Supplement: Splice Expression Variation Analysis (SEVA): Variability Analysis to Detect Significant Alternative Splicing Events (TCGA Analysis only)

Bahman Afsari

Sunday, June 12, 2016

1 Loading the real data

First, we load the data as:

```
library('Homo.sapiens')
library('org.Hs.eg.db')
library('GenomicRanges')
library("GSReg")
library(EBSeq)
library(limma)
library('gplots')
library(ggplot2)
library('ROCR')
library(Matrix)
library(qvalue)
```

2 Cross-study Validation with TCGA

Now, we cross-study the genes we identified using TCGA as the tes-set. (For memory issues we run SEVA on the batches of 1000 genes and augmented the results)

```
#####################################33
### TCGA
##
### loading data
#########################



load("../Data/TCGA/HPVPosTCGAJuncRPM_25Aug2015_Pheno.Rda")
load("../Data/TCGA/TCGA_RSEM_processed_091615.RDa")
load(file = "../Cache/ForTCGAAnalysis.rda")


TCGA.RSEM <- as.matrix(TCGA.RSEM)

#We divide the TCGA data to batches of 1000 genes to make it manageable
junctionPValueTCGAaug <- c()
for( i in 1:14){
```

```
  junctionPValueTCGA <- GSReg.SEVA(junc.RPM=junc.RPM.TCGA,
                                                                phenoVect=as.factor(phenoVect.TCGA
                                 sparse = T,
                                 verbose = F,
                                 geneexpr=TCGA.RSEM,
                                 minmeanloggeneexp= 3,
                                 GenestoStudy =
                                          as.vector(na.omit(intersect(names(junctionPValue),
                                                                      rownames(TCGA.RSEM))
                                                                 [(1:1000)+i*1000])))


  # junctionPValueTCGA <- SEVA.meangeneFilter(junc.RPM=junc.RPM.TCGA,
  #                                           phenoVect=phenoVect.TCGA,
  #                                           geneexpr=TCGA.RSEM,
  #                                           minmeanloggeneexp= 3,
  #                                           GenestoStudy =
  #                                             as.vector(na.omit(intersect(names(junctionPValue),
  #                                                                         rownames(TCGA.RSEM))
  #                                                                    [(1:1000)+i*1000])))
  gc()
  junctionPValueTCGAaug <- c(junctionPValueTCGAaug,junctionPValueTCGA)
}
```

