the eigengenes) on microarray data, and infer them on biologically-related RNA Seq data. The resulting signatures are directly comparable even if the set of samples (patients) are independent and disjoint in the two analyzed datasets.

2 How to run Pigengene ?

2.1 Installation

Pigengene is an *R* package that can be downloaded and installed from *Bioconductor* by the followig commands in R:

```
source("http://bioconductor.org/biocLite.R")
biocLite("Pigengene")
```

Alternatively, if the source code is already available, the package can be installed by the following command in Linux:

```
R CMD INSTALL Pigengene_x.y.z.tar.gz
```

where x.y.z. determines the version. The second approach requires all the dependencies be installed manually, therefore, the first approach is preferred.

2.2 A quick overview

Pigengene identifies gene modules (clusters), computes an eigengene for each module, and uses these biological signatures as features for classification. The main function is `one.step.pigengene` which requires a gene expression profile and the corresponding conditions (types). Individual functions are also provided to facilitate running the pipeline in a customized way. The inferred biological signatures (eigengenes) are useful for supervised or unsupervised analyses.

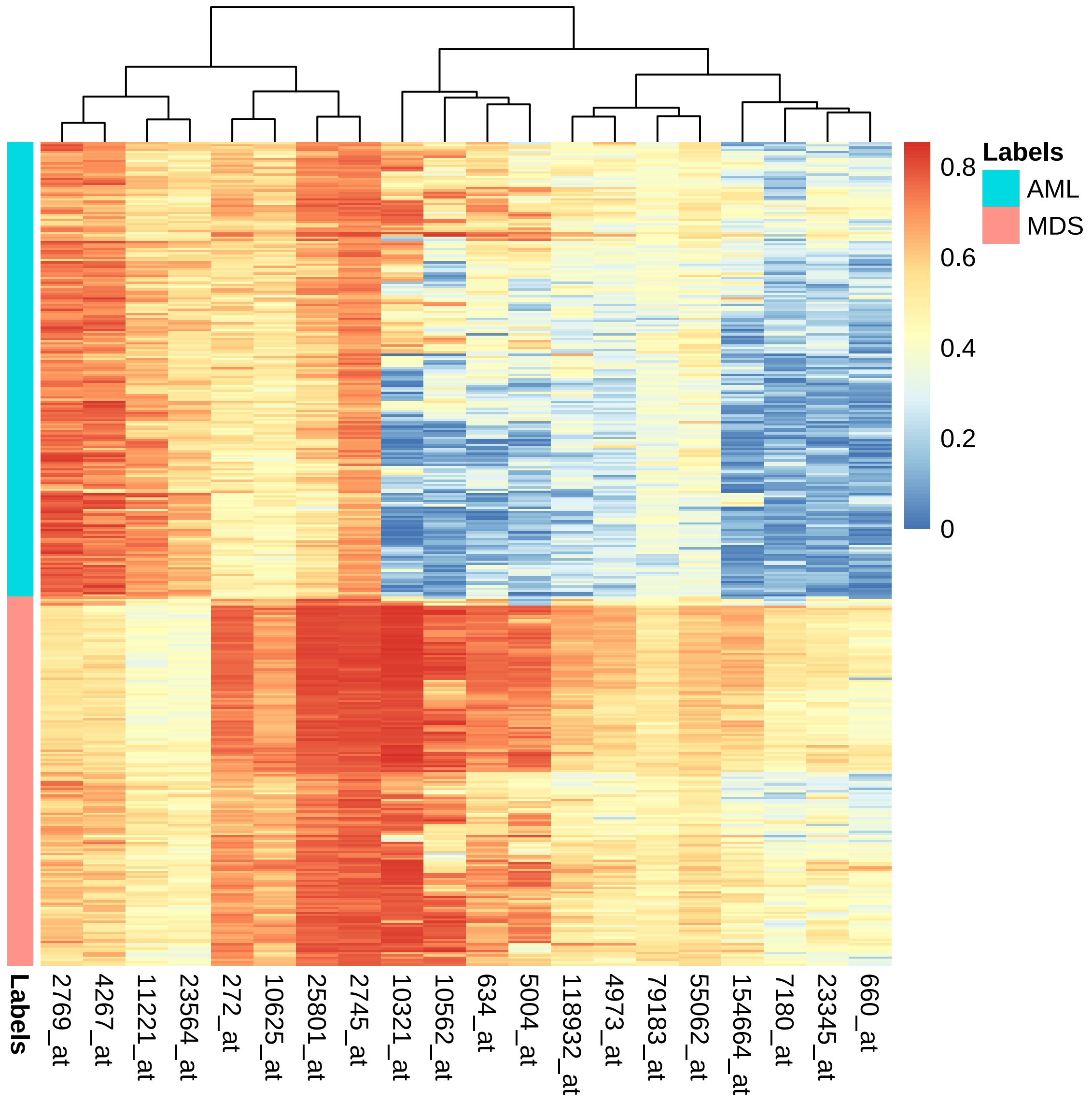
2.3 What is an eigengene?

