Scatterplot analysis for the integrative analysis of expression and methylation data

Sánchez-Pla, Alex[1,3], Miró, Berta[3], Carmona, Francesc[1],
Bazzoco, Sara[2], Arango, Diego[2]
[1] Departament of Genetics Microbiology and Statistics, Universitat de Barcelona
[2] CIBBIM-Nanomedicine. Biomedical Research in Digestive Tumors, (VHIR), Barcelona
[3] Statistic and Bioinformatics Unit. Vall d'Hebron Research Institute. (VHIR). Barcelona

Contents

1 Introduction

This document describes how to implement a pipeline for selecting L-shaped genes from two data matrices, an expression and a methylation matrix, matched by rows (genes) and columns(samples), using the algorithms implemented by the authors and described elsewhere.
The idea is to "keep-it-simple" so that it the process can be followed by a non-technical reader.
The steps correpond roughly to what one user would do when exploring two expression-methylation datasets

1. Data input: For each dataset
   (a) Select (Load) expression values from a csv (or binary) file
   (b) Select (Load) methylation values from a csv (or binary) file
   (c) Check that the data are well composed and matched
2. For each selection method (NaÃŕve, CMI, Heuristic)

    (a) Set parameters if needed)
    (b) Compute binary and numeric scores to for each gene
    (c) Output result in tab format: geneId, NumScore, binScore

   This will yield a results table (a list) for each method.

3. Define consensus list of l-shaped genes (e.g. by intersection or union or selection of output L-lists)

4. Plot consensus lists scatterplots

5. Compute genome positions of selected genes and

    (a) Plot genes in chromosome
    (b) Add gene positions to results table

6. Do enrichment analysis of the selecte gene list

```
## NULL
```

```
addToLinksFile("GRMSelectionPipeline.csv", "Informe")
```

2 Datasets used for the study

2.1 On the preprocessing and selection of the researcher's data

In a first version of this analysis the data for microarrays, RNAseq and methylation were used.
In further analyses we have concluded that RNAseq data were less reliable than microarray data so that we decided to omit RNAseq data from then on.
As a consequence we can use what we called "2 way data" that is, data that have been matched by pairs which have 30 instead of 25 samples.
The preprocessing of the data of microarrays and methylation was done in 3 processes:

1. `CorrelationAnalysis0New-ReadAndPreprocessDataV2`

2. `CorrelationAnalysis1NEW-MatchData`

3. `CorrelationAnalysis2NEW-ComputeCorrs`

The analysis performed in `CorrelationAnalysis0New-ReadAndPreprocessDataV2` has produced two methylation datasets. One where cpgs have been unitized to genes by averaging their values and another where this has been done by taking the probe with highest variance.
The analysis performed in `CorrelationAnalysis1NEW-MatchData` has selected common genes and common samples between methylation and microarray data. Datasets `dataMarrCR2`, `dataMethCR2` and `dataMethVarCR2` contain microarray expression and methylation (unitized by mean and by variance) data matched by gene and samples. These datasets have been saved into a binary file: `matchedMarrRNAseqMethDataByPAIRS.Ro`
and written separately to files `matched2MicroarrayData.csv`, `matched2MethylationData.csv` and `matched2MethVar.cs`
These will be the datasets used as "researcher's data in the analyses

2.2 Colon Cancer datasets

There are three[1] datasets available for analysis obtained from three sources. The first dataset has been obtained by the researcher, (Diego Arango) and will be named as `DA`. A second one, which we name as `GEO`, has been extracted from the Gene Expression Omnibus and the third dataset is the Colon Cancer dataset from The Cancer Genome Atlas project and will be designated as `TCGA`.

1. `DA`: Expression microarrays (`matched2MicroarrayData.csv`) and Illumina 25k methylation arrays (`matched2MicroarrayData.csv`) on 30 cell llines.

[1] An RNAseq dataset was also available but we decided not to use it because we were unsure of how reliable the values were

2. `GEO`: Expression microarrays (`geoMicroarrays.csv`) and Illumina 25k methylation arrays (`geoMetilacion.csv`) on 25 CRC samples.

3. TCGA. Expression microarrays (`TCGAExprData.csv`) and Illumina 25k methylation arrays (`TCGAMetilData.csv`) on 426 Colon Adenocarcinoma (`COAD` dataset) samples from TCGA.

A summary of each datasets follows below:

- `DA` dataset has been provided by the researcher (DA="Diego Arango") and is the original dataset on which it was intended to select L-shaped genes.

```
require(printr)
load(file.path(dadesDir,"matchedMarrRNAseqMethDataByPAIRS.Rda"))
DAExprData <- as.matrix(dataMarrCR2)
DAMetilData <- as.matrix(dataMethCR2)
rm(dataMarrCR0,dataMarrCR2, dataMethCR1, dataMethCR2, dataMethVarCR1, dataMethVarCR2, dataRNAse
#write.csv2(DAExprData, file=file.path(dadesDir,"DatosMicroarrays.csv"),row.names = TRUE)
# write.csv2(DAMetilData, file=file.path(dadesDir,"DatosMetilacion.csv"),row.names = TRUE)
# DAExprData <- as.matrix(read.table(file=file.path(dadesDir,"DatosMicroarrays.csv"), header=TRU
# DAMetilData <- as.matrix(read.table(file=file.path(dadesDir,"DatosMetilacion.csv"), header=TRU
# DARNAseqData <- as.matrix(read.table(file=file.path(dadesDir,"DatosRNAseq.csv"), header=TRUE,
cat("DA Microarray data : ", dim(DAExprData), "\n")

## DA Microarray data :  11359 30

cat("DA Methylation data: ", dim(DAMetilData), "\n")

## DA Methylation data:  11359 30

# cat("DA RNASeq data     : ", dim(DARNAseqData), "\n")
```

- `geo` dataset is formed by data from expression microarrays (Illumina beadchips, `geoMicroarrays.csv`) and methylation array (Illumina 25kMethArray, `geoMetilacion.csv`) on 25 CRC samples. The data have been collected from the GEO database records: GSE25062 for the methylation data and GSE25070 for the expression data. The dataset has been created by taking the expression values available for 26 CRC tumors and matching with their corresponding methylation values.

```
geoExprData <-  as.matrix(read.table(file=file.path(dadesDir,"GEOExpData.csv"), header=TRUE, sep
geoMetilData <-  as.matrix(read.table(file=file.path(dadesDir,"GEOMethData.csv"), header=TRUE, s
cat("GEO Microarray data : ", dim(geoExprData), "\n")

## GEO Microarray data :  11191 25

cat("GEO Methylation data: ", dim(geoMetilData), "\n")

## GEO Methylation data:  11191 25
```

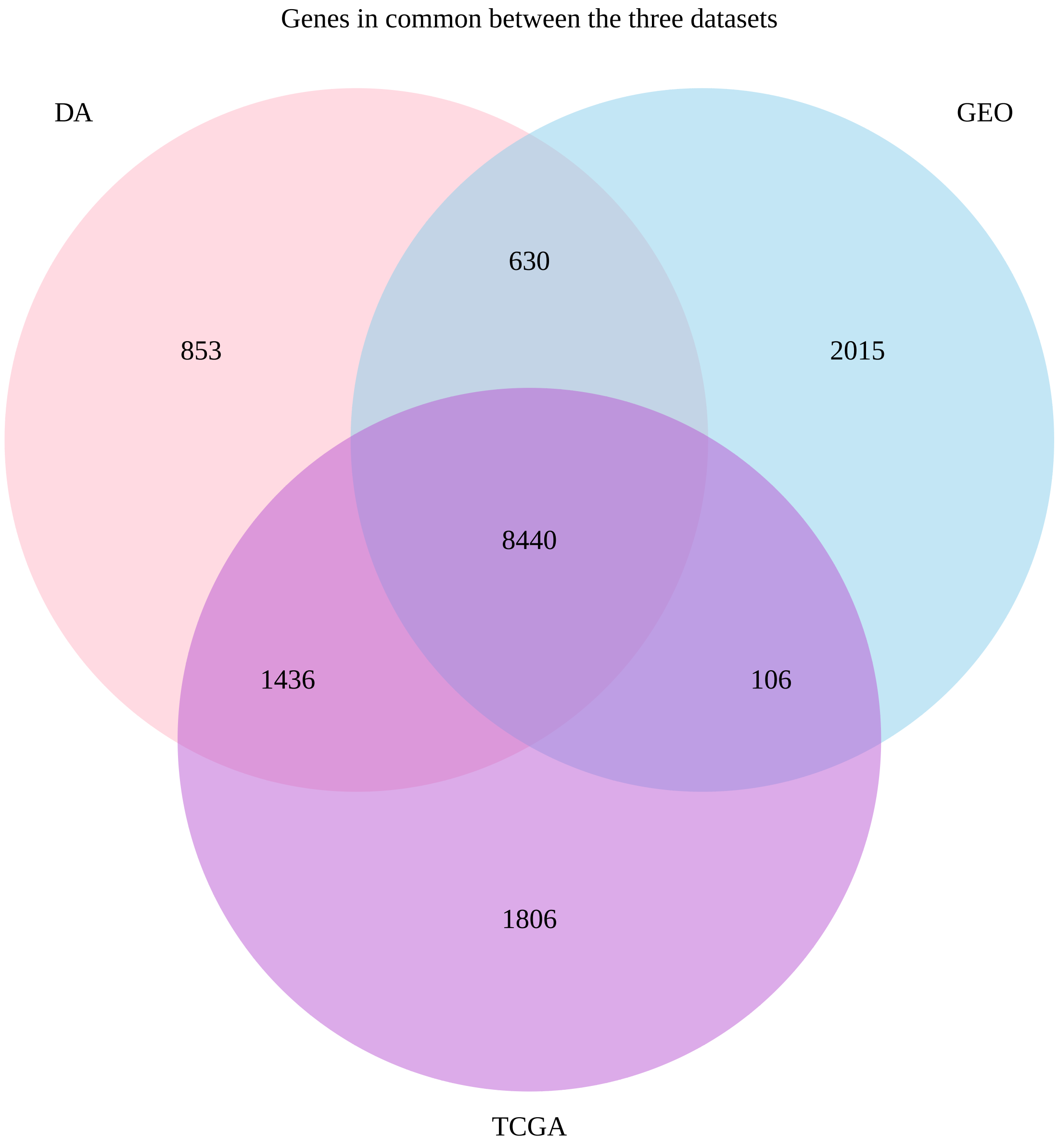
- `TCGA` dataset has been obtained from The Cancer Genome Atlas Database (TCGA) (Colon Adenocarcinoma (`COAD`) Nature 2012 dataset) and downloaded through the cBioportal website.

```
TCGAExprData <-  as.matrix(read.table(file="dades/TCGA-cBioPortal-Expressions.csv", header=TRUE,
TCGAMetilData <-  as.matrix(read.csv(file="dades/TCGA-cBioPortal-Methylations.csv", header=TRUE,
cat("TCGA Microarray data : ", dim(TCGAExprData), "\n")

## TCGA Microarray data :  11788 223

cat("TCGA Methylation data: ", dim(TCGAMetilData), "\n")

## TCGA Methylation data:  11788 223
```

```
inCommon<- length(intersect(rownames(DAExprData), rownames(geoExprData)))
inCommon2 <- length(intersect(rownames(DAExprData), rownames(TCGAExprData)))
inCommon3 <- length(intersect(rownames(geoExprData), rownames(TCGAExprData)))
```

There are 9070 genes in common between the DAX dataset and the GEO datasets. There are 9876 common genes between the DAX dataset and the TCGA dataset and 8546 common genes between the GEO dataset and the TCGA dataset.
This can be visualized using a Venn diagram

```
myVenn4 <- venn.diagram(x=list(DA=rownames(DAExprData),
                              GEO=rownames(geoExprData),
                              TCGA=rownames(TCGAExprData)),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue", "mediumorchid"),
                       main="Genes in common between the three datasets")
grid.newpage()
grid.draw(myVenn4)
```