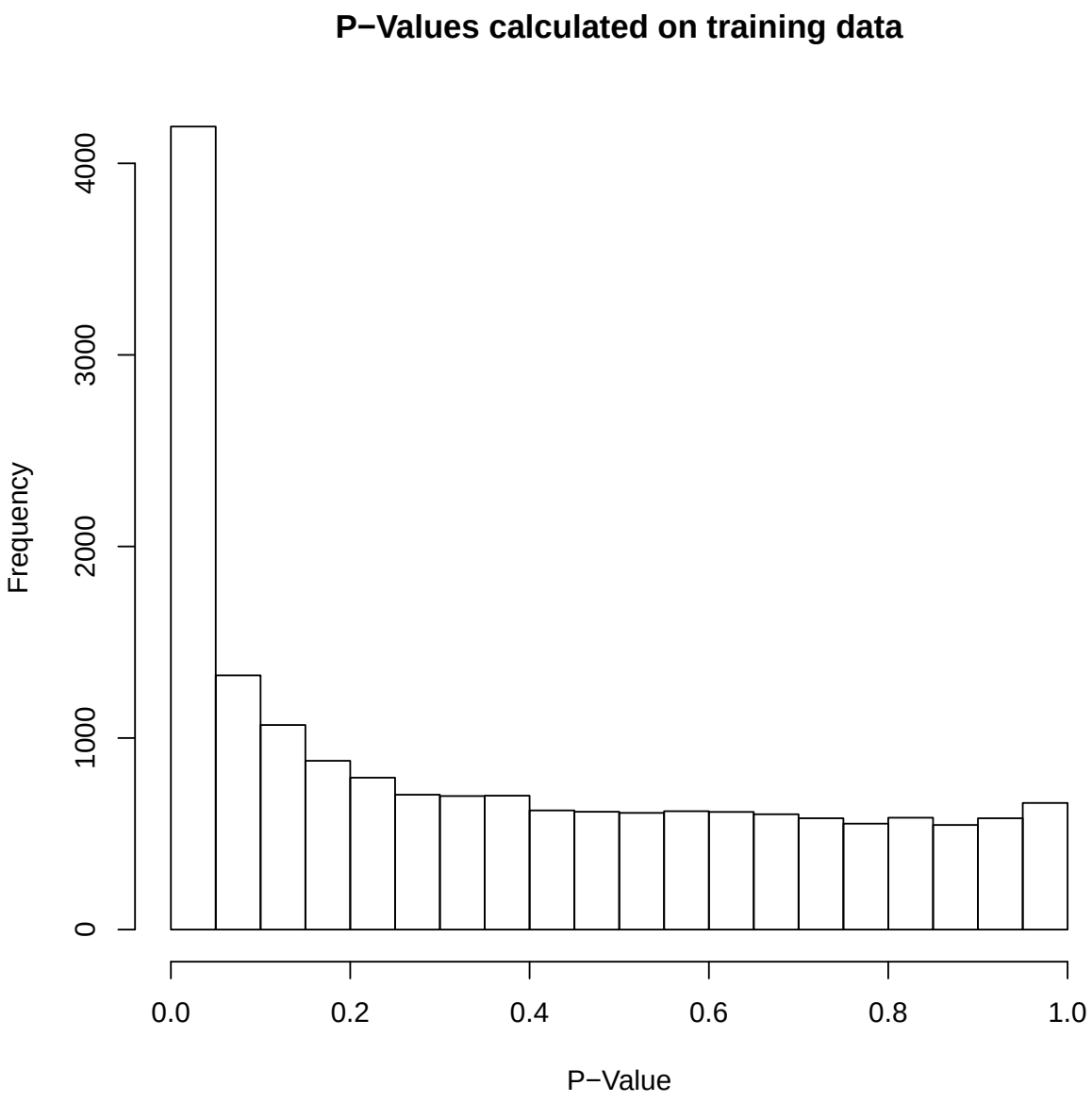
```
## Warning in sqrt(myvartotal): NaNs produced

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)
```

```
## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)

