Stability ranking with the staRank package

Juliane Siebourg

April 16, 2015

Stability ranking can be used to obtain variable rankings that are highly reproducible. Such rankings are needed for robust variable selection on univariate data. In most biological experiments several replicate measurement for a variable of interest, e.g. a gene, are available. An obvious way to combine and prioritize these values is to take their average or median. Other procedures like a t-test or rank sum test deliver a statistic that also accounts for the variance in the data. Stability ranking provides a way of ranking measured elements based on one of the methods above and combining it with a bootstrapping routine to estimate the stability with which an element occurs within the top k set of the ranking. The final ranking orders the elements accoring to this stability. The theory behind the procedure is described in [1]. Stability selection in general is described in [2]. This file contains two example stability analysis that show how the staRank package should be used. In a first toy example, stability selection using all available base ranking methods is performed on simulated data. In a second example, an RNAi dataset on Salmonella infection [3] is analyzed.

Simulation example

First we create an artificial dataset of p genes (rows) with n replicate measurement values (columns). For each gene the effects are drawn from a normal distribution and their replicate variance is drawn from a gamma distribution.

```
> # sample parameters for data
> p<-20        # genes
> n<-4        # replicates
> trueEffects<-rnorm(p,0,5) # gene effect
> s<-rgamma(p,shape=2.5,rate=0.5) # sample variance
> # draw n replicate values of a gene
> simData <- matrix(0,nr=p,nc=n,
+                 dimnames=list(paste("Gene",c(1:p)),paste("Replicate",c(1:n))))
> for(i in 1:p){
+         simData[i,]<-replicate(n,rnorm(1,mean=trueEffects[i],sd=s[i]))
+ }
```

Now stability ranking can be performed on the dataset, to find top scoring genes that are highly reproducible. Stability ranking can be applied using different base ranking methods. Implemented in the package are the mean, median and the test statistics of the rank sum test (Mann-Whitney test), t-test and RSA [4].

```
> # load the stability ranking package
> library(staRank)
> # implemented ranking methods
> method<-c('mean','median','mwtest','ttest','RSA')
```

The ranking is performed calling the main function stabilityRanking. Here this is done using the default settings but for different base methods.

```
> stabilityList<-list()
> stabilityList[['mean']]<-stabilityRanking(simData,method='mean')
```

```
> stabilityList[['median']]<-stabilityRanking(simData,method='median')
> stabilityList[['mwtest']]<-stabilityRanking(simData,method='mwtest')
> stabilityList[['ttest']]<-stabilityRanking(simData,method='ttest')
> stabilityList[['RSA']]<-stabilityRanking(simData,method='RSA')
```

The stability ranking proceedure can also be used with any external ranking method. This means `stabilityRanking` can be called on rankings directly. All that is needed is a named vector of the original ranking and a matrix containing sample rankings. In the following this is demonstrated by performing a mean ranking on the data and on bootstrap samples of it.

```
> # 1. create a ranking
> extRanking <- sort(apply(simData,1,mean))
> # 2. create 100 sample rankings
> sampleRankings<-matrix(NA,nr=nrow(simData),nc=100)
> for(i in 1:100){
+         sampleRankings[,i]<-
+                         apply(simData,1,function(x){mean(sample(x,length(x),replace=TRUE))})
+ }
> rownames(sampleRankings)<-rownames(simData)
> sampleRankings<-sampleRankings[names(extRanking),]
> # 3. run stabilityRanking
> stabilityList[['externalMean']]<-
+                 stabilityRanking(extRanking,sampleRankings,method='externalMean')
```

In this case the parameter `method` is only needed for identification.

