

# ELMER: An R/Bioconductor Tool Inferring Regulatory Element Landscapes and Transcription Factor Networks Using Methylomes

Lijing Yao

March 18, 2015

## Contents

---

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Installing and loading ELMER . . . . .	2
<b>2</b>	<b>Download example data</b>	<b>2</b>
<b>3</b>	<b>Quick start: running TCGA example</b>	<b>3</b>
<b>4</b>	<b>Input data</b>	<b>4</b>
4.1	DNA methylation data . . . . .	4
4.2	Gene expression data . . . . .	5
4.3	Sample information . . . . .	5
4.4	Probe information . . . . .	5
4.5	Gene information . . . . .	6
4.6	MEE.data . . . . .	7
<b>5</b>	<b>Illustration of ELMER analysis steps</b>	<b>8</b>
5.1	Selection of probes within biofeatures . . . . .	8
5.2	Identifying differentially methylated probes . . . . .	8
5.3	Identifying putative probe-gene pairs . . . . .	9
5.4	Motif enrichment analysis on the selected probes . . . . .	10
5.5	Identifying regulatory TF . . . . .	10
<b>6</b>	<b>Generating figures</b>	<b>11</b>
6.1	Scatter plots . . . . .	11
6.1.1	Scatter Plot of 20 nearby genes . . . . .	11
6.1.2	Scatter Plot of One Pair . . . . .	11
6.1.3	TF expression vs. average DNA methylation . . . . .	11
6.2	Schematic plot . . . . .	11
6.2.1	Nearby Genes . . . . .	12
6.2.2	Nearby Probes . . . . .	13
6.3	Motif enrichment plot . . . . .	13
6.4	TF ranking plot . . . . .	13

## 1 Introduction

---

This document provides an introduction of the *ELMER*, which is designed to use DNA methylation and gene expression data sets from a large number of tissue samples to infer regulatory element landscapes and transcription factor network. It includes functions for identifying probes at distal regulatory regions with differential DNA methylation levels, predicting genes whose expression associates with the differentially methylated probes and discovering the functional regulatory TFs. This package can be easily applied to TCGA public available cancer data sets and to custom DNA methylation and gene expression data sets.

### 1.1 Installing and loading ELMER

To obtain a copy of *ELMER*, you will need to install *devtools*

```
install.packages(devtools)
library(devtools);
devtools::install_github("lijingya/ELMER");
```

## 2 Download example data

---

The following steps can be used to download the example data set for *ELMER*

```
## Loading required package: IlluminaHumanMethylation450kanno.ilmn12.hg19
## Loading required package: minfi
## 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 object is masked from 'package:stats':
##
##   xtabs
##
## The following objects are masked from 'package:base':
##
##   anyDuplicated, append, as.data.frame, as.vector, cbind, colnames, do.call,
##   duplicated, eval, evalq, Filter, Find, get, intersect, is.unsorted, lapply,
##   Map, mapply, match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##   Position, rank, rbind, Reduce, rep.int, rownames, sapply, setdiff, sort,
##   table, tapply, union, unique, unlist, unsplit
##
## Loading required package: Biobase
## Welcome to Bioconductor
##
##   Vignettes contain introductory material; view with 'browseVignettes()'. To
##   cite Bioconductor, see 'citation("Biobase")', and for packages
```

```
##      'citation("pkgname")'.
##
## Loading required package: lattice
## Loading required package: GenomicRanges
## Loading required package: S4Vectors
## Loading required package: stats4
## Loading required package: IRanges
## Loading required package: GenomeInfoDb
## Loading required package: Biostrings
## Loading required package: XVector
## Loading required package: bumpHunter
## Loading required package: foreach
## Loading required package: iterators
## Loading required package: locfit
## locfit 1.5-9.1 2013-03-22
##
## Attaching package: 'ELMER'
##
## The following objects are masked from 'package:minfi':
##
##      getMeth, getProbeInfo

#Example file download from URL: https://dl.dropboxusercontent.com/u/61961845/ELMER.example.tar.gz
URL <- "https://dl.dropboxusercontent.com/u/61961845/ELMER.example.tar.gz"
download.file(URL,destfile = "ELMER.example",method= "wget",
              extra = c("--no-check-certificate -a download.log"))
untar("./ELMER.example")
library(ELMER)
```