2.3 TRUE and FALSE ’L-shaped’ genes

A problem with this study is that truly or falsely positive L-shaped genes are not known.
In the absence of this a list of *apparently truly positive* and *apparently truly negative* genes has been provided by the researcher.

```
# Genes True i False
trueLGeneDF <-read.table("dades/genesTrueLNEW.txt")
(trueLGeneNames <- as.character(trueLGeneDF[,1]))

##  [1] "ABCG2"      "ADH6"       "AKR1C4"     "BCL11B"     "CA9"
##  [6] "CCR6"       "CST7"       "CX3CL1"     "DAPP1"      "CYP27A1"
## [11] "ELAVL2"     "FAM84A"     "GREM1"      "HSD17B2"    "INHBB"
## [16] "KCNV1"      "LHFP"       "MEP1A"      "LRP2"       "MAGEE1"
## [21] "MEP1A"      "MYT1"       "NOX1"       "OAS2"       "PHYHIPL"
## [26] "POF1B"      "POPDC3"     "PRDM16"     "PRDM5"      "QPCT"
## [31] "RASGRF2"    "RAB6B"      "RASEF"      "RNF186"     "SOX2"
## [36] "SOSTDC1"    "SPON1"      "ST6GALNAC1" "STEAP4"     "STK33"
## [41] "TAPBPL"     "SYN2"       "THRB"       "TRAM1L1"    "WNK4"

falseLGeneDF <- read.table("dades/genesFalseLNEW.txt")
(falseLGeneNames <- as.character(falseLGeneDF[,1]))

##  [1] "ACOX2"     "ADA"       "AKR1B1"    "ALDH1A3"   "AMT"
##  [6] "ANXA3"     "ARHGAP4"   "ARL14"     "ATP6AP2"   "AUTS2"
## [11] "BHLHB9"    "BMP7"      "C19orf33"  "C1QTNF6"   "CAB39L"
## [16] "CDH17"     "CDX1"      "CFTR"      "CIDEB"     "CMTM3"
## [21] "DNAJA4"    "DUSP9"     "ELF3"      "ELMO3"     "FBP1"
## [26] "FGD4"      "FKBP10"    "FOXC1"     "FUCA2"     "GNG4"
## [31] "GPRC5A"    "GRAMD3"    "H1F0"      "HIST1H2BH" "HNMT"
## [36] "HOOK1"     "HOOK3"     "HOXB3"     "HOXB2"     "HOXB5"
## [41] "HS3ST1"    "LCMT2"     "LAMA3"     "LDHB"

trueLExpr <- DAExprData[rownames(DAExprData) %in% trueLGeneNames ,]
falseLExpr <- DAExprData[rownames(DAExprData) %in% falseLGeneNames ,]
trueLMet <- DAMetilData[rownames(DAMetilData) %in% trueLGeneNames ,]
falseLMet <- DAMetilData[rownames(DAMetilData) %in% falseLGeneNames ,]
DATrueFalseExpr <- as.matrix(rbind(trueLExpr, falseLExpr))
DATrueFalseMet <- as.matrix(rbind(trueLMet, falseLMet))
```

2.4 Additional data checks

The data for these analyses must have a common structure: **Each pair of matrices (Expression-Methylation) must have the same rome and column names**, that is both datasets must contain information for the same genes and same samples at their corresponding positions.
This can be checked using a simple function such as `checkData` available in the package.

```
try(if(!checkPairing(DAExprData, DAMetilData)) stop("Row names and/or column names do not match"))
try(if(!checkPairing(geoExprData, geoMetilData)) stop("Row names and/or column names do not match"))
try(if(!checkPairing(TCGAExprData, TCGAMetilData)) stop("Row names and/or column names do not match"
try(if(!checkPairing(trueLExpr, trueLMet)) stop("Row names and/or column names do not match"))
try(if(!checkPairing(falseLExpr, falseLMet)) stop("Row names and/or column names do not match"))
try(if(!checkPairing(DATrueFalseExpr, DATrueFalseMet)) stop("Row names and/or column names do not ma
```

3 Selection of L-shaped genes

3.1 Methods

After a long process of trial and error we consider that there are three “best” approaches for selecting L-shaped genes. The methods are extensively described in another document so only a brief description is provided below:

1. **Naive**: A gene is called L-shaped if Methylation and expression are significantly negatively correlated.
2. **CMI**: A gene is called L-Shaped if the *Conditional Mutual Information* (CMI) of the expression and methylation values computed at different points between 0 and 1 reaches a minimum which is small enough according predefined thresholds. This minimum can be considered to be the cutoff for methylation.
3. **Heuristic**: A gene is called L-shaped if the majority of the points of the scatterplot stay on the left and lower cells of a 3×3 grid. How many values can deviate from this "majority" in each cell can be tuned by the user.

3.2 Selecting L-shaped genes with the Naive method

A gene is called L-shaped if methylation and expression are significantly negatively correlated. If the number of samples is big, as in the TCGA dataset, many genes will be significantly correlated. In these cases it may be worth to set an additional threshold such as $r < -0.XXX$ where $-0.XXX$ is a threshold to be set by the user. A trial and error process suggests that a -0.5 cutoff may be appropriate

```
#calcAgain <- TRUE
#if (!file.exists("results/naiveSelections.Rda") || calcAgain){
naiveDA <- naiveSelection (DAExprData, DAMetilData, pValCutoff=0.25, rCutoff=-0.5, type="Spearman",ac
selectedNaiveDA <-naiveDA[naiveDA$SigNegCorr,]
```

```
    naiveGEO <- naiveSelection (geoExprData, geoMetilData, pValCutoff=0.25,  rCutoff=-0.5, type="Spea
    selectedNaiveGEO <-naiveGEO[naiveGEO$SigNegCorr,]
```

```
naiveTCGA <- naiveSelection (TCGAExprData, TCGAMetilData, pValCutoff=0.25,  rCutoff=-0.5, type="Spear
selectedNaiveTCGA <-naiveTCGA[naiveTCGA$SigNegCorr,]
```

```
save(naiveDA, naiveGEO, naiveTCGA,
         selectedNaiveDA, selectedNaiveGEO, selectedNaiveTCGA,
         file="results/naiveSelections.Rda")

selectedGenesNaiveDA <- rownames(selectedNaiveDA)
selectedGenesNaiveGEO <- rownames(selectedNaiveGEO)
selectedGenesNaiveTCGA <- rownames(selectedNaiveTCGA)
save(selectedGenesNaiveDA,
         selectedGenesNaiveGEO,
         selectedGenesNaiveTCGA,
         file="results/naiveSelectionsGeneNames.Rda")
  # }else{
  #  load("results/naiveSelections.Rda")
  #  load("results/naiveSelectionsGeneNames.Rda")
  #}
```

```r
numNaiveDA <- sum(naiveDA$SigNegCorr)
numNaiveGEO <- sum(naiveGEO$SigNegCorr)
numNaiveTCGA <- sum(naiveTCGA$SigNegCorr)
```

The distribution of the correlation coefficients of the selected genes is shown in figure 1
The number of genes selected using the Naive method on the DA, TCGA and GEO datasets are respectively:

```r
cat("numNaive (DA): ", numNaiveDA,"\n")

## numNaive (DA):  456

cat("numNaive (GEO): ", numNaiveGEO,"\n")

## numNaive (GEO):  72

cat("numNaive (TCGA): ", numNaiveTCGA,"\n")

## numNaive (TCGA):  190
```

This can be visualized using Venn Diagrams:

```r
myVenn3<- venn.diagram(x=list(naiveDA=rownames(selectedNaiveDA),
                              naiveGEO=rownames(selectedNaiveGEO),
                              naiveTCGA=rownames(selectedNaiveTCGA)),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue", "mediumorchid"),
                       main="Genes in common between the three -Naive selected- gene lists")
grid.newpage()
grid.draw(myVenn3)
```

```
opt<-par(mfrow=c(2,2))
hist(selectedNaiveDA[,1], xlim=c(-1,0), main="Significant correlations in DA dataset")
hist(selectedNaiveGEO[,1], xlim=c(-1,0), main="Significant correlations in GEO dataset")
hist(selectedNaiveTCGA[,1], xlim=c(-1,0), main="Significant correlations in TCGA dataset")
par(opt)
```