In most functions of this package, eigenegenes are computed or used as robust biological signatures. Briefly, each eigengene is a weighted average of the expression of all genes in a given set of genes (also known as a gene module or a cluster of genes). The weights are adjusted in a way that the explained variance is maximized. This guarantees that the loss in the biological information in minimized.

2.4 A toy example

For a quick start, the application of *Pigengene* pipeline on some leukemia dataset is demonstrated below [2]. The first step is to load the package and data in *R*:

```
library(Pigengene)

## Loading required package:  graph

## Loading required package:  BiocGenerics
## Loading required package:  parallel

##
## Attaching package:  'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ, clusterExport,
##     clusterMap, parApply, parCapply, parLapply, parLapplyLB, parRapply,
##     parSapply, parSapplyLB

## The following objects are masked from 'package:stats':
##
##     IQR, mad, xtabs

## The following objects are masked from 'package:base':
##
##     Filter, Find, Map, Position, Reduce, anyDuplicated, append, as.data.frame,
##     cbind, colnames, do.call, duplicated, eval, evalq, get, grep, grepl,
##     intersect, is.unsorted, lapply, lengths, mapply, match, mget, order, paste,
##     pmax, pmax.int, pmin, pmin.int, rank, rbind, rownames, sapply, setdiff,
##     sort, table, tapply, union, unique, unsplit, which, which.max, which.min

##

data(aml)
data(mds)
d1 <- rbind(aml,mds)
Labels <- c(rep("AML",nrow(aml)),rep("MDS",nrow(mds)))
names(Labels) <- rownames(d1)
Disease <- as.data.frame(Labels)
p1 <- pheatmap.type(d1[,1:20],annRow=Disease,show_rownames=FALSE)
```



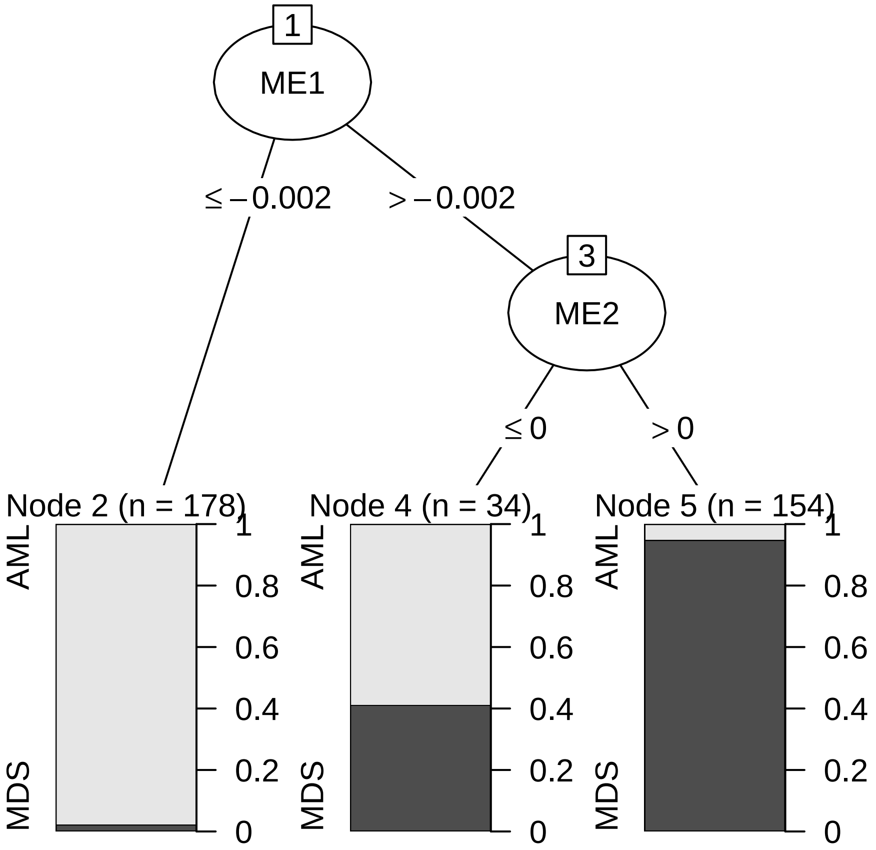
Please note that the provided data in the package is sub-sampled for a quicker demonstration. For real applications, the expression of thousands of genes should be provided in order to the co-expression network analysis to be appropriate. It is common to first perform differential expression analysis, sort all the genes based on p-values, and use the top-third as the input [3]. Analyzing such input with *Pigengene* can take a few hours and may require 5-10 GB of memory. The following command runs Pigengene pipeline on the toy data:

```
p1 <- one.step.pigengene(Data=d1,saveDir='pigengene', bnNum=0, verbose=1,
      seed=1, Labels=Labels, toCompact=FALSE, doHeat=FALSE)

## Pigengene started analizing 366 samples using 1000 genes...

## Warning:  executing %dopar% sequentially:  no parallel backend registered

## adjacency: replaceMissing: 0

## Pigengenes...
## Warning in cbind(Size, pvalues$pvals): number of rows of result is not a multiple
of vector length (arg 1)

## Pigengene plots in:

## /tmp/RtmpQzTvZZ/Rbuild353e74e5d37b/Pigengene/vignettes/pigengene/plots

## Making decision trees...

## minPerLeaf: 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37

## costs:

##     AML MDS
## AML   0   1
## MDS   1   0

## toCompact: FALSE
```

