Kmer distance: Pairwise distance base on kmer strategy

Cai Huang

Georgia Instutute of Technology,
Atlanta, GA, USA
chuang95@gatech.edu

Fredrik O. Vannberg

Georgia Instutute of Technology,
Atlanta, GA, USA
fredrik.vannberg@biology.gatech.edu

July 30, 2014

Abstract

To date the comparison of genomic DNA sequences have routinely utilized shorter conserved regions for comparative genomics. Current phylogenetic analysis therefore can create divergent results based on which genetic loci are utilized for this analysis. Sequence similarity is also commonly determined by first carrying out gap penalty pairwise alignments for a set of sequences, and the similarity is quantified based upon this alignment. Here we provide a R package which has two kmer base algorithms to compute the pairwise comparison of genomic DNA sequence. First function is Boolean analysis by using XOR discrete function, second function is linear algebra analysis by using hierarchical PCA.

Contents

1 Loading the R package 1

2 Compute pairwise distance by using XOR 1

3 Compute pairwise distance by using hierarchical PCA 2

4 Compare the results generated by XOR and hierarchical PCA 2

5 Session info 3

1 Loading the R package

First, download and install the *kmerDistance* R package. Launch R and load the package.

```
require(Matrix)
```

```
## Loading required package:  Matrix
## Loading required package:  methods
```

```
library(kmerDistance)
```

The examples in this document are integer kc files generated by using Kanalyze[1] with following command.

$$\mathrm{ls} * \mathrm{fna}|\mathrm{xargs} - \mathrm{Ionejava} - \mathrm{jarkanalyze.jarcount} - \mathrm{k8} - \mathrm{oone\_8.kc} - \mathrm{ffasta} - \mathrm{pkanalyze.outfmt} = \mathrm{int} - \mathrm{rone}$$

The kc files contents kmer and counts for each genome sequence. In our test data set, there are 6 kc files generated from 6 virus genome sequences, and we choose kmer length equal to 8

2 Compute pairwise distance by using XOR

The first function `kmerDistance.dif` compute pairwise distance of kmer data by using XOR discrete function, and we call it boolean analysis. We can call this function.

```
x <- kmerDistance.dif(8,"../../kmerDistance/data/") #kmer.length = 8, path.to.data to data folder
```

The `x` return value is a distance matrix. Each column and row represents one virus genome. We can check the distance matrix.

```
names(x)
```

```
## [1] "Mycobacterium.phage.Phaedrus.complete.genome.fna_8.kc"
## [2] "Mycobacterium.phage.Phlyer.complete.genome.fna_8.kc"
## [3] "Mycobacterium.phage.Pipefish.complete.genome.fna_8.kc"
## [4] "Staphylococcus.phage.55.complete.genome.fna_8.kc"
## [5] "Staphylococcus.phage.69.complete.genome.fna_8.kc"
## [6] "Staphylococcus.phage.71.complete.genome.fna_8.kc"
```

```
names(x) <- c(1:6)
x
```

```
##         1       2       3      4      5      6
## 1 0.00000 0.04437 0.10764 0.6296 0.6423 0.6327
## 2 0.04437 0.00000 0.09808 0.6282 0.6429 0.6326
## 3 0.10764 0.09808 0.00000 0.6255 0.6399 0.6290
## 4 0.62958 0.62823 0.62552 0.0000 0.2745 0.2079
## 5 0.64226 0.64290 0.63994 0.2745 0.0000 0.2796
## 6 0.63266 0.63260 0.62903 0.2079 0.2796 0.0000
```

First 3 virus are *Mycobacterium.phage* and last 3 are *Staphylococcus.phage*. From the matrix, we see the pairwise distance among first 3 genomes and last 3 genomes are lower than the pairwise distance across the them. Base on this, we can clearly cluster them to two family.

3 Compute pairwise distance by using hierarchical PCA

The second function. `kmerDistance.hpca` compute the pairwise distance of kmer data by using hierarchical PCA, and we call it linear algebra analysis. We can call this function.

```
y <- kmerDistance.hpca(8,"../../kmerDistance/data/") #kmer.length = 8, path.to.data to data folder
```

This function also returns y value as a distance matrix. Each column and row represents one virus genome. We can check the distance matrix.

```
names(y)
```

```
## [1] "Mycobacterium.phage.Phaedrus.complete.genome.fna_8.kc"
## [2] "Mycobacterium.phage.Phlyer.complete.genome.fna_8.kc"
## [3] "Mycobacterium.phage.Pipefish.complete.genome.fna_8.kc"
## [4] "Staphylococcus.phage.55.complete.genome.fna_8.kc"
## [5] "Staphylococcus.phage.69.complete.genome.fna_8.kc"
## [6] "Staphylococcus.phage.71.complete.genome.fna_8.kc"
```

```
names(y) <- c(1:6)
y
```

```
##          1        2        3       4     5       6
## 1 0.000000 0.001199 0.005339 0.92827 1.154 0.89736
## 2 0.001199 0.000000 0.005555 0.92876 1.155 0.89793
## 3 0.005339 0.005555 0.000000 0.92916 1.152 0.89808
## 4 0.928267 0.928765 0.929163 0.00000 1.260 0.07331
## 5 1.154281 1.155395 1.151979 1.25968 0.000 1.18652
## 6 0.897357 0.897929 0.898078 0.07331 1.187 0.00000
```