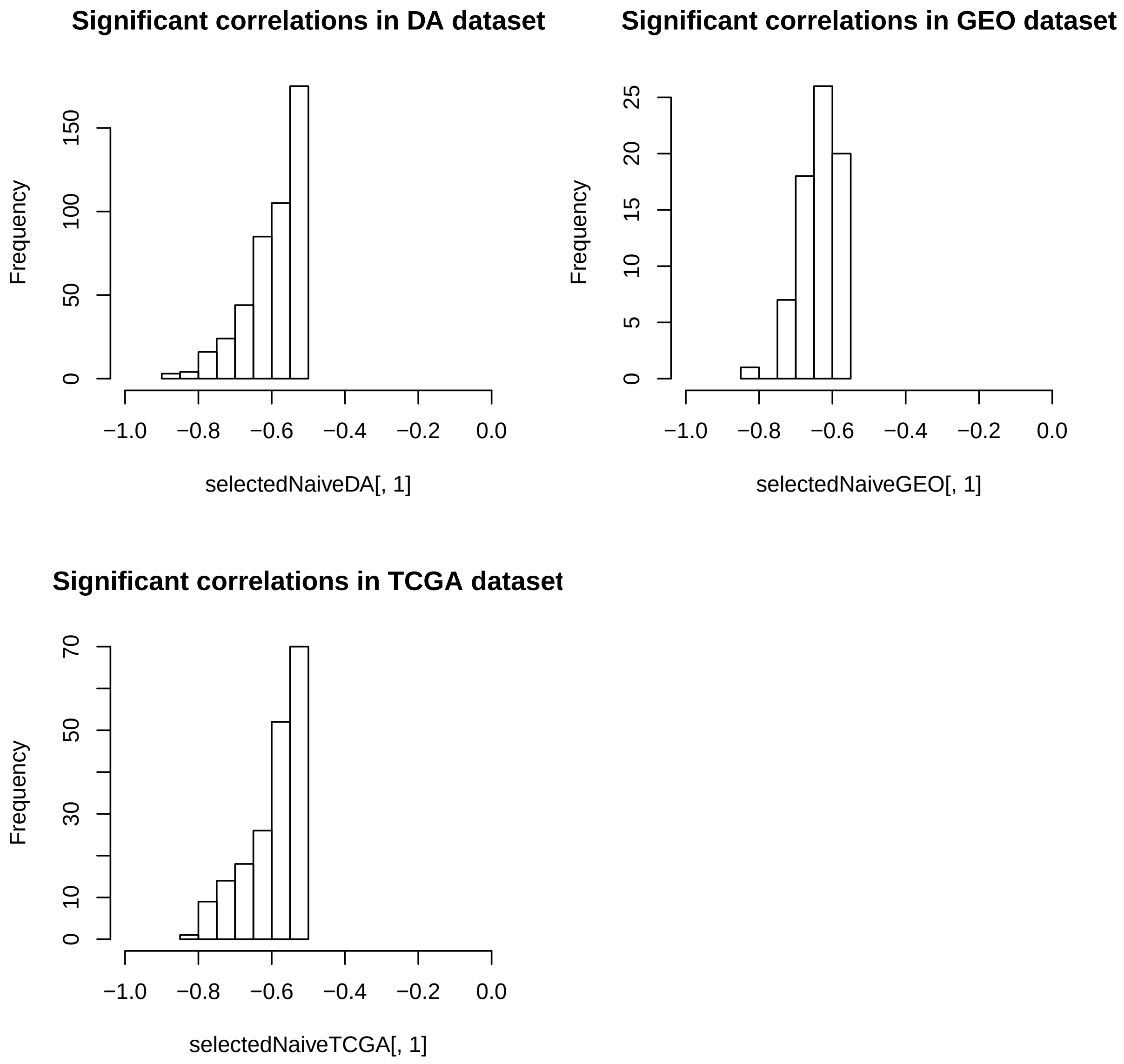
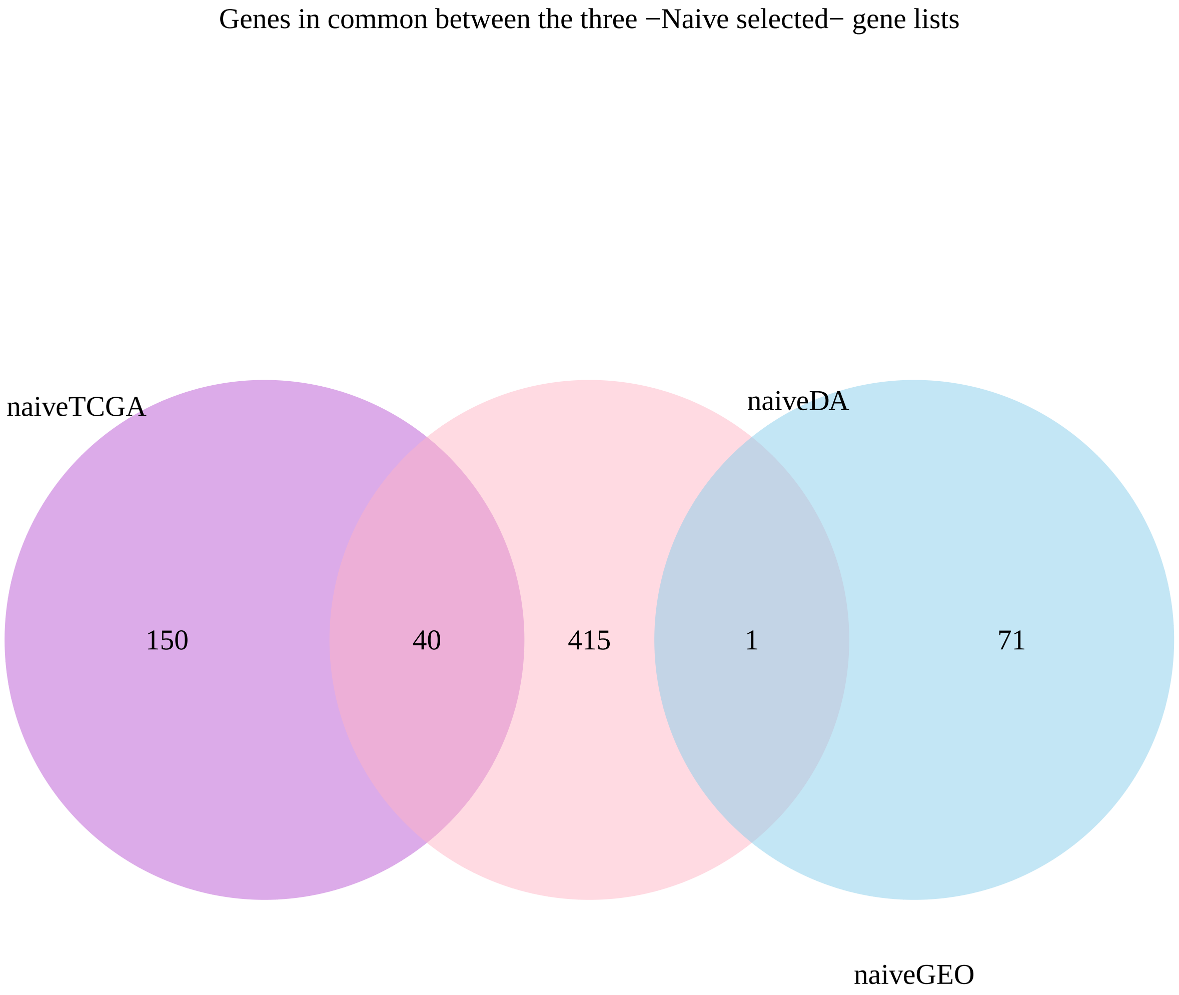
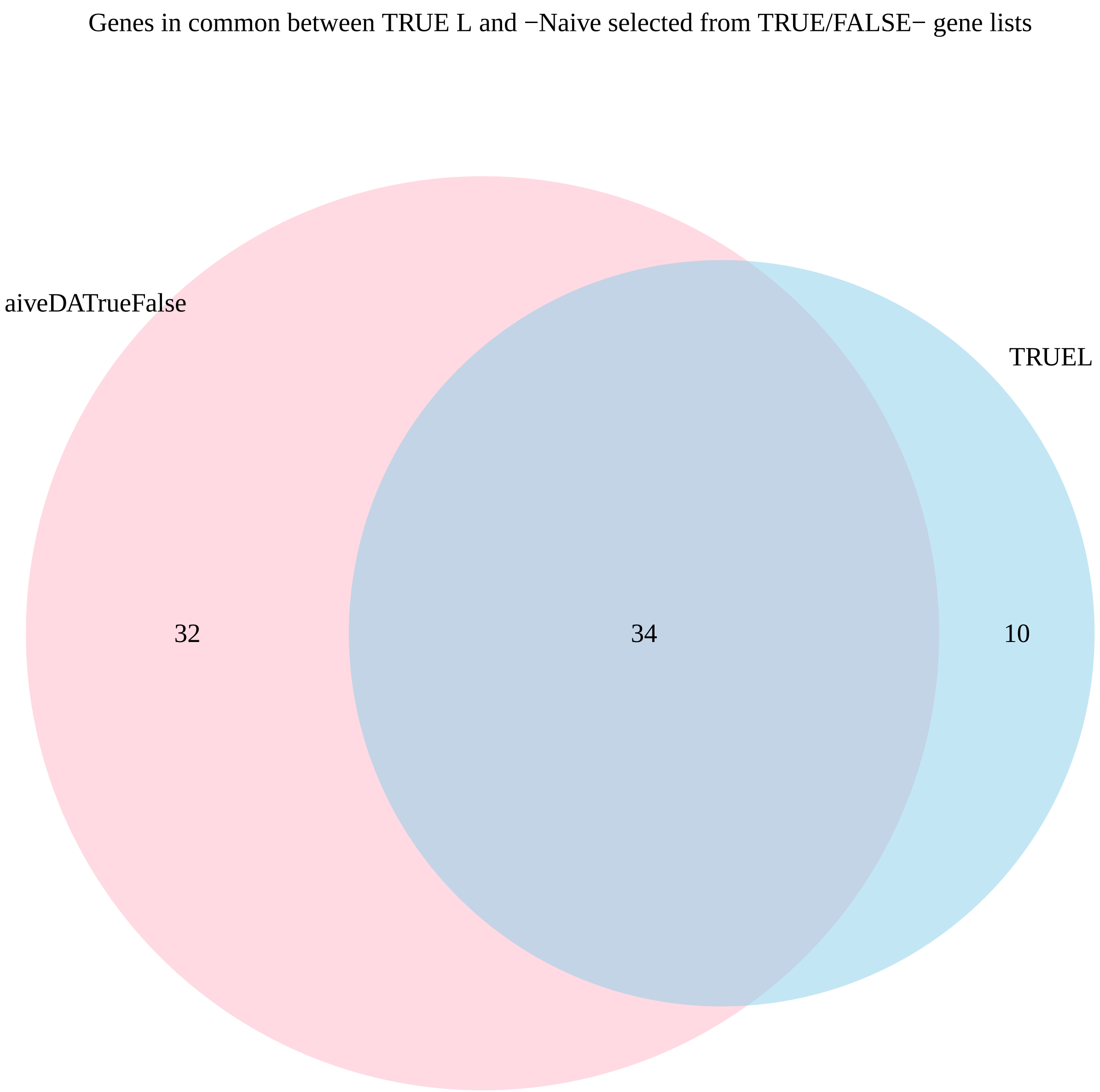


Figure 1: Distribution of the correlation coefficients of the selected genes. Depending on the ‘pvalue‘ and the ‘r‘ cutoff more or less genes will be retained

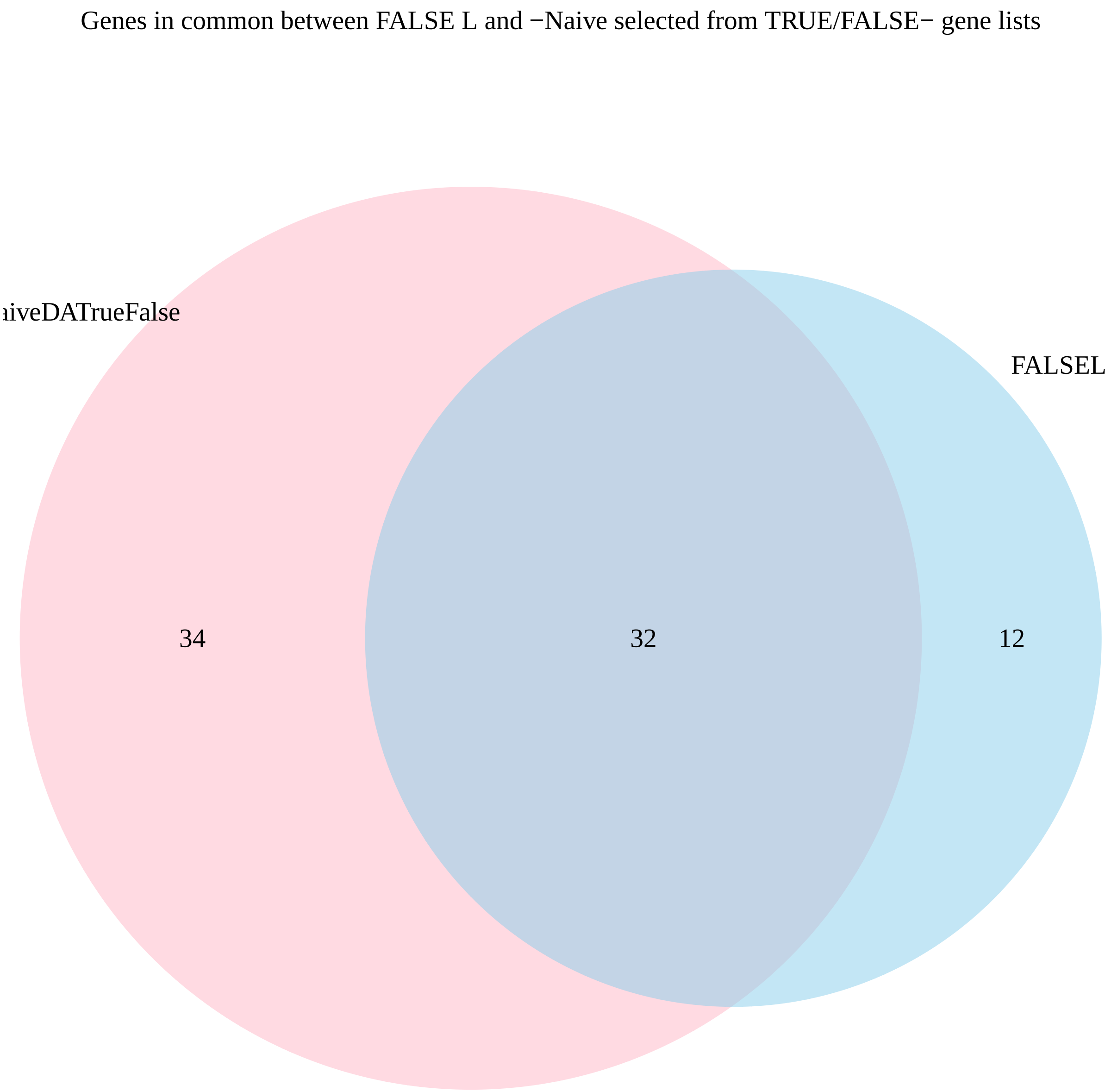


3.2.1 Naive selection on TRUE and FALSE candidates

```
naiveDATrueFalse <- naiveSelection (as.matrix(DATrueFalseExpr), as.matrix(DATrueFalseMet), pValCutoff
selectedNaiveDATrueFalse <-naiveDATrueFalse[naiveDATrueFalse$SigNegCorr,]
myVenn2<- venn.diagram(x=list(naiveDATrueFalse=rownames(selectedNaiveDATrueFalse), TRUEL=trueLGeneNam
grid.newpage()
grid.draw(myVenn2)
```



```
myVenn2<- venn.diagram(x=list(naiveDATrueFalse=rownames(selectedNaiveDATrueFalse),
                              FALSEL=falseLGeneNames),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue"),
                       main="Genes in common between FALSE L and -Naive selected from TRUE/FALSE- ge
grid.newpage()
grid.draw(myVenn2)
```