We can now examine the results of the procedure. The function returns an object of type `RankSummary`. This contains three different rankings of the data. The base, stability and averaged ranking. The averaged ranking is also an aggregated ranking that averages the rank of a variable over all samples.

```
> #results for the mean ranking
> baseRank(stabilityList$mean)

    Gene 18     Gene 19     Gene 13     Gene 10     Gene 17     Gene 14
-15.6111326 -12.2208620  -7.9097433  -1.8254773  -1.7473269  -1.1459189
     Gene 4      Gene 8     Gene 15      Gene 6      Gene 2     Gene 16
 -0.6258728  -0.5382658   1.1107324   2.1342636   2.1964025   2.9819957
     Gene 5     Gene 20      Gene 3      Gene 1      Gene 7     Gene 11
  3.7851368   5.9521591   6.1883471   7.3588396   7.5875529   8.6615898
     Gene 9     Gene 12
 10.0798754  12.9037322

> stabRank(stabilityList$mean)

Gene 18 Gene 13 Gene 19 Gene 17 Gene 10  Gene 8 Gene 14  Gene 6  Gene 4 Gene 16
    1.0     2.5     2.5     4.0     5.0     6.5     6.5     8.0     9.5     9.5
 Gene 5 Gene 15 Gene 20  Gene 3  Gene 1  Gene 2  Gene 7 Gene 11  Gene 9 Gene 12
   11.5    11.5    13.0    14.0    16.0    16.0    16.0    18.0    19.5    19.5

> avrgRank(stabilityList$mean)

Gene 18 Gene 19 Gene 13 Gene 17 Gene 10 Gene 14  Gene 8  Gene 4 Gene 15  Gene 6
   1.19    1.97    2.89    5.54    6.12    6.64    7.62    7.75    8.76   10.06
 Gene 2 Gene 16  Gene 5 Gene 20  Gene 3  Gene 1  Gene 7 Gene 11  Gene 9 Gene 12
  10.67   11.32   12.12   14.40   14.71   16.29   16.67   17.63   17.84   19.81
```

This can be summarized in a matrix with `getRankmatrix(stabilityList$mean)`.

For comparison of the three rankings we also compute their correlation using Spearman's rank correlation coefficient.

```
> rankCor(stabilityList[['mean']])

           stability      base   average
stability 1.0000000 0.9558712 0.9619067
base      0.9558712 1.0000000 0.9969925
average   0.9619067 0.9969925 1.0000000
```

Another value that is returned is a vector containing the sizes of the stable sets per cutoff. This gives information about how stable the base ranking itself is.

```
> stableSetSize(stabilityList[['mean']])

 [1]  0  1  3  3  3  3  4  5  7  7  8 10 10 12 12 13 14 17 18 20
```

We can for example compare this property for all the base ranking methods.

```
> # plot the stable genes per cutoff k* for each method
> mCol<-rainbow(5)
> name_out<-c('Mean','Median','Rank sum','t-test','RSA')
> plot(1,type='n',xlim=c(0,p),ylim=c(0,p),xlab='k*',ylab='stable genes')
> abline(0,1,col='lightgray')
> for(i in 5:1){
+         lines(stableSetSize(stabilityList[[i]]),lwd=2,col=mCol[i])
+ }
> legend('topleft',name_out,col=mCol,lwd=2)
```

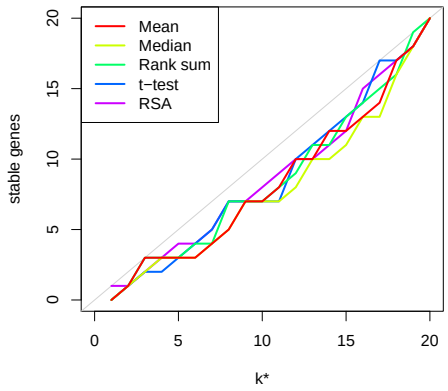

Figure 1: Stable set sizes for each cutoff using different ranking methods on the toy example.

The gray diagonal in this plot indicates perfect stability.