### 3 Quick start: running TCGA example

Function, TCGA.pipe, is the easy usage for downloading TCGA data and performing all the analyses in ELMER. For illustration purpose, we skip the downloading step. The user can use the getTCGA function to download TCGA data or use TCGA.pipe by including "download" in the analysis option.

```
TCGA.pipe("LUSC",wd="/ELMER.example",cores=detectCores()/2,permu.size=300,
          analysis = c("distal.enhancer","diffMeth","pair","motif","TF.search"),
          diff.dir="hypo",rm.chr=paste0("chr",c(1:22,"X","Y")))

## #####
## Select distal enhancer probes
## #####

## Warning in (function (probe, distal = TRUE, feature, TSS, TSS.range = list(upstream = 2000, :
## Default probes coordinates are for HM450K DNA methylation array

## #####
## Get differential DNA methylation loci
## #####
##
##
## ~~~ MEE.data:  initializer ~~~
## #####
## Predict pairs
```

```
## #####
##
##
## ~~~ MEE.data:  initializator ~~~
## Identify putative probe-gene pair for hypomethylated probes
## Calculate empirical P value.
##
## #####
## Motif search
## #####
##
##
## Identify enriched motif for hypomethylated probes
## 12 motifs are enriched.
## #####
## Search responsible TFs
## #####
##
##
## ~~~ MEE.data:  initializator ~~~
## Identify regulatory TF for enriched motif in hypomethylated probes
```

## 4 Input data

The whole pipeline analyses in *ELMER* needs at least 4 input files: a matrix of DNA methylation from HM450K platform; a matrix of gene expression for the same samples; a GRanges object containing the information for probes on HM450K such as names and coordinates; a gene annotation which is also a GRanges object. When TCGA data are used, the sample information will be automatically generated by `fetch.mee` function. However sample information should be provided when using custom data.

### 4.1 DNA methylation data

Raw DNA methylation data can be processed by *Methylumi* or *minfi* generating DNA methylation information for each CpG. The DNA methylation level at each CpG is referred to as a beta ( $\beta$ ) value, calculated as  $(M/(M+U))$ , where M represents the methylated allele intensity and U the unmethylated allele intensity. Beta values range from 0 to 1, reflecting the fraction of methylated alleles at each CpG in the each tumor; beta values close to 0 indicates low levels of DNA methylation and beta values close to 1 indicates high levels of DNA methylation. Generate a matrix with DNA methylation beta values for all the samples (columns) and probe loci (rows) and save matrix as `meth.rda`

```
load("./ELMER.example/Result/LUSC/LUSC_meth_refined.rda")
Meth[1:10, 1:2]

##          TCGA-43-3394-11A-01D-1551-05  TCGA-43-3920-11B-01D-1551-05
## cg00045114          0.8190894          0.8073763
## cg00050294          0.8423084          0.8241138
## cg00066722          0.9101127          0.9162212
## cg00093522          0.8751903          0.8864599
## cg00107046          0.3326016          0.3445508
## cg00116430          0.6097183          0.5952469
## cg00152117          0.7074149          0.6439695
## cg00163018          0.5928909          0.8250584
## cg00173804          0.9162264          0.9303684
```

```
## cg00223046 0.7826863 0.7744760
```

## 4.2 Gene expression data

Gene expression values can be generated from different platforms such as array or RNA-seq, gene level or transcript level gene expression calling. Generate a matrix with gene expression values for all the samples (columns) and genes (rows) and save matrix as RNA.rda

```
load("./ELMER.example/Result/LUSC/LUSC_RNA_refined.rda")
GeneExp[1:10, 1:2]

##          TCGA-22-5472-01A-01R-1635-07 TCGA-22-5489-01A-01R-1635-07
## ID126767          0.0000000          0.000000
## ID343066          0.4303923          0.000000
## ID26574           10.0817831         10.717673
## ID24              6.4462711          6.386644
## ID23456           8.5929182          9.333097
## ID5825            10.5578756          9.878333
## ID25              10.7233258         11.075515
## ID27              8.9761542          9.569239
## ID29777           9.6415206          9.353424
## ID80325           8.9840983          9.177624
```