## Warning in match(as.vector(x), y, 0L): Reached total allocation of 8010Mb:
## see help(memory.size)
```

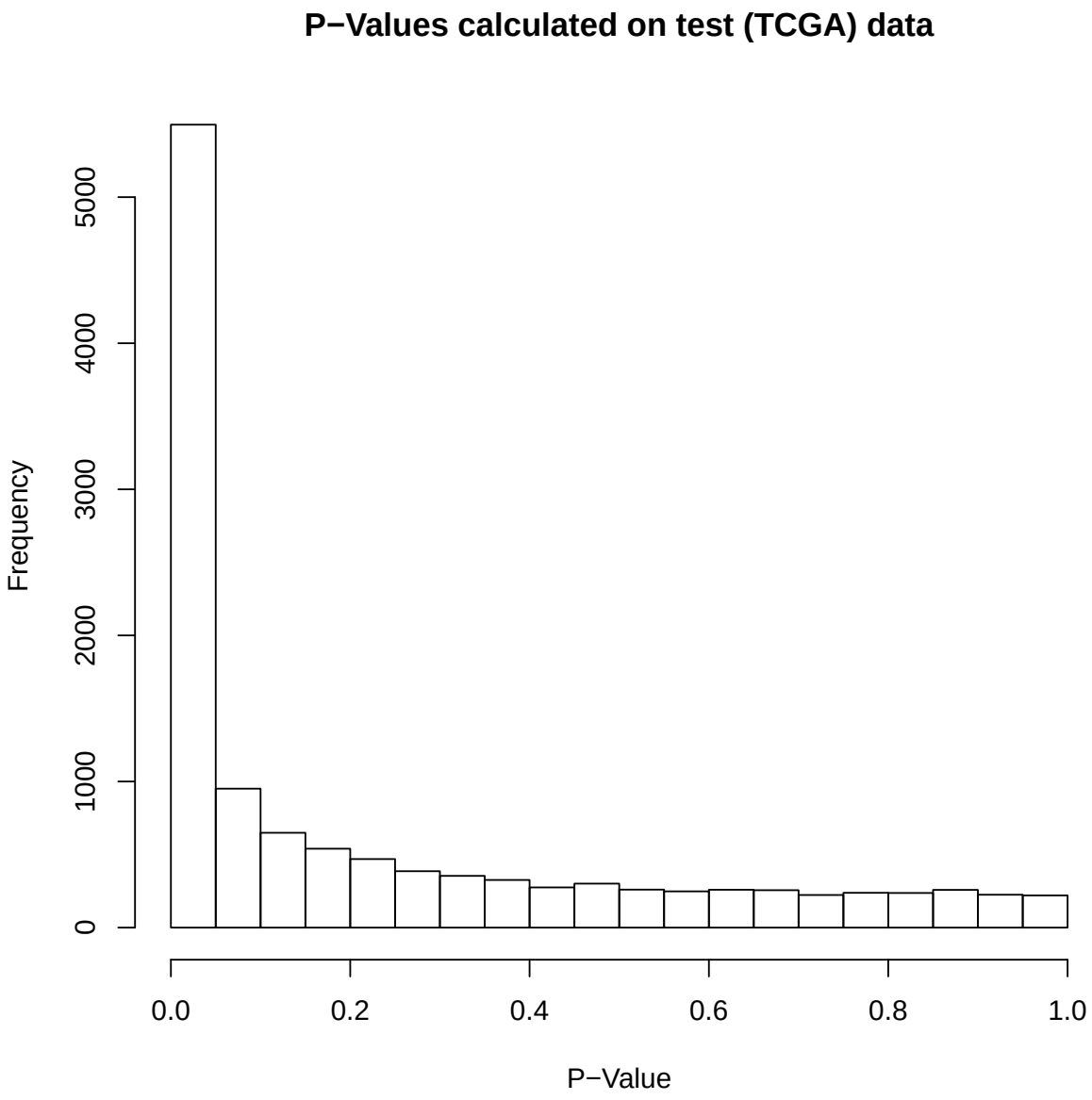
```
save(list=c("junctionPValueTCGA","junctionPValueTCGAaug"),file = "../Cache/junctionPValueTCGA.rda")
```