Results and figures are saved in pigengene folder under the current directory. For more advanced applications, the user is encouraged to analyze the data step-by-step and customize the individual functions such as compute.pigenegene and make.decision.tree.

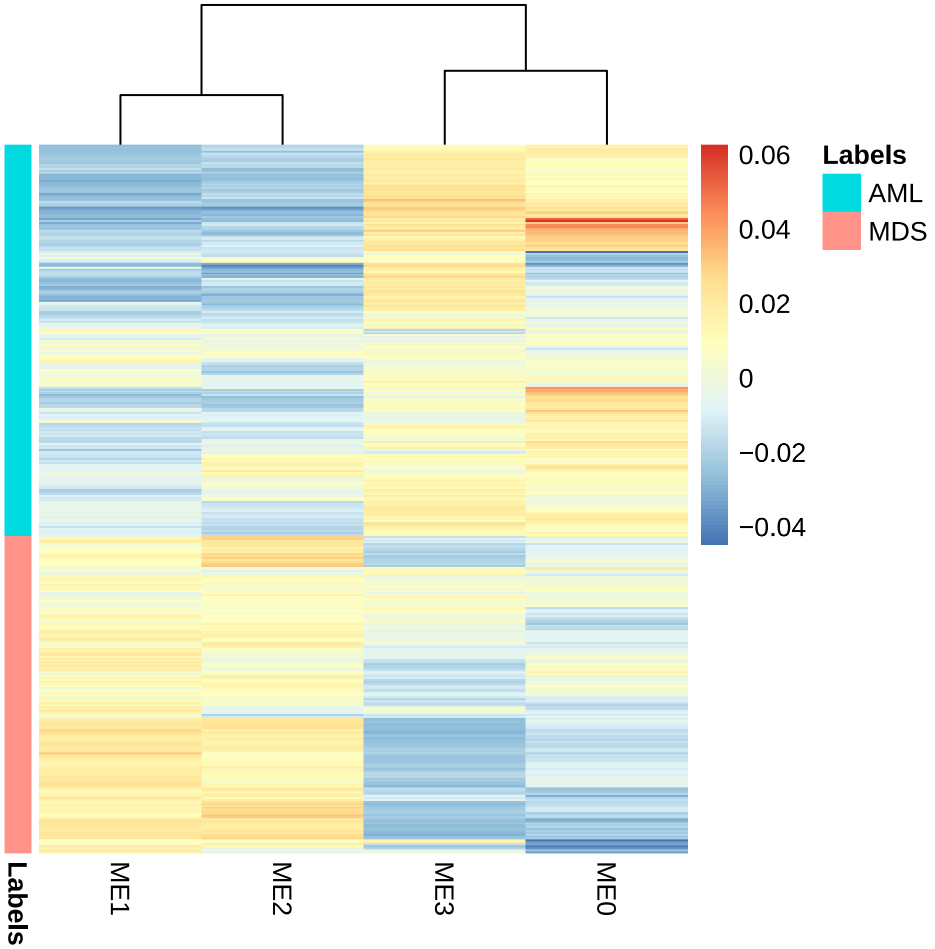
In addition to the provided decision trees, the user can also take alternative approaches to perform classification, clustering, survival analysis, etc. *using eigengenes as robust biological signatures (informative features)*. Eigengenes and other useful objects can be retrieved from the output. For instance, c5treeRes is a list containing the results of fitting decision trees to the eigengenes. As shown above, a couple of trees were fitted, one per a value for minPerLeaf. The following command plots the tree corresponding to 34, i.e., it was fitted requiring the minimum number of samples per every leaf to be at least 34.

```
plot(p1$c5treeRes$c5Trees[["34"]])
```