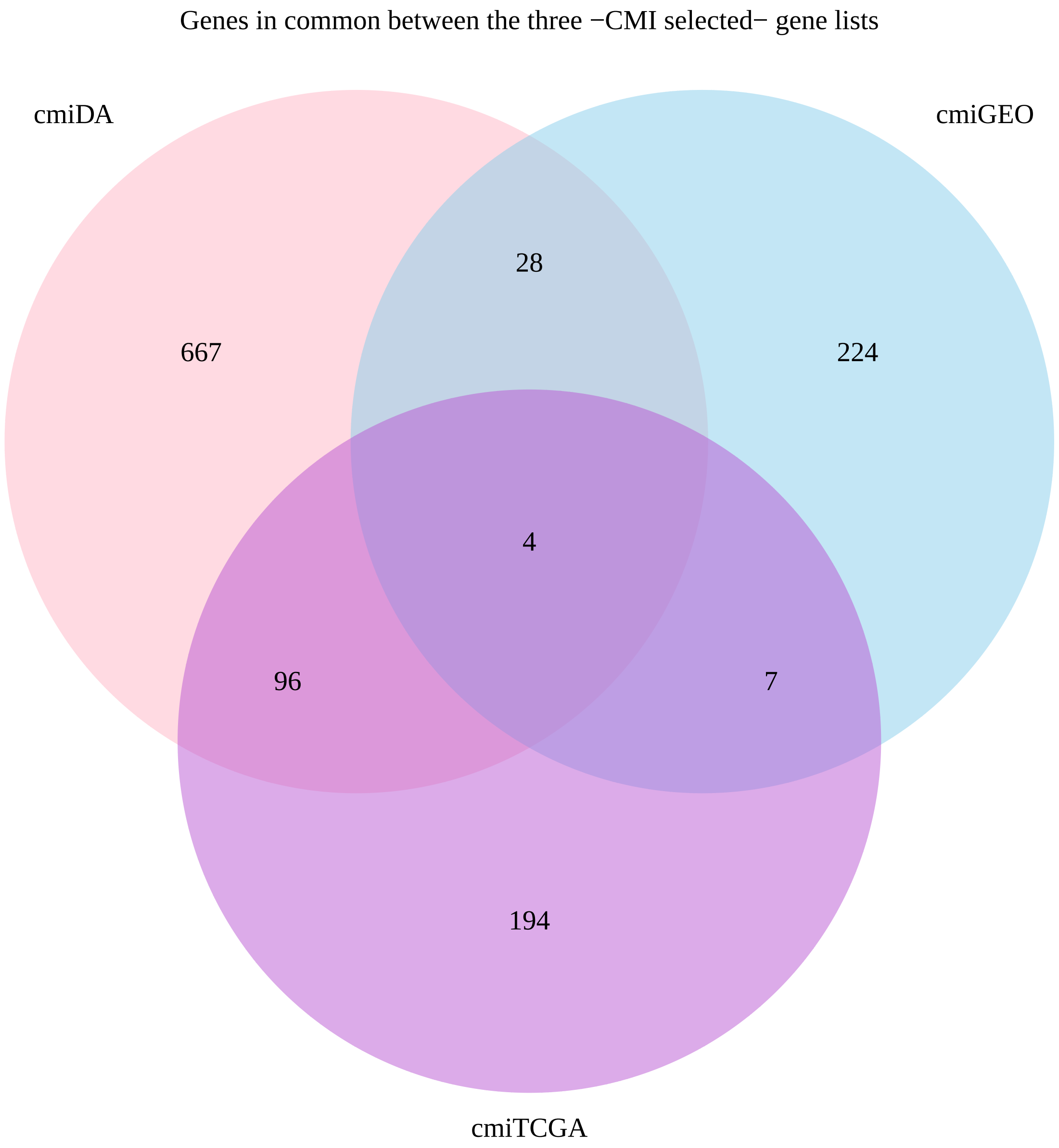
It seems that both TRUE and FALSE L candidates are mostly negatively correlated.

3.3 Selecting L-shaped genes with the CMI method

```
calcAgainCMI <- FALSE
if (!file.exists("results/cmiSelections.Rda") || calcAgainCMI){
  system.time(cmiDA <- cmiSelection (methData = DAMetilData, exprData = DAExprData ))
  selectedCmiDA <-cmiDA[cmiDA[,"meth_regulated"],]
  system.time(cmiGEO <- cmiSelection (methData = geoMetilData, exprData = geoExprData))
  selectedCmiGEO <-cmiGEO[cmiGEO[,"meth_regulated"],]
  system.time(cmiTCGA <- cmiSelection (methData = TCGAMetilData, exprData = TCGAExprData))
  selectedCmiTCGA <-cmiTCGA[cmiTCGA[,"meth_regulated"],]
  save(cmiDA, cmiGEO, cmiTCGA,
         selectedCmiDA, selectedCmiGEO, selectedCmiTCGA,
         file="results/cmiSelections.Rda")
  selectedGenesCmiDA <- rownames(selectedCmiDA)
  selectedGenesCmiGEO <- rownames(selectedCmiGEO)
  selectedGenesCmiTCGA <- rownames(selectedCmiTCGA)
  save(selectedGenesCmiDA, selectedGenesCmiGEO, selectedGenesCmiTCGA,
         file="results/cmiSelectionsGeneNames.Rda")
}else{
    load("results/cmiSelections.Rda")
    load("results/cmiSelectionsGeneNames.Rda")
}
```

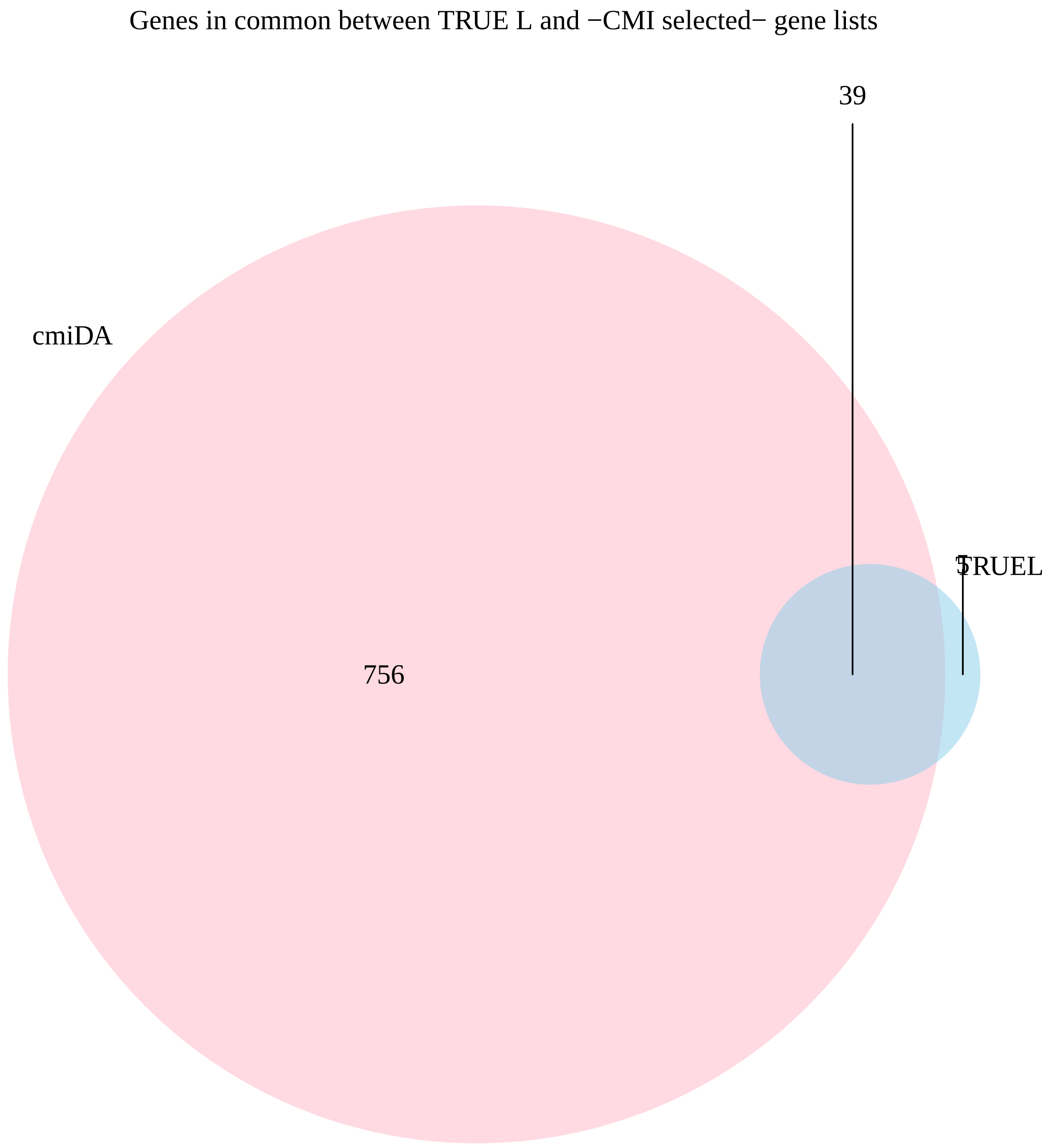
The number of genes selected using the CMI method on the DA, TCGA and GEO datasets are respectively 795, 263 and 301.
This can be visualized using Venn Diagrams:

```
myVenn3<- venn.diagram(x=list(cmiDA=rownames(selectedCmiDA),
                              cmiGEO=rownames(selectedCmiGEO),
                              cmiTCGA=rownames(selectedCmiTCGA)),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue", "mediumorchid"),
                       main="Genes in common between the three -CMI selected- gene lists")
grid.newpage()
grid.draw(myVenn3)
```

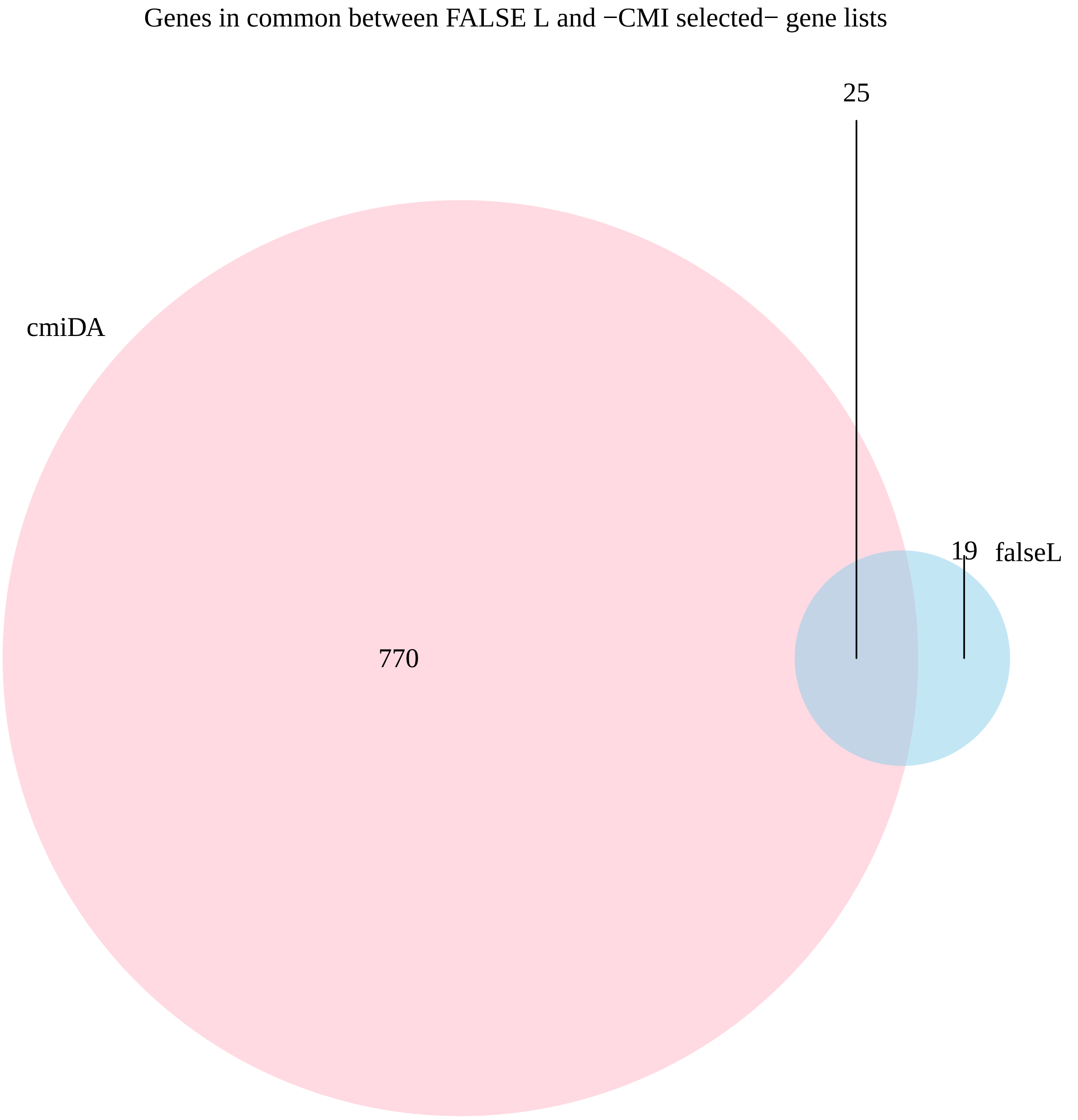


3.3.1 CMI selection on TRUE and FALSE candidates

```
cmiDATrueFalse <- cmiSelection (DATrueFalseExpr, DATrueFalseMet)
selectedCmiDATrueFalse <-cmiDATrueFalse[cmiDATrueFalse[,"meth_regulated"],]

myVenn2<- venn.diagram(x=list(cmiDA=rownames(selectedCmiDA),
                              TRUEL=trueLGeneNames),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue"),
                       main="Genes in common between TRUE L and -CMI selected- gene lists")
grid.newpage()
grid.draw(myVenn2)
```



```
myVenn2<- venn.diagram(x=list(cmiDA=rownames(selectedCmiDA),
                              falseL=falseLGeneNames),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue"),
                       main="Genes in common between FALSE L and -CMI selected- gene lists")
grid.newpage()
grid.draw(myVenn2)
```



In this case it seems that the CMI method has selected more genes from the TRUE than from the FALSE example list.