Now, checking if the genes identified on the original data generates enriched p-values on the TCGA data.

```
load(file = "../Cache/junctionPValueTCGA.rda")
load(file = "../Cache/SEVAJoe.rda")


#Pvalues based on  the original data (Training data a.k.a. Joe's Data)
originaldatapval <- sapply(junctionPValue,function(x) x$pvalue) #All P-values on the Training set
hist(x = originaldatapval,
     xlab="P-Value",main=paste("P-Values calculated on training data"))
```



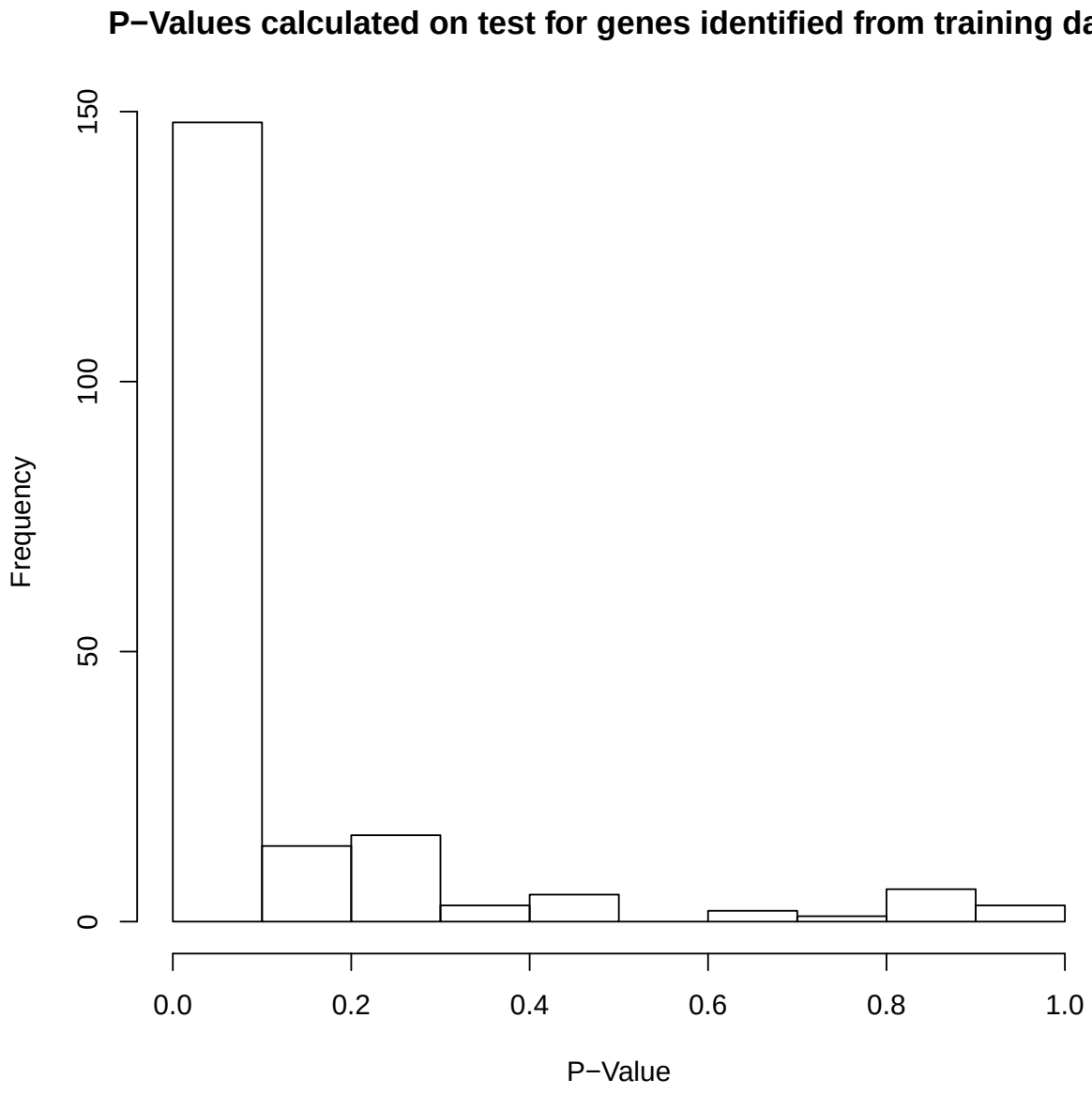

```
SEVATCGAGenes <- intersect(names(junctionPValueTCGAaug),
                                    SEVAGenesPure)


tcgaallpvals <- sapply(X = junctionPValueTCGAaug,FUN = function(x) x$pvalue)

hist(x = tcgaallpvals,
     xlab="P-Value",main=paste("P-Values calculated on test (TCGA) data"))
```




```
print(cor.test(tcgaallpvals,originaldatapval[names(tcgaallpvals)],method = "spearman"))
```

```
##
##  Spearman's rank correlation rho
##
## data:  tcgaallpvals and originaldatapval[names(tcgaallpvals)]
## S = 2.6734e+11, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.1103273
```

```
tcgapval <- sapply(junctionPValueTCGAaug[SEVATCGAGenes],function(x) x$pvalue)


hist(x = tcgapval[SEVATCGAGenes],
     xlab="P-Value",main=
       paste("P-Values calculated on test for genes identified from training data"))
```




```
cat("percentage that survived on test",
    mean(tcgapval<0.01/length(tcgapval)))
```

```
## percentage that survived on test 0.3232323
```

```r
cat("Quatile of the p-value distribution SEVA genes using TCGA data")
```

```
## Quatile of the p-value distribution SEVA genes using TCGA data
```

```r
print(quantile(tcgapval))
```

```
##           0%          25%          50%          75%         100%
## 0.000000e+00 4.267376e-06 5.585362e-03 1.093300e-01 9.791673e-01
```