## 4.3 Sample information

Sample information should be stored as a data.frame object containing sample ID, group labels (such as tumor, normal) and other description for each sample. When TCGA data were used, tumor, normal group label will be automatically generated by fetch.mee function by specifying option TCGA=TRUE.

```
mee <- fetch.mee(meth=Meth, exp=GeneExp, TCGA=TRUE)
## ~~~ MEE.data:  initializer ~~~

head(getSample(mee))

##          ID          meth.ID          exp.ID
## TCGA-43-3394-11 TCGA-43-3394-11 TCGA-43-3394-11A-01D-1551-05 TCGA-43-3394-11A-01R-1758-07
## TCGA-56-8305-01 TCGA-56-8305-01 TCGA-56-8305-01A-11D-2294-05 TCGA-56-8305-01A-11R-2296-07
## TCGA-56-8307-01 TCGA-56-8307-01 TCGA-56-8307-01A-11D-2294-05 TCGA-56-8307-01A-11R-2296-07
## TCGA-56-8308-01 TCGA-56-8308-01 TCGA-56-8308-01A-11D-2294-05 TCGA-56-8308-01A-11R-2296-07
## TCGA-56-8309-01 TCGA-56-8309-01 TCGA-56-8309-01A-11D-2294-05 TCGA-56-8309-01A-11R-2296-07
## TCGA-58-8386-01 TCGA-58-8386-01 TCGA-58-8386-01A-11D-2294-05 TCGA-58-8386-01A-11R-2296-07
##          TN
## TCGA-43-3394-11 Normal
## TCGA-56-8305-01 Tumor
## TCGA-56-8307-01 Tumor
## TCGA-56-8308-01 Tumor
## TCGA-56-8309-01 Tumor
## TCGA-58-8386-01 Tumor
```

## 4.4 Probe information

Probe information should be stored as a GRanges object containing the coordinate of each probe on the DNA methylation array and names of each probe. The default probe information is for HM450K.

```
probe <- getAnnotation(IlluminaHumanMethylation450kanno.ilmn12.hg19, what="Locations")
probe <- GRanges(seqnames=probe$chr,
                 ranges=IRanges(probe$pos,
                                width=1,
                                names=rownames(probe)),
                 strand=probe$strand,
                 name=rownames(probe))
mee <- fetch.mee(meth=Meth, exp=GeneExp, TCGA=TRUE, probeInfo=probe)

## ~~~ MEE.data: initializer ~~~

getProbeInfo(mee)

## GRanges object with 1725 ranges and 1 metadata column:
##           seqnames          ranges strand |          name
##           <Rle>           <IRanges> <Rle> | <character>
## cg00116430 chr1 [ 94188268, 94188268] + | cg00116430
## cg00889627 chr1 [ 1959630, 1959630] - | cg00889627
## cg01071265 chr1 [160952651, 160952651] + | cg01071265
## cg01074104 chr1 [ 41324394, 41324394] - | cg01074104
## cg01393939 chr1 [ 87803705, 87803705] + | cg01393939
## ...
## cg27584013 chr1 [ 23012439, 23012439] + | cg27584013
## cg27589988 chr1 [215147891, 215147891] + | cg27589988
## cg27637706 chr1 [ 3472204, 3472204] - | cg27637706
## ch.1.131529R chr1 [ 3283394, 3283394] + | ch.1.131529R
## ch.1.173213985R chr1 [174947362, 174947362] + | ch.1.173213985R
## -----
## seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

## 4.5 Gene information

Gene information should be stored as a GRanges object containing coordinates of each gene, gene id, gene symbol and gene isoform id. The default gene information is the UCSC gene annotation.