3.4 Selecting L-shaped genes with the *heuristic* method

The heuristic method is intended to select L-shaped scatterplots by overimposing a grid on it and defining certain regions which have to (or don't have to) contain a minimum (or maximum) percentage of points if the scatterplot is to be called L–shaped.
Besides this the method computes a scoring in such a way that scores in "good regions" score positively and points in "bad regions" score negatively. ***An appropriate setting of scores and weights should yield positive scores for L-shaped scatterplots and negative scores for those that are not.***
One of the main interests of the approach presented here is the possibility to "tune" the selection process by changing the scoring parameters.

```
(reqPercentages  <- matrix (c(10, 20, 1, 5, 40, 20, 0, 5, 10), nrow=3, byrow=TRUE))
```

10	20	1
5	40	20
0	5	10

```
(maxminCounts <- toReqMat(dim(DAMetilData)[2], reqPercentages)) # Informative. NOt used in calculati
```

3	6	0
2	12	6
0	2	3

```
(reqPercentages4TCGA <- matrix (c(4, 20, 5, 1, 40, 20, 0, 1, 4), nrow=3, byrow=TRUE))
```

4	20	5
1	40	20
0	1	4

```
(maxminCounts4TCGA <- toReqMat(dim(TCGAMetilData)[2], reqPercentages4TCGA))
```

9	45	11
2	89	45
0	2	9

```
(theWeightMifL=matrix (c(2,-2,-25,1,0,-2,1,1,2), nrow=3, byrow=TRUE))
```

2	-2	-25
1	0	-2
1	1	2

```
(theWeightMifNonL=matrix (c(0,-2,-25,0,0,-2,0,0,0), nrow=3, byrow=TRUE))
```

0	-2	-25
0	0	-2
0	0	0

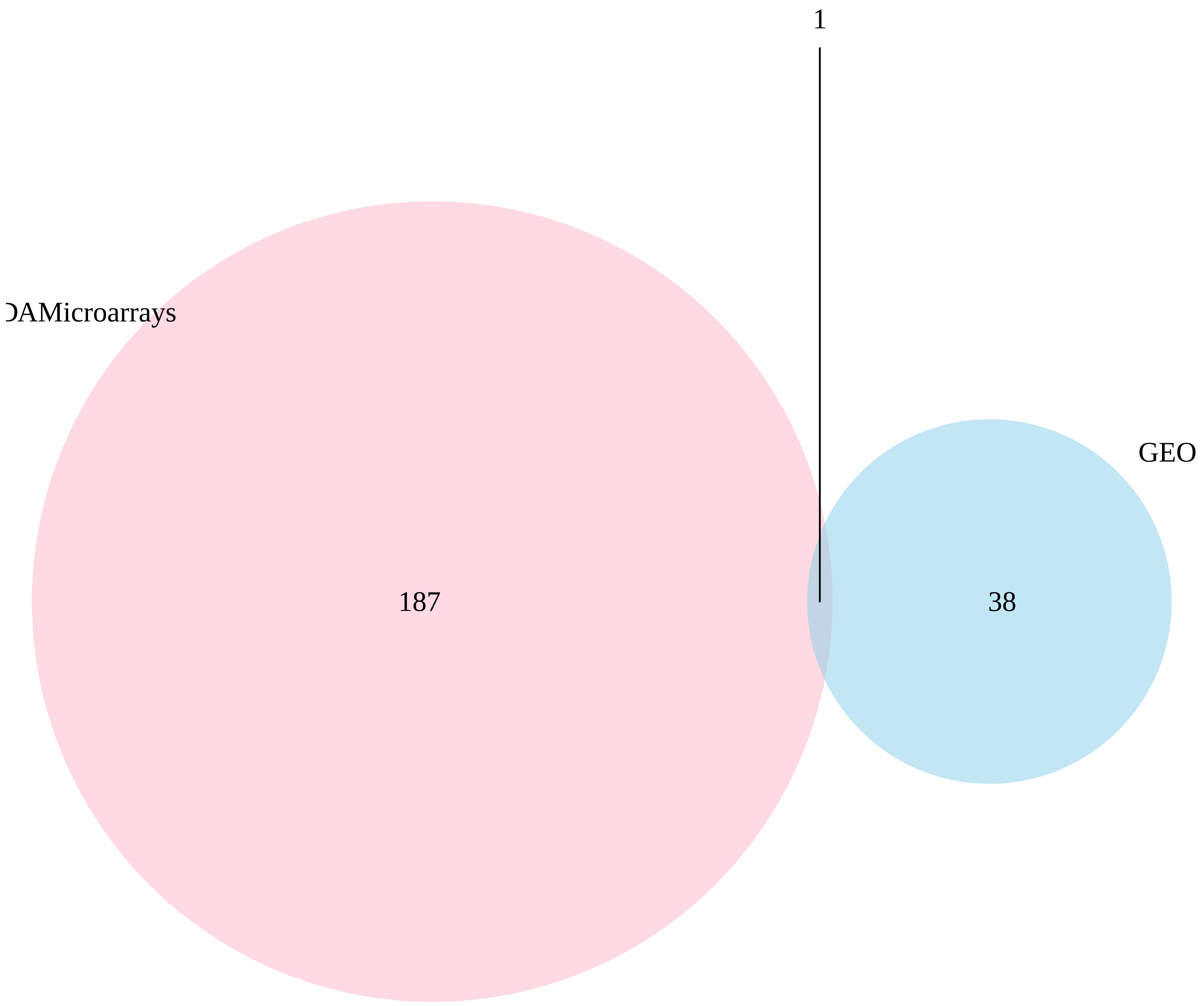
```
calcAgainHeuristic <- TRUE # Com donava error l'he calculat pas a pas i he gravat els resultats
if (!file.exists("results/heuristicSelections.Rda") || calcAgainHeuristic){
  # DA dataset
  sampleSize <- dim(DAMetilData)[2]
  numGenes <-   dim(DAMetilData)[1]

  messageTitle("Scoring ALL genes in the DA (microarrays) dataset")

  scoresDA1 <- scoreGenesMat (mets=DAMetilData,
                                                                      expres=DAExprData,
                            aReqPercentsMat=reqPercentages,
                            aWeightMifL=theWeightMifL,
                            aWeightMifNonL=theWeightMifNonL )
  cat("Number of scatterplots scored  : ", dim(scoresDA1)[1],"\n")
  cat("Number of L-shape scatterplots : ", sum(scoresDA1[,1]),"\n")
  head(scoresDA1)
  table(scoresDA1[,1])
  selectedHeuristicDA <- scoresDA1[scoresDA1$logicSc,]

  #GEO dataset
  messageTitle("Scoring ALL genes in the GEO dataset")
  sampleSize <- dim(geoMetilData)[2]
  scoresGEO <- scoreGenesMat (mets=geoMetilData,
                            expres=geoExprData,
                            aReqPercentsMat=reqPercentages,
                            aWeightMifL=theWeightMifL,
                            aWeightMifNonL=theWeightMifNonL )
  cat("Number of scatterplots scored  : ", dim(scoresGEO)[1], "\n")
  cat("Number of L-shape scatterplots : ", sum(scoresGEO[,1]), "\n")
  table(scoresGEO[,1])
  selectedHeuristicGEO <- scoresGEO[scoresGEO$logicSc,]

  #TCGA dataset
  (numGenes <-   dim(TCGAMetilData)[1])
  (sampleSize <- dim(TCGAMetilData)[2])

  reqPercentages <- matrix (c(2, 20, 5, 1, 40, 20, 0, 1, 2), nrow=3, byrow=TRUE)
  (maxminCounts <- toReqMat(sampleSize, reqPercentages))

  (theWeightMifL=matrix (c(2,-2,-sampleSize/5,1,0,-2,1,1,2), nrow=3, byrow=TRUE))
  (theWeightMifNonL=matrix (c(0,-2,-sampleSize/5,0,0,-2,0,0,0), nrow=3, byrow=TRUE))

  messageTitle("Scoring ALL genes in the TCGA (microarrays) dataset")
  scoresTCGA <- scoreGenesMat (mets=TCGAMetilData,
                                                                      expres=TCGAExprData,
                                                                       x1=1/3, x2=2/3,
                            aReqPercentsMat=reqPercentages,
                            aWeightMifL=theWeightMifL,
                            aWeightMifNonL=theWeightMifNonL )
  cat("Number of scatterplots scored  : ", dim(scoresTCGA)[1],"\n")
  cat("Number of L-shape scatterplots : ", sum(scoresTCGA[,1]),"\n")
  head(scoresTCGA)
  table(scoresTCGA[,1])
  selectedHeuristicTCGA <- scoresTCGA[scoresTCGA$logicSc,]

  save(scoresDA1, scoresGEO, scoresTCGA,
       selectedHeuristicDA, selectedHeuristicGEO, selectedHeuristicTCGA,
         file="results/heuristicSelections.Rda")
  selectedGenesHeuristicDA <- rownames(selectedHeuristicDA)
  selectedGenesHeuristicGEO <- rownames(selectedHeuristicGEO)
  selectedGenesHeuristicTCGA <- rownames(selectedHeuristicTCGA)
  save(selectedGenesHeuristicDA,selectedGenesHeuristicGEO, selectedGenesHeuristicTCGA, file="results
}else{
    load("results/heuristicSelections.Rda")
   load("results/heuristicSelectionsGeneNames.Rda")
}
```

```
## Scoring ALL genes in the DA (microarrays) dataset
## --------------------------------------------------
## Number of scatterplots scored  :  11359
## Number of L-shape scatterplots :  188
## Scoring ALL genes in the GEO dataset
## -------------------------------------
## Number of scatterplots scored  :  11191
## Number of L-shape scatterplots :  39
## Scoring ALL genes in the TCGA (microarrays) dataset
## ----------------------------------------------------
## Number of scatterplots scored  :  11788
## Number of L-shape scatterplots :  735
```