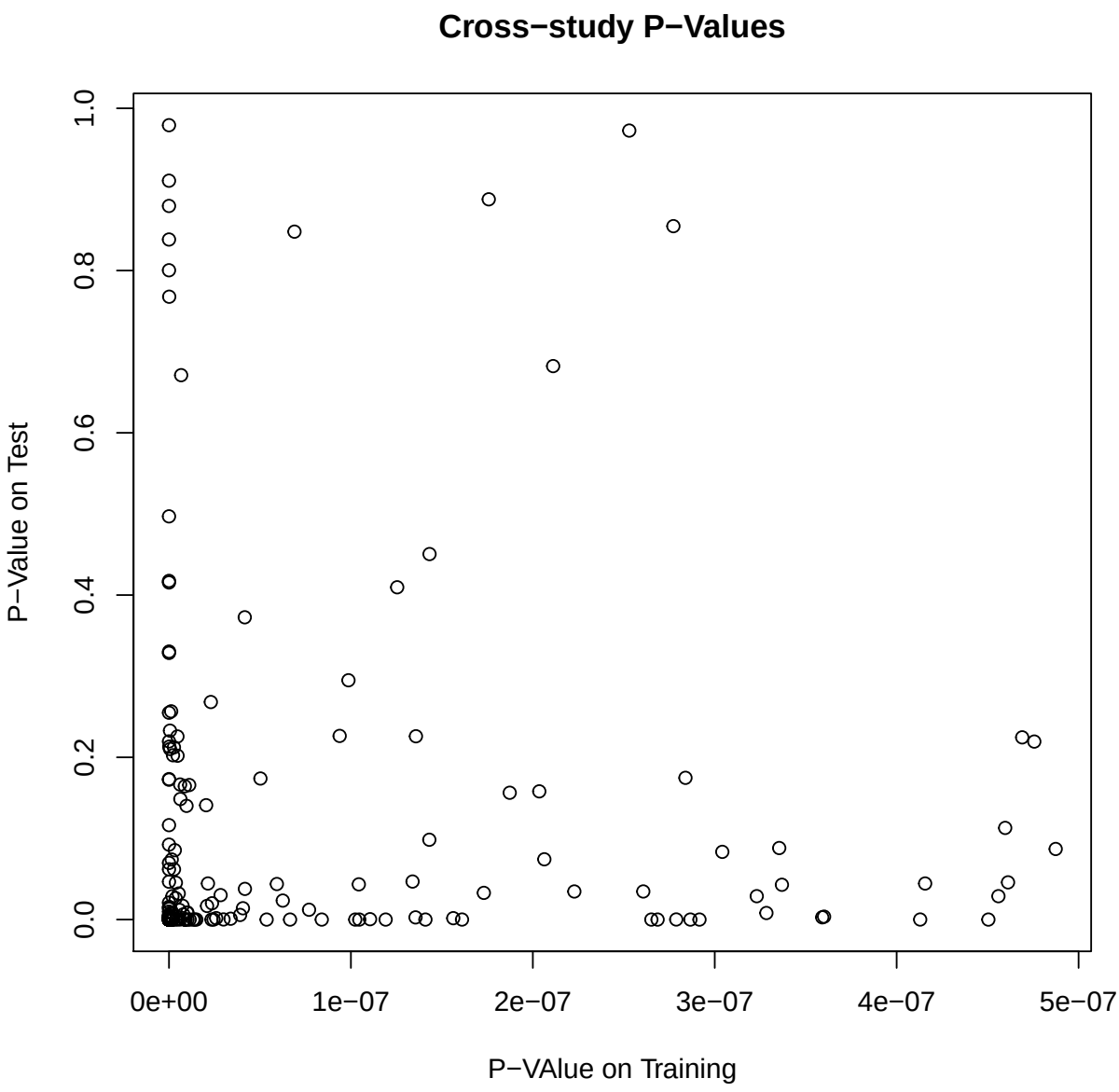
```r
tcgaallpvals <- sapply(X = junctionPValueTCGAaug,FUN = function(x) x$pvalue)

cat("Enrichment of the p-values on the test data for the genes identified from training.")
```

```
## Enrichment of the p-values on the test data for the genes identified from training.
```

```r
wilcox.test(index=SEVATCGAGenes,sapply(junctionPValueTCGAaug,function(x) abs(x$zscore)),alternative = "
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  sapply(junctionPValueTCGAaug, function(x) abs(x$zscore))
## V = 74073000, p-value < 2.2e-16
## alternative hypothesis: true location is greater than 0
```

```r
plot(originaldatapval[names(tcgapval)],
     tcgapval,
     ylab = "P-Value on Test",
     xlab = "P-VAlue on Training",
     main ="Cross-study P-Values")
```




```
#Nperm <- 5000
#randpi0 <- vector(mode = "numeric",length = Nperm)
#set.seed(1)
#for( i in 1:Nperm){
#  randompvalue <- sample(tcgaallpvals,size = length(tcgapval))
#  randpi0[i] <- qvalue(randompvalue)$pi0
#}



cat("Quatile of the p-value distribution random genes using TCGA data")
```

```
## Quatile of the p-value distribution random genes using TCGA data
```

```r
print(quantile(tcgaallpvals))
```

```
##           0%          25%          50%          75%         100%
## 0.000000000 0.002977169 0.077212061 0.392145388 0.999967804
```

```r
#print(wilcox.test(x=tcgapval,y=tcgaallpvals,alternative = "less",conf.int = T,conf.level = 0.95))


save(list=ls(),file = "../Cache/SEVATCGA.rda")
```