The tree corresponding to other values are saved in `pigengene` folder. Of note, is the `pigenegene` object that contains the matrix of inferred eigenegenes. Each row corresponds to a sample, and each column represents an eigengene.

```
dim(p1$pigengene$eigengenes)

## [1] 366   4

p1 <- pheatmap.type(p1$pigengene$eigengenes,annRow=Disease,show_rownames=FALSE)
```



2.5 Citation

The methodology and an interesting application of *Pigengene* on studying hematological malignancies is presented in the following reference [4].

```
citation("Pigengene")
```

To cite package 'Pigengene' in publications use:

Amir Foroushani et al.(2016) Large-scale gene network analysis reveals the significance of extracellular matrix pathway and homeobox genes in acute myeloid leukemia, Foroushani et al., In preparation. URL: http://oncinfo.org/.

A BibTeX entry for LaTeX users is

@Article, author = Amir Foroushani and et al., title = Large-scale gene network analysis reveals the significance of extracellular matrix pathway and homeobox genes in acute myeloid leukemia, journal = In preparation, year = 2016,

3 Session Information

The output of `sessionInfo` on the system that compiled this document is as follows:

```
toLatex(sessionInfo())
```

- R version 3.3.1 (2016-06-21), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: BiocGenerics 0.20.0, Pigengene 1.0.0, graph 1.52.0
- Loaded via a namespace (and not attached): AnnotationDbi 1.36.0, Biobase 2.34.0, BiocStyle 2.2.0, C50 0.1.0-24, DBI 0.5-1, Formula 1.2-1, GO.db 3.4.0, Hmisc 3.17-4, IRanges 2.8.0, MASS 7.3-45, Matrix 1.2-7.1, RColorBrewer 1.1-2, RSQLite 1.0.0, Rcpp 0.12.7, Rgraphviz 2.18.0, S4Vectors 0.12.0, WGCNA 1.51, acepack 1.3-3.3, bnlearn 4.0, chron 2.3-47, cluster 2.0.5, codetools 0.2-15, colorspace 1.2-7, compiler 3.3.1, data.table 1.9.6, doParallel 1.0.10, dynamicTreeCut 1.63-1, evaluate 0.10, fastcluster 1.1.21, foreach 1.4.3, foreign 0.8-67, formatR 1.4, ggplot2 2.1.0, grid 3.3.1, gridExtra 2.2.1, gtable 0.2.0, highr 0.6, impute 1.48.0, iterators 1.0.8, knitr 1.14, lattice 0.20-34, latticeExtra 0.6-28, magrittr 1.5, matrixStats 0.51.0, munsell 0.4.3, nnet 7.3-12, partykit 1.1-1, pheatmap 1.0.8, plyr 1.8.4, preprocessCore 1.36.0, rpart 4.1-10, scales 0.4.0, splines 3.3.1, stats4 3.3.1, stringi 1.1.2, stringr 1.1.0, survival 2.39-5, tools 3.3.1

References

[1] Ka Yee Yeung and Walter L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, 2001.

[2] Ken I Mills, Alexander Kohlmann, P Mickey Williams, Lothar Wieczorek, Wei-min Liu, Rachel Li, Wen Wei, David T Bowen, Helmut Loeffler, Jesus M Hernandez, et al. Microarray-based classifiers and prognosis models identify subgroups with distinct clinical outcomes and high risk of aml transformation of myelodysplastic syndrome. *Blood*, 114(5):1063–1072, 2009.

[3] Bin Zhang, Chris Gaiteri, Liviu-Gabriel Bodea, Zhi Wang, Joshua McElwee, Alexei A Podtelezhnikov, Chunsheng Zhang, Tao Xie, Linh Tran, Radu Dobrin, et al. Integrated systems approach identifies genetic nodes and networks in late-onset alzheimer's disease. *Cell*, 153(3):707–720, 2013.

[4] A Foroushani, R Agrahari, R Docking, A Karsan, and H Zare. Large-scale gene network analysis reveals the significance of extracellular matrix pathway and homeobox genes in acute myeloid leukemia. *In preparation*.