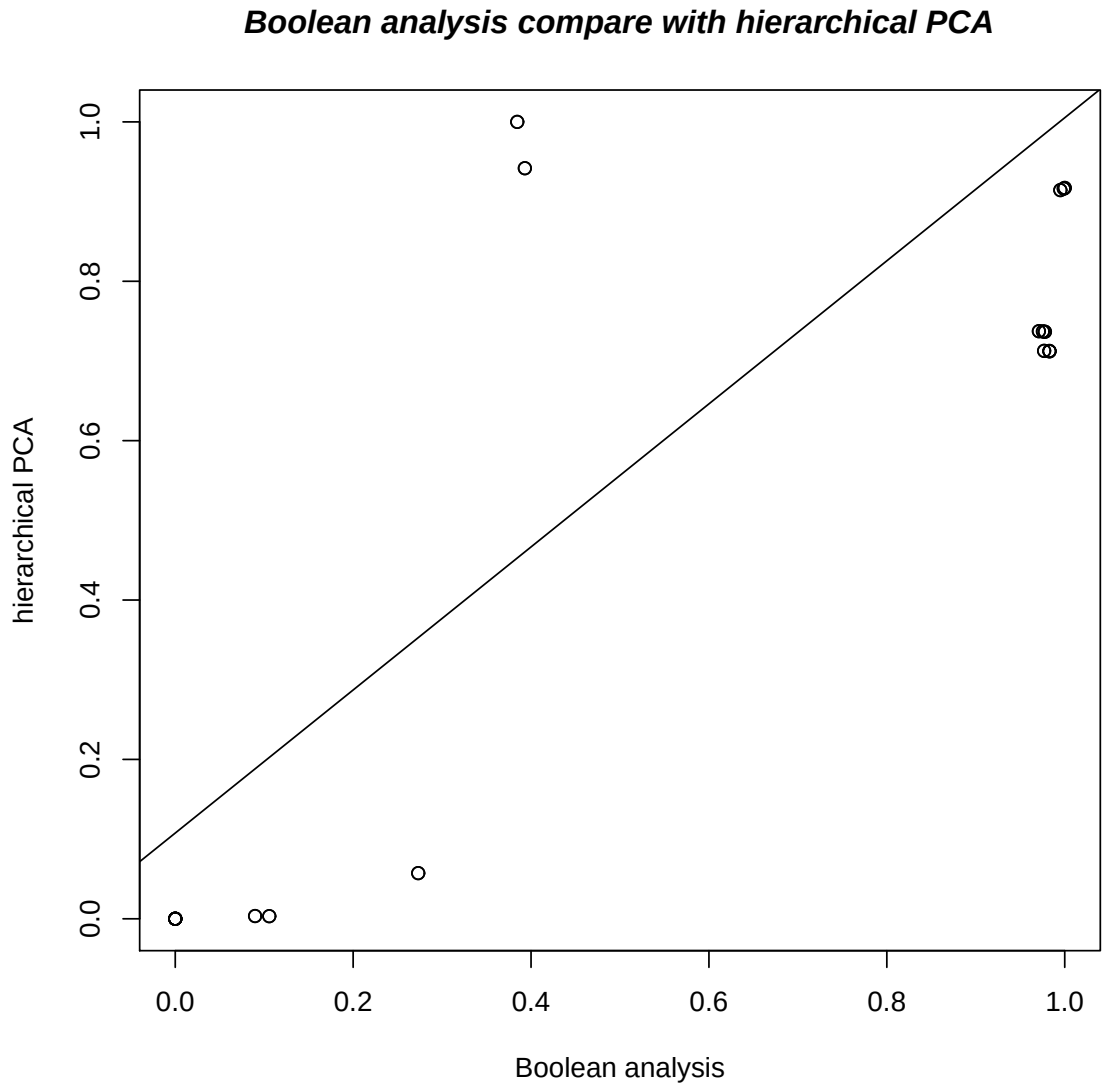
Similarly, we see the pairwise distance are clearly separating the two group of viruses.

4 Compare the results generated by XOR and hierarchical PCA

Further more, we plot the x and y pairwise distance matrix together, and show the nice correlation between our two algorithm.

```
#prepare the result as vector
x=as.vector(data.matrix(x))
y=as.vector(data.matrix(y))
#normalize the data to between 0 and 1
```

```
x[x>0]=(x[x>0]-min(x[x>0]))/(max(x[x>0])-min(x[x>0]))
y[y>0]=(y[y>0]-min(y[y>0]))/(max(y[y>0])-min(y[y>0]))
#fit a linear regression model to show the correlation between x and y
reg=lm(x~y)
plot(x, y,xlab="Boolean analysis",ylab="hierarchical PCA")
title(main="Boolean analysis compare with hierarchical PCA", col.main="black", font.main=4)
abline(reg)
```




```
#show the coefficient of determination
summary(reg)$r.squared
```

```
## [1] 0.6914
```

From the plot we can see the linear regression model shows a good correlation between our two function. Also the dots are located to two separate parts of the plot, which give us the evidence of two clusters among the genome data.

5 Session info

```
sessionInfo()
```

```
## R version 3.0.3 (2014-03-06)
## Platform: x86_64-apple-darwin10.8.0 (64-bit)
##
```

```
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] methods   stats     graphics  grDevices utils     datasets  base
##
## other attached packages:
## [1] kmerDistance_0.0.2 Matrix_1.1-4       knitr_1.6
##
## loaded via a namespace (and not attached):
## [1] evaluate_0.5.5  formatR_0.10    grid_3.0.3      highr_0.3
## [5] lattice_0.20-29 stringr_0.6.2   tools_3.0.3
```

References

[1] Audano, P. and Vannberg, F. (2014). Kanalyze: a fast versatile pipelined k-mer toolkit. *Bioinformatics*, **30**(14), 2070–2.