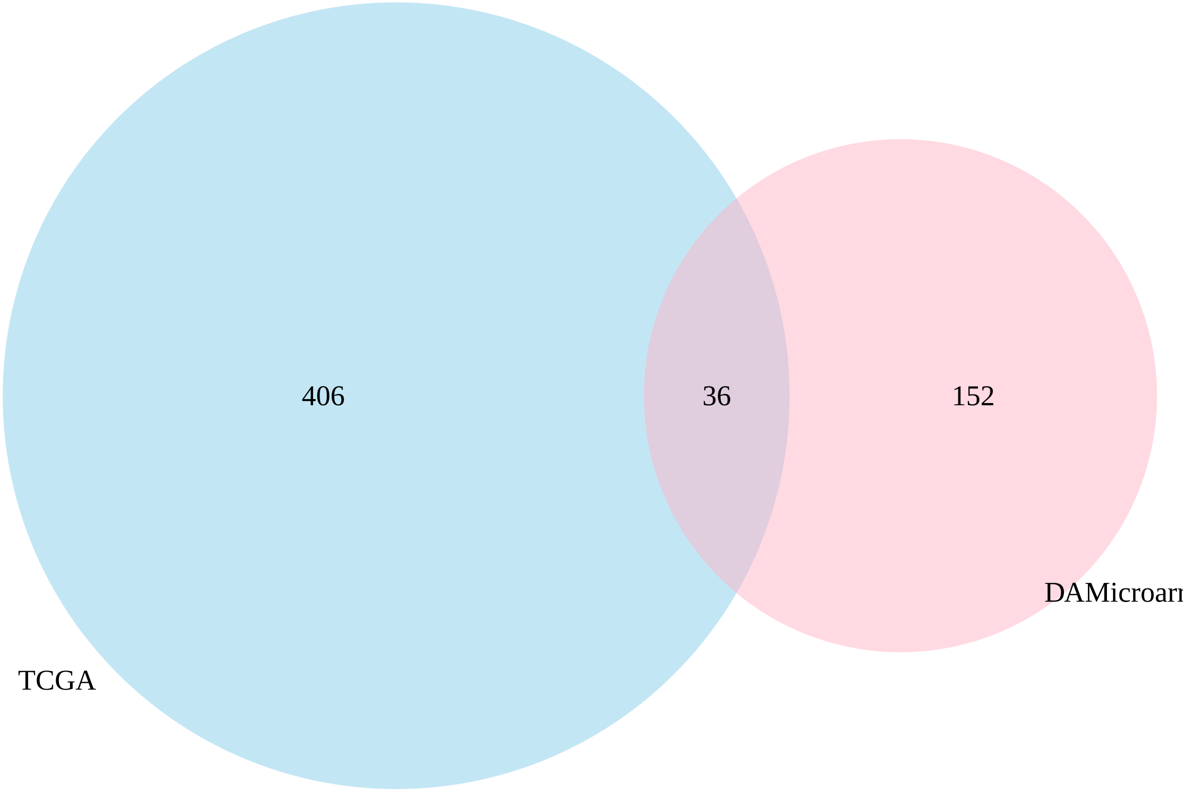
We may select L genes and plot only these. The resulting plots are available in files

- `DAExprLGenesScores.pdf`
- `GEOLGenesScores.pdf`
- `TCGALGenesScores.pdf`

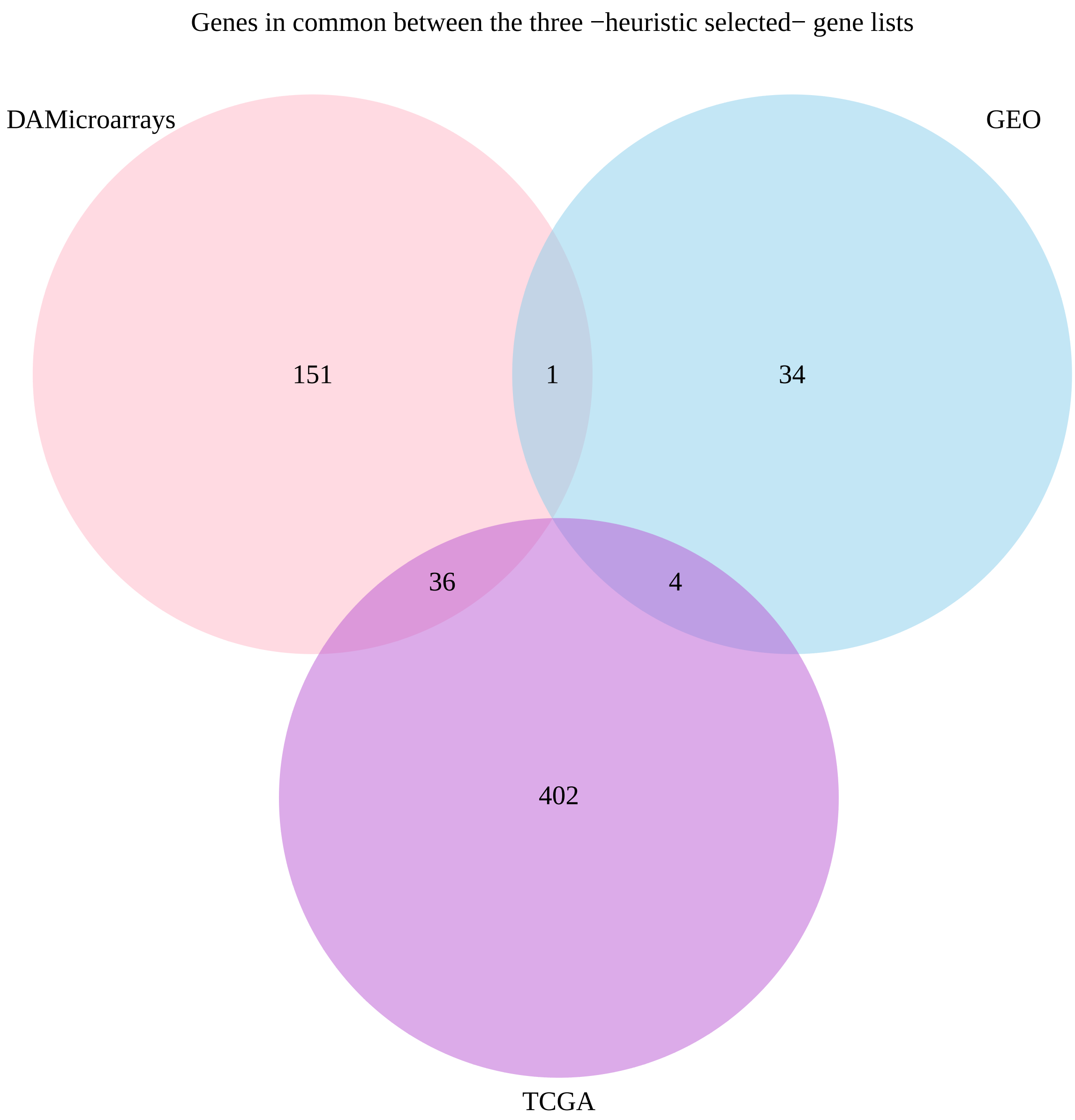
```
LgenesDAExpr <- DAExprData[scoresDA1[,"logicSc"],]
dim(LgenesDAExpr)

## [1] 188  30

geneListLDAExpr <- rownames(DAExprData[scoresDA1[,"logicSc"],])
plotGenesMat (mets=DAMetilData[geneListLDAExpr,],
              expres=DAExprData[geneListLDAExpr,],
              fileName ="results/DAExprLGenesScores.pdf",
              text4Title = scoresDA1[geneListLDAExpr,"numericSc"])

## pdf
##   2

LgenesGEOExpr <- geoExprData[scoresGEO[,"logicSc"],]
dim(LgenesGEOExpr)

## [1] 39 25

geneListLGEOExpr <- rownames(geoExprData[scoresGEO[,"logicSc"],])
plotGenesMat (mets=geoMetilData[geneListLGEOExpr,],
              expres=geoExprData[geneListLGEOExpr,],
              fileName ="results/geoExprLGenesScores.pdf",
              text4Title = scoresGEO[geneListLGEOExpr,"numericSc"])

## pdf
##   2

myVenn2<- venn.diagram(x=list(DAMicroarrays=geneListLDAExpr,
                              GEO=geneListLGEOExpr),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue"))
grid.newpage()
grid.draw(myVenn2)
```



```
LgenesTCGAExpr <- TCGAExprData[scoresTCGA$numericSc >100, ]
dim(LgenesTCGAExpr)

## [1] 442 223

geneListLTCGAExpr <- rownames(TCGAExprData[scoresTCGA$numericSc >100, ])


myVenn3<- venn.diagram(x=list(DAMicroarrays=geneListLDAExpr,
                              TCGA=geneListLTCGAExpr),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue"))
grid.newpage()
grid.draw(myVenn3)
```



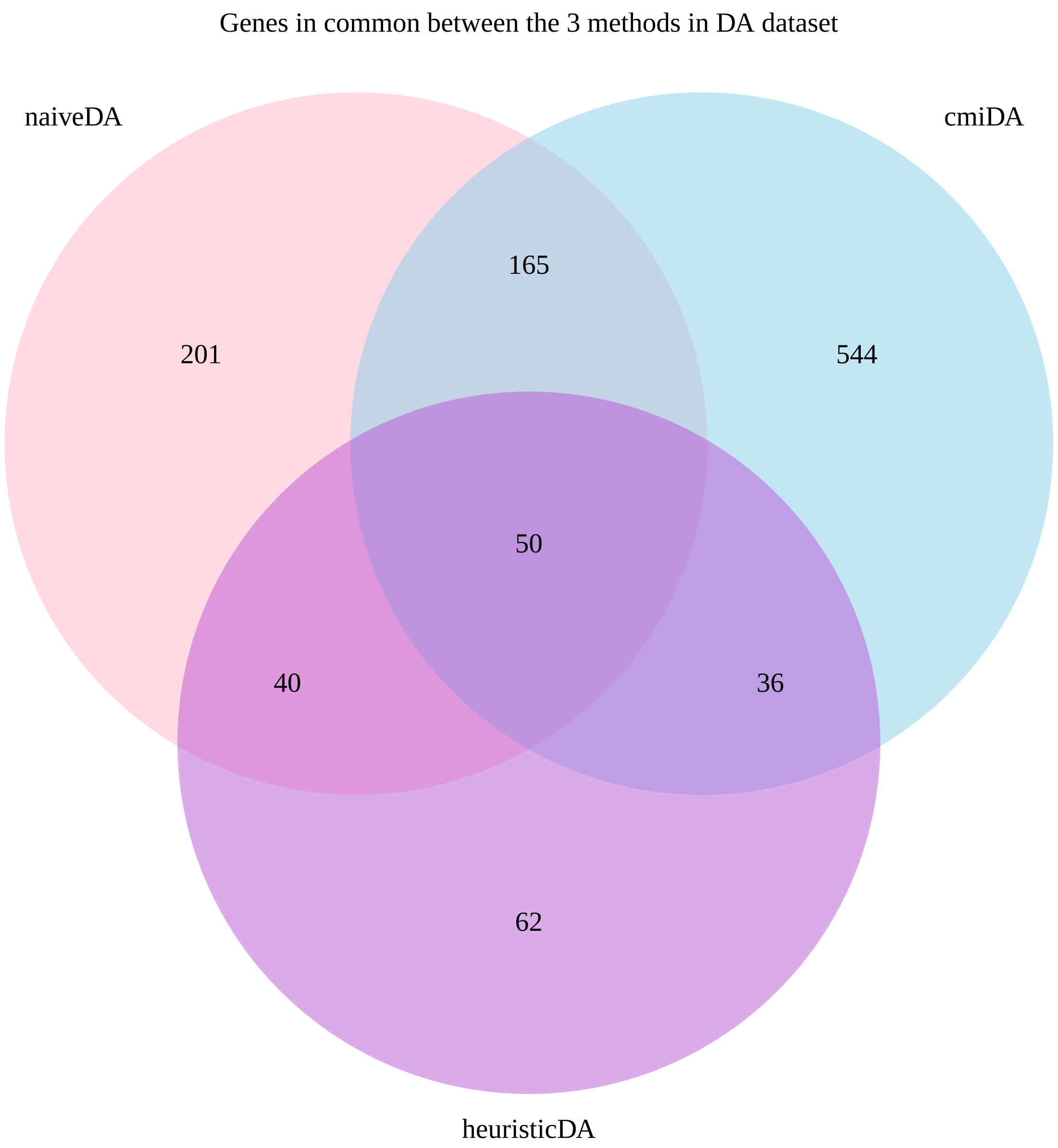
```
myVenn4H<- venn.diagram(x=list(DAMicroarrays=geneListLDAExpr,
                              GEO=geneListLGEOExpr,
                              TCGA=geneListLTCGAExpr),
                              filename=NULL, lty = "blank",
                              fill=c("pink1", "skyblue", "mediumorchid"),
                           main="Genes in common between the three -heuristic selected- gene lists")
grid.newpage()
grid.draw(myVenn4H)
```