Stability ranking of an RNAi dataset

The package contains an RNAi screen on human cells under *Salmonella* infection. Briefly, in HeLa cells all genes were knocked-down individually by RNA interference. Subsequently they were infected with *Salmonella*. The cells were imaged with a microscope and from the images several features were extracted (for details see original paper [3]). After loading the data we select all genes where the infection on average decreased more than one standard deviation. Then we run stability ranking on this selection, for mean, rank sum test and RSA as base ranking methods.

```
> #load the data set
> data(salmonellaInfection)
> data<-as.matrix(salmonellaInfection)
> down<-rownames(data[which(apply(data,1,mean)<=-1),])
> data<-data[down,]

> salmonellaStability<-list()
> salmonellaStability[['mean']]<-stabilityRanking(data,method='mean')
> salmonellaStability[['mwtest']]<-stabilityRanking(data,method='mwtest')
> salmonellaStability[['RSA']]<-stabilityRanking(data,method='RSA')
```

The correlation shows that the stability ranking leads to a more diverged result.

```
> rankCor(salmonellaStability$mwtest)

          stability      base   average
stability 1.0000000 0.7997028 0.7675219
base      0.7997028 1.0000000 0.9675864
average   0.7675219 0.9675864 1.0000000
```

Plotting the data points and their mean values according to their ranking shows the differences of the methods. For this we first gather all rankings in a matrix.

```
> # gather all rankings in a matrix
> rankings<-lapply(salmonellaStability,getRankmatrix)
> rankings<-Reduce(cbind,lapply(rankings,function(x){x[down,]}))
```

Then, for each gene the three data points and and their mean are plotted at their position in the ranking (x-axis).

```
> # plot data according to rankings
> par(mfrow=c(1,5),mar=c(4,4,2,2))
> range<-range(data)
> for(m in c(2,3,1,4,8)){
+         plot(1,type='n',xlim=c(0,length(down)), ylim=range,main='',
+                 ylab='infection score', xlab=paste(colnames(rankings)[m],'rank'))
+         orderedData<-data[order(rankings[,m]),]
+         for(j in 1:3){
+                 points(orderedData[,j],pch=20,cex=0.5)
+         }
+         points(apply(orderedData,1,mean),col='red',pch=20,cex=0.5)
+ }
```

One can observe that the overall variance in this dataset is quite high, resulting from the fact, that different siRNAs were used for each gene. This also becomes obvisous when looking at the stable set sizes of the original rankings.

```
> # plot the stable genes per cutoff k* for each method
> mCol<-rainbow(3)
> name_out<-c('Mean','Rank sum','RSA')
> plot(1,type='n',xlim=c(0,length(down)),ylim=c(0,length(down)),
+                 xlab='k*',ylab='stable genes')
> abline(0,1,col='lightgray')
> for(i in 1:3){
+         lines(stableSetSize(salmonellaStability[[i]]),lwd=2,col=mCol[i])
+ }
> legend('topleft',name_out,col=mCol,lwd=2)
```

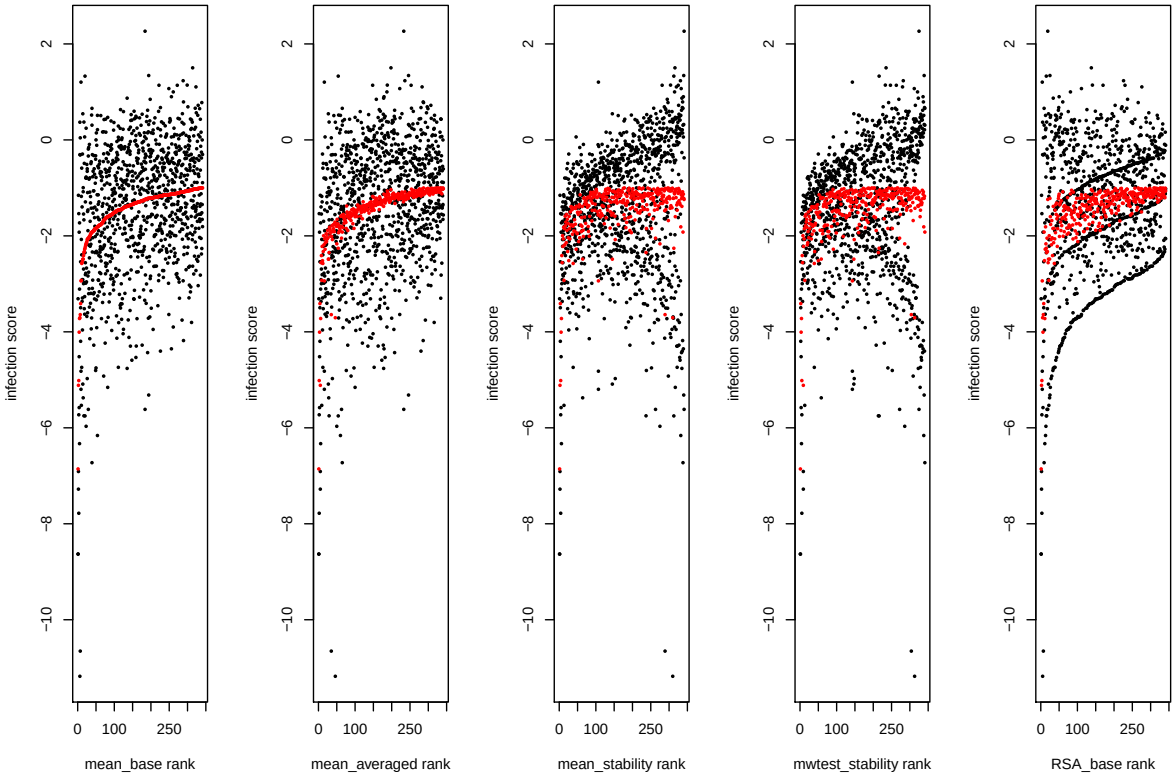

Figure 2: The infection data plotted according to a ranking (x-axis). Black dots are infection values, plotted columnwise per gene, red dots are their mean.

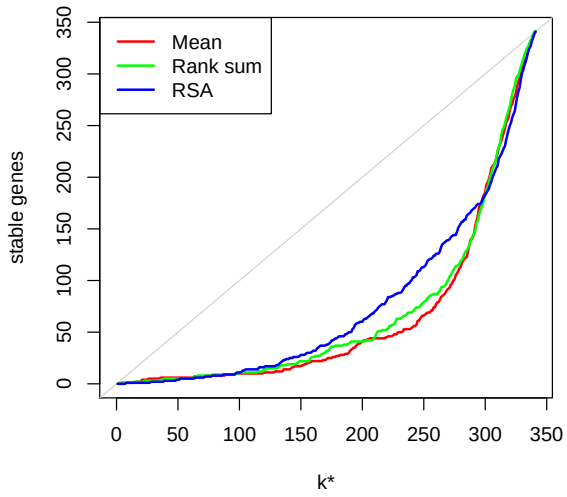

Figure 3: Stable sets of rankings on the salmonella infection data.

Within the top 10% of the ranking (35 genes) only 5 are stable. Calling `summary` on a `RankSummary` object we can quickly access the top 1, 10 and 50% cutoffs and stable set sizes.

```
> summary(salmonellaStability$mean)

     Length       Class        Mode
          1 RankSummary          S4
```

References

[1] Siebourg J., Merdes G., Misselwitz B., Hardt W.-D.and Beerenwinkel N. *Stability of gene rankings from RNAi screens*. Bioinformatics, 2012, doi: 10.1093/bioinformatics/bts192

[2] Meinshausen N. and Bühlman P. *Stability selection*. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 2010

[3] Misselwitz B. et al., *RNAi screen of Salmonella invasion shows role of COPI in membrane targeting of cholesterol and Cdc42*. Molecular Systems Biology, 2011

[4] Koenig R. et al., *A probability-based approach for the analysis of large-scale RNAi screens*. Nature Methods, 2007