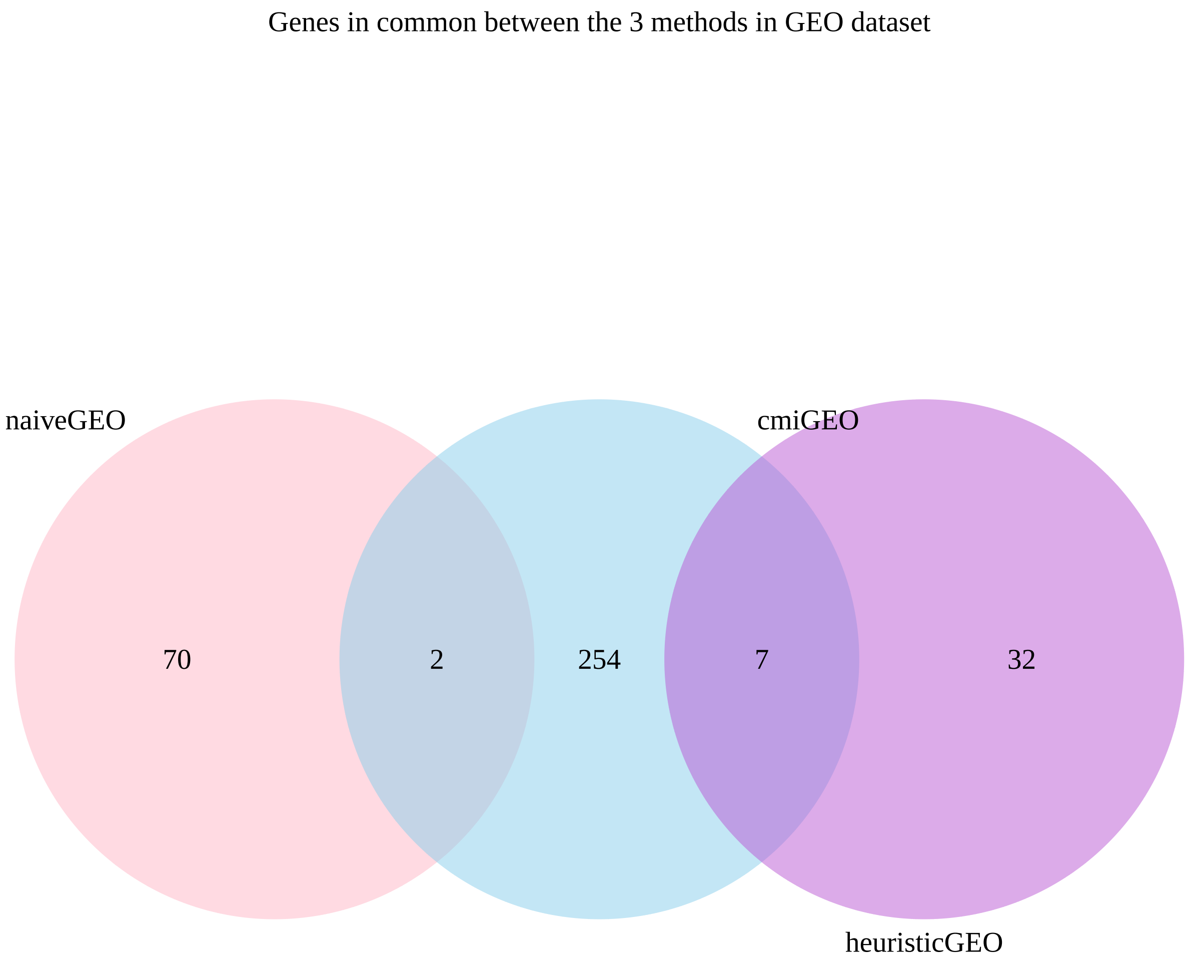
4 Comparison between the selected lists

What is the level of concordance between the three methods? This obviously may depend on the parameters of each method but for the values set her we obtain the following coincidences.

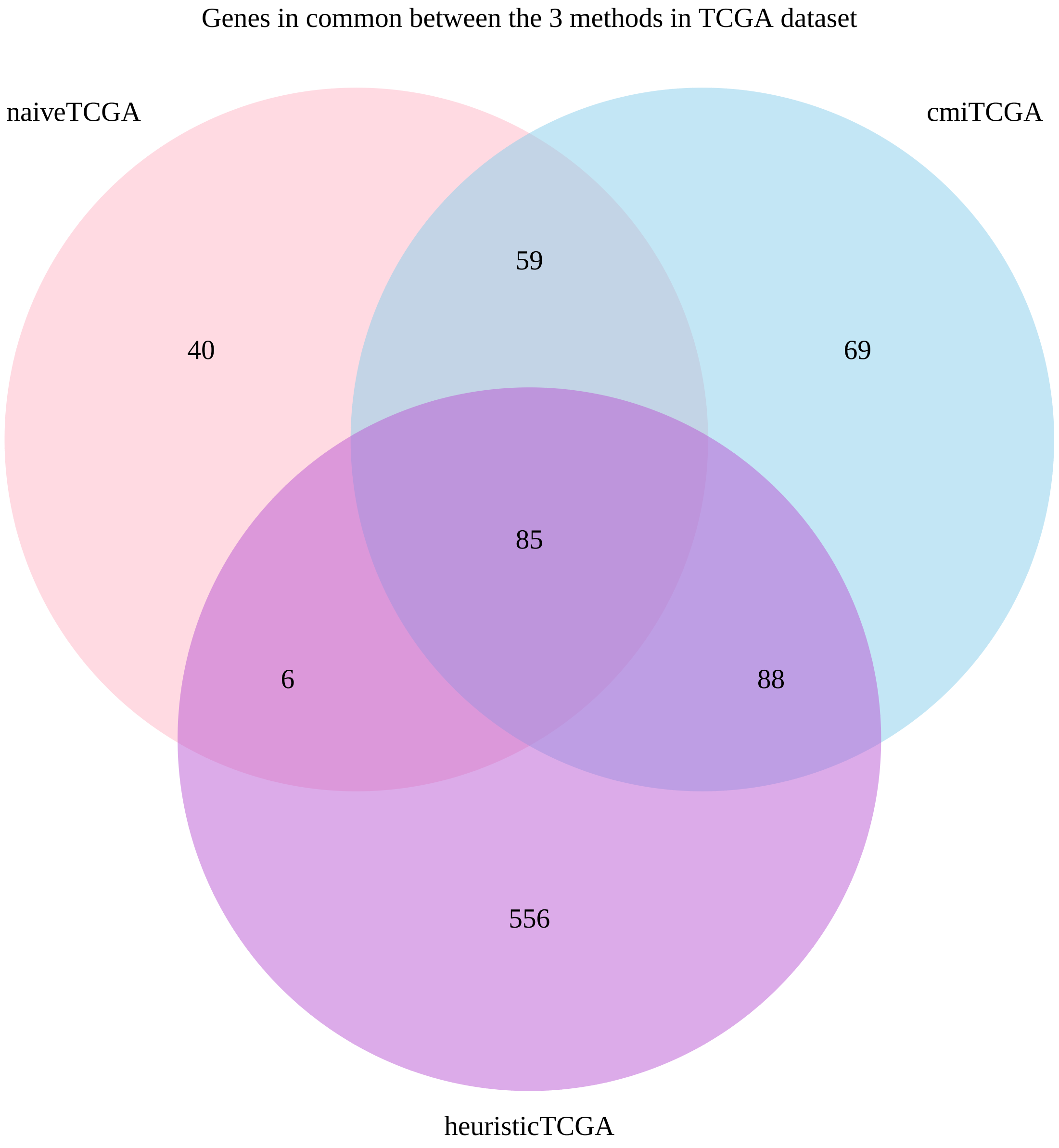
```
myVenn3DA<- venn.diagram(x=list(naiveDA=rownames(selectedNaiveDA),
                                cmiDA=rownames(selectedCmiDA),
                                heuristicDA = rownames(selectedHeuristicDA)),
                                filename=NULL, lty = "blank",
                                fill=c("pink1", "skyblue", "mediumorchid"),
                       main="Genes in common between the 3 methods in DA dataset")
grid.newpage()
grid.draw(myVenn3DA)
```



```
myVenn3GEO<- venn.diagram(x=list(naiveGEO=rownames(selectedNaiveGEO),
                                 cmiGEO=rownames(selectedCmiGEO),
                                 heuristicGEO = rownames(selectedHeuristicGEO)),
                                 filename=NULL, lty = "blank",
                                 fill=c("pink1", "skyblue", "mediumorchid"),
                       main="Genes in common between the 3 methods in GEO dataset")
grid.newpage()
grid.draw(myVenn3GEO)
```



```
myVenn3TCGA <- venn.diagram(x=list(naiveTCGA=rownames(selectedNaiveTCGA),
                                cmiTCGA=rownames(selectedCmiTCGA),
                                heuristicTCGA = rownames(selectedHeuristicTCGA)),
                                filename=NULL, lty = "blank",
                                fill=c("pink1", "skyblue", "mediumorchid"),
                       main="Genes in common between the 3 methods in TCGA dataset")
grid.newpage()
grid.draw(myVenn3TCGA)
```



4.1 Common results

A simple way to facilitat the comparison between the different scores and classifications provided by the distinct methods is simply to align these scores in a tabular format.

```
if (!exists ("naiveDA"))
  load("results/naiveSelections.Rda")
if (!exists ("cmiDA"))
  load("results/cmiSelections.Rda")
if (!exists ("scoresDA1"))
  load("results/heuristicSelections.Rda")
dim(naiveDA); head(naiveDA)

## [1] 11359     5
```

	r (Sp)	p (Sp)	adj.Spear.Pval	distCor	SigNegCorr
A1BG	0.1092	0.5656	0.8820	0.3175	FALSE
A2M	-0.3700	0.0442	0.3391	0.4542	FALSE
A2ML1	-0.1355	0.4753	0.8455	0.2765	FALSE
A4GALT	-0.3606	0.0503	0.3571	0.4278	FALSE
A4GNT	-0.1657	0.3814	0.8017	0.2531	FALSE
AAAS	-0.1293	0.4960	0.8541	0.2544	FALSE

```
dim(cmiDA); head(cmiDA)

## [1] 11359     4
```

	cMI_min	t_opt	ratio	meth_regulated
A1BG	0.0003	0.69	0.0903	FALSE
A2M	0.0060	0.49	0.0726	FALSE
A2ML1	0.0233	0.34	0.3245	FALSE
A4GALT	0.0338	0.34	0.3833	FALSE
A4GNT	0.0063	0.55	0.4343	FALSE
AAAS	0.0000	0	1.0000	FALSE

```
dim(scoresDA1); head(scoresDA1)

## [1] 11359     2
```

	logicSc	numericSc
A1BG	FALSE	-78
A2M	FALSE	-20
A2ML1	FALSE	0
A4GALT	TRUE	21
A4GNT	FALSE	-91
AAAS	FALSE	0

```
DAvalues <- cbind(naiveDA, cmiDA, scoresDA1);
head(DAvalues)
```

	r (Sp)	p (Sp)	adj.Spear.Pval	distCor	SigNegCorr	cMI_min	t_opt	ratio	meth_regulated
A1BG	0.1092	0.5656	0.8820	0.3175	FALSE	0.0003	0.69	0.0903	FALSE
A2M	-0.3700	0.0442	0.3391	0.4542	FALSE	0.0060	0.49	0.0726	FALSE
A2ML1	-0.1355	0.4753	0.8455	0.2765	FALSE	0.0233	0.34	0.3245	FALSE
A4GALT	-0.3606	0.0503	0.3571	0.4278	FALSE	0.0338	0.34	0.3833	FALSE
A4GNT	-0.1657	0.3814	0.8017	0.2531	FALSE	0.0063	0.55	0.4343	FALSE
AAAS	-0.1293	0.4960	0.8541	0.2544	FALSE	0.0000	0	1.0000	FALSE

```
GEOvalues <- cbind(naiveGEO, cmiGEO, scoresGEO)
head(GEOvalues)
```

	r (Sp)	p (Sp)	adj.Spear.Pval	distCor	SigNegCorr	cMI_min	t_opt	ratio	meth_regulated
7A5	0.4346	0.0299	0.5629	0.4752	FALSE	0.0107	0.15	0.7215	FALSE
A1BG	-0.0415	0.8437	0.9815	0.4135	FALSE	0.0202	0.91	0.4165	FALSE
A2BP1	0.0762	0.7175	0.9675	0.1491	FALSE	0.0498	0.18	0.2377	FALSE
A2ML1	-0.1392	0.5068	0.9291	0.1918	FALSE	0.0567	0.3	0.4678	FALSE
A4GALT	-0.1023	0.6265	0.9526	0.3119	FALSE	0.0104	0.22	0.5322	FALSE
A4GNT	-0.4654	0.0191	0.4740	0.2552	FALSE	0.0319	0.44	0.1731	TRUE

```
TCGAvalues <- cbind(naiveTCGA, cmiTCGA, scoresTCGA)
head(TCGAvalues)
```

	r (Sp)	p (Sp)	adj.Spear.Pval	distCor	SigNegCorr	cMI_min	t_opt	ratio	meth_regulated
A1BG	-0.0784	0.2437	0.4304	0.1104	FALSE	0.0103	0.62	0.7452	FALSE
A2M	-0.1889	0.0046	0.0229	0.2023	FALSE	0.0121	0.69	0.4491	FALSE
A2ML1	-0.0203	0.7627	0.8700	0.1178	FALSE	0.0100	0.37	0.2897	FALSE
A4GALT	-0.0866	0.1977	0.3758	0.1560	FALSE	0.0145	0.43	0.5731	FALSE
A4GNT	0.0893	0.1841	0.3586	0.1167	FALSE	0.0167	0.53	0.6076	FALSE
AAAS	-0.1915	0.0041	0.0208	0.1989	FALSE	0.0021	0.21	0.5165	FALSE

```
write.csv(DAvalues, file="results/LshapeScoresDA.csv")
write.csv(GEOvalues, file="results/LshapeScoresGEO.csv")
write.csv(TCGAvalues, file="results/LshapeScoresTCGA.csv")
# require(WriteXLS)
# WriteXLS(x=c("DAvalues", "GEOvalues", "TCGAvalues"), ExcelFileName = "results/LshapeScores.xls")
```

4.2 Plotting expression-methylation scaterplots

The best way to decide if a gene can be called L–shaped is to look at the methylation–expression scatterplot. Once we have computed the distinct scores we can combine these with the plots as shown below.

- DAExprAllScores.pdf
- DARNAseqLGenesScores.pdf
- GEOLGenesScores.pdf

```
if (!(exists("DAMetilData")))
  load(file="dades/DataMatrices-DA.Rda")
if (!(exists("geoMetilData")))
  load(file="dades/DataMatrices-GEO.Rda")
if (!(exists("TCGAMetilData")))
  load(file="dades/DataMatrices-TCGA.Rda")
```

```
if (!exists("DAvalues"))
  DAvalues<- read.csv(file="results/LshapeScoresDA.csv", row.names = 1)
DAscores4plots <- DAvalues[,c(1,3,10,11)]
if (!exists("GEOvalues"))
  GEOvalues<- read.csv(file="results/LshapeScoresGEO.csv", row.names = 1)
GEOscores4plots <- GEOvalues[,c(1,3,10,11)]
if (!exists("TCGAvalues"))
  TCGAvalues<- read.csv(file="results/LshapeScoresTCGA.csv", row.names = 1)
TCGAscores4plots <- TCGAvalues[,c(1,3,10,11)]
formatScores <- function(x){
  paste( "\n", "r(Sp)=", round(x[1], 4) , "; adj-p=", round(x[2],5),"\n", "Heur-L=", as.logical(x[3]
  }

DAscores<- apply(DAscores4plots,1, formatScores) # head(DAscores)
GEOscores<- apply(GEOscores4plots,1, formatScores) # head(GEOscores)
TCGAscores<- apply(TCGAscores4plots,1, formatScores) # head(TCGAscores)
```

```
replot <- TRUE
if(replot){
      plotGenesMat (mets=DAMetilData,
              expres=DAExprData,
              fileName = "results/DAExprAllScores.pdf",
              text4Title = DAscores)
      plotGenesMat (mets=geoMetilData,
              expres=geoExprData,
              fileName ="results/GEOLGenesScores.pdf",
              text4Title = GEOscores)
      plotGenesMat (mets=TCGAMetilData,
              expres=TCGAExprData,
              fileName = "results/TCGALGenesScores.pdf",
              text4Title = TCGAscores)
  }

## pdf
##   2
```

5 Locating genes along the genome

It is known that methylation happens at some specific regions in the genome ("cpG islands") more often than in other places (sometimes called "oceans"). We would like to know if genes that have been found to be regulated by methylation *in this study* are located at random in the genome or, alternatively, they concentrate in specific regions.

Altogether after applying the three methods –naive, cmi, heuristic– to the three datasets –DA, GEO, TCGA– we have selected 9 gene lists.

5.1 Annotating the genes in the chromosomes

Each gene list can be annotated with the positions of the genes in the chromosomes using the human reference genome assembly `hg19`, of UCSC (University of California, Santa Cruz) through the bioconductor packages, `Homo.sapiens` and `TxDb.Hsapiens.UCSC.hg19.knownGene`.

For each method and dataset the transcript coordinates have been obtained, saved to binary files `transcriptCoords-`*`Method-Dataset`*`.Rda` and written to .csv files `transcriptCoords-`*`Method-Dataset`*`.csv` in the folder `results`, where *Method* may be `Naive, CMI, Heuristic` and `Dataset` may be DA, GEO, TCGA.

For example the table of annotations corresponding to the `selectedNaiveDA` gene list has the following aspect:

	ENTREZID	SYMBOL	TXNAME	TXID	TXCHROM	TXSTART	TXEND
31	10103	TSPAN1	1244	uc001cpd.3	chr1	46640749	46651634
41	10158	PDZK1IP1	5278	uc001cqw.3	chr1	47649261	47655771
57	10472	ZBTB18	3989	uc031psv.1	chr1	244212241	244220780
61	10763	NES	6755	uc001fpq.3	chr1	156638556	156647189
81	11000	SLC27A3	2533	uc001fcz.3	chr1	153747768	153752633
171	1382	CRABP2	6756	uc001fpr.3	chr1	156669400	156675459

5.2 Visualization of selected genes in the chromosomes

An interesting question when selecting genes putatively regulated by Methylation is if the genes are located at random or if they are clustered or grouped according to some criterion.

Two distinct classes of objects can be reasonably related with methylated genes: `CpG Islands` and *DNAse I Hypersensitive Sites (DHS)*.

1. CpG islands are said to mark important regions in the genome because over 65% of gene promoter regions can be found within CpG islands. A dataset with the model CpG island mapped onto the hg19 genome can be obtained from the USCS (`http://hgdownload.cse.ucsc.edu/goldenpath/hg19/database/cpgIslandExt.txt.gz`)

2. DNAse I hypersensitive sites are regions of chromatin that are sensitive to cleavage by the DNase I enzyme. Since the discovery of DHSs 30 years ago, they have been used as markers of regulatory DNA regions. In these specific regions of the genome, chromatin has lost its condensed

structure, exposing the DNA and making it accessible. This raises the availability of DNA to degradation by enzymes, such as DNase I. These accessible chromatin zones are functionally related to transcriptional activity, since this remodeled state is necessary for the binding of proteins such as transcription factors. See `https://en.wikipedia.org/wiki/DNase_I_hypersensitive_site`. DHs can be downloaded from: `http://hgdownload.cse.ucsc.edu/goldenpath/hg19/encodeDCC/wgEncodeRegDnaseClustered/`.

In order to facilitate the study of the relation between these objects and the genes regulated by methylation CpG islands and DH sites will be plotted along the genome jointly with the selected genes.
The resulting plots will be printed to pdf files `Genes2Chromosomes-`*`Method-Dataset`*`.pdf` in the folder `results`, where *Method* may be `Naive, CMI, Heuristic` and `Dataset` may be DA, GEO, TCGA

5.2.1 Preparing data sources information

As a first step to plotting the coordinates of the *necessary* CpG islands and DH sites are obtained. By "necessary" we mean only those that correspond to positions located between the first transcript and the last transcripts associated with selected genes. Information is read from public files and stored in objects of type `GRanges`, a Bioconductor class to efficiently store information about sequences.
Next for each set of transcripts coordinates

```
redraw <- TRUE
if (redraw){
  for (i in 1:length(transcriptCoordsList)){
    plotGenesInChroms (transcriptCoordsList[[i]],
                       paste("results/genePositions", names(transcriptCoordsList)[i],".pdf", sep="")
                       minbase=minbase, maxbase=maxbase, islandData=islandData, dnaseData=dnaseData)
  }
}
```

Figure 2 shows the genes selected using the Naive method on the DA dataset in the first chromosome only.

```
# Select columns needed for plotting
anotacs4<-transcriptCoordsList[[1]][,c('TXCHROM', 'TXSTART', 'TXEND')]
anotacs4<-anotacs4[complete.cases(anotacs4),]
#change column names to read it into GenomicRanges
colnames(anotacs4)<-c("chromosome","start","end")
genRangList<-makeGRangesFromDataFrame(anotacs4) #if we keep geneid we add keep.extra.columns=TRUE in
# load context information necessary for plotting
data <- read.table(paste("dades","cytoBandIdeo.txt", sep="/"), header=F, sep="\t")
colnames(data) <-c('chrom', 'chromStart', 'chromEnd', 'name', 'gieStain')
ideoTrack <- IdeogramTrack(genome="mm10",chromosome = "chr2", bands=data)

# Go to plotting
axisT<-GenomeAxisTrack()
genList1<-AnnotationTrack(anotacs4, name = "Genes", genome ="hg19", chromosome ="chr1",
                          stacking ="dense", col= "#5E2366", fill= "#5E2366" )
ideoT1<-IdeogramTrack(genome="hg19", bands = data,chromosome = "chr1", showId=FALSE)
islandData1 <- islandData[seqnames(islandData) == "chr1" &  (start(islandData) >= minbase & end(isla
islandTrack1 <- AnnotationTrack(range=islandData1, genome="hg19", name="CpG Islands",
                               chromosome="chr1")
# DNaseI hypersensitive site data track
dnaseTrack1 <- DataTrack(range=dnaseData, genome="hg19", name="DNAseI",
                        type="gradient", chromosome="chr1")
pdf(file="results/chromosome1NaiveDA.pdf", width= 8, height = 12)
  plotTracks(list(ideoT1,axisT,genList1, dnaseTrack1, islandTrack1),
             sizes=c(1,2,2,1,20), main="chr1", cex.main=1,  littleTicks = TRUE, showTitle=TRUE)
dev.off()
```

Figure 2: Distribution of genes on Chromosome 1. The genes are plotted as vertical purple lines, blue lines are the DNAse I hypersensitive sites and grey lines represent an accumulation of CpG islands.

5.3 Overlap between CpG islands and selected genes

Additionally to drawing genes and CpG regions together one can find out the overlap between gene regions and CpG islands. This can be done by calling "CpG islands" genomic windows whose GC content>50% and observed-to-expected CG ratio>0.6.
The overlap can be computed based on two methods: "standadrd" and "within" defined in the documentation of Bioconductor `GRanges` package.
For example, for the first chromosome and the gene list obtained by the Naive method on DA dataset we have:

```r
#Subset CpG islands and create a new GRanges object
islandHMM2 <- islandHMM[islandHMM$GCcontent>50 & islandHMM$obsExp>0.6,]

islandDataSel <- GRanges(seqnames=Rle(islandHMM2$chr),
                      ranges=IRanges(start=islandHMM2$start, end=islandHMM2$end),
                      strand=Rle(strand(rep("*",nrow(islandHMM2)))))
## Overlapping two GRanges objects:
table(!is.na(findOverlaps(islandDataSel, genRangList, select="arbitrary")))
countFeatOverlaps <- countOverlaps(islandDataSel, genRangList)
findFeatOverlaps <- findOverlaps(islandDataSel, genRangList)
subsetFeatOverlaps <- subsetByOverlaps(islandDataSel, genRangList)
findFeatOverlapsWITHIN <- findOverlaps(genRangList, islandDataSel, type="within")
```

References

[1] Sarah Bazzocco, Hafid Alazzouzi, M. Carme Ruiz de Villa, Alex Sanchez-Pla, John M. Mariadason, Diego Arango (2013) *Genome-Wide Analysis of DNA Methylation in Colorectal Cancer*. Submitted.

[2] Yihua Liu and Peng Qiu. (2012) *Integrative analysis of methylation and gene expression data in TCGA* IEEE International Workshop on Genomic Signal Processing and Statistics (GENSIPS)

[3] R Development Core Team (2005). R: A language and environment for statistical computing, reference index version 2.14.0. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, `http://www.R-project.org`

[4] Jeffrey Racine. (2012) A primer on regression splines. `http://cran.r-project.org/web/packages/crs/vignettes/spline_primer.pdf`