```
load(system.file("extdata", "UCSC_gene_hg19.rda", package = "ELMER"))
## In TCGA expression data, geneIDs were used as the rowname for each row. However, numbers
## can't be the rownames, "ID" was added to each gene id functioning as the rowname.
## If your geneID is consistent with the rownames of the gene expression matrix, adding "ID"
## to each geneID can be skipped.
txs$GENEID <- paste0("ID", txs$GENEID)
geneInfo <- promoters(txs, upstream = 0, downstream = 0)
save(geneInfo, file = ".ELMER.example/Result/LUSC/geneAnnot.rda")
mee <- fetch.mee(meth=Meth, exp=GeneExp, TCGA=TRUE, geneInfo=txs)

## ~~~ MEE.data: initializer ~~~

getGeneInfo(mee)

## GRanges object with 13741 ranges and 3 metadata columns:
##           seqnames          ranges strand | tx_name      GENEID      SYMBOL
##           <Rle>           <IRanges> <Rle> | <character> <character> <character>
## 10000 chr1 [243651535, 244006584] - | uc001hzz.1 ID10000 AKT3
## 10000 chr1 [243663021, 244006584] - | uc001iab.2 ID10000 AKT3
## 10000 chr1 [243663021, 244006886] - | uc021plu.1 ID10000 AKT3
## 10001 chr14 [ 71050957, 71067384] - | uc001xmf.3 ID10001 MED6
```

```
## 10001 chr14 [ 71050957, 71067384] - | uc010tth.2 ID10001 MED6
## ... ... ... ... ...
## 9988 chr7 [86781677, 86825648] + | uc003uik.3 ID9988 DMTF1
## 9988 chr7 [86781677, 86825648] + | uc011khh.2 ID9988 DMTF1
## 9988 chr7 [86781870, 86825648] + | uc003uil.3 ID9988 DMTF1
## 9988 chr7 [86792198, 86809018] + | uc003uim.1 ID9988 DMTF1
## 9988 chr7 [86792198, 86825648] + | uc003uin.3 ID9988 DMTF1
## -----
## seqinfo: 93 sequences (1 circular) from hg19 genome
```

## 4.6 MEE.data

The above 5 components will generate a MEE.data object as the main input for multiple functions in *ELMER*.

```
mee <- fetch.mee(meth=Meth, exp=GeneExp, TCGA=TRUE, probeInfo=probe, geneInfo=txs)
```

```
## ~~~ MEE.data: initializer ~~~
```

```
mee
```

```
## *** Class MEE.data, method show ***
```

```
## * meth
```

```
## num [1:1725, 1:234] 0.819 0.842 0.91 0.875 0.333 ...
```

```
## - attr(*, "dimnames")=List of 2
```

```
## ..$ : chr [1:1725] "cg00045114" "cg00050294" "cg00066722" "cg00093522" ...
```

```
## ..$ : chr [1:234] "TCGA-43-3394-11A-01D-1551-05" "TCGA-56-8305-01A-11D-2294-05" "TCGA-56-8307-01A-11D-2294-05" ...
```

```
## NULL
```

```
## * exp
```

```
## num [1:3894, 1:234] 0 0.214 10.048 5.007 8.63 ...
```

```
## - attr(*, "dimnames")=List of 2
```

```
## ..$ : chr [1:3894] "ID126767" "ID343066" "ID26574" "ID24" ...
```

```
## ..$ : chr [1:234] "TCGA-43-3394-11A-01R-1758-07" "TCGA-56-8305-01A-11R-2296-07" "TCGA-56-8307-01A-11R-2296-07" ...
```

```
## NULL
```

```
## * sample
```

```
## 'data.frame': 234 obs. of 4 variables:
```

```
## $ ID : chr "TCGA-43-3394-11" "TCGA-56-8305-01" "TCGA-56-8307-01" "TCGA-56-8308-01" ...
```

```
## $ meth.ID: chr "TCGA-43-3394-11A-01D-1551-05" "TCGA-56-8305-01A-11D-2294-05" "TCGA-56-8307-01A-11D-2294-05" ...
```

```
## $ exp.ID : chr "TCGA-43-3394-11A-01R-1758-07" "TCGA-56-8305-01A-11R-2296-07" "TCGA-56-8307-01A-11R-2296-07" ...
```

```
## $ TN : chr "Normal" "Tumor" "Tumor" "Tumor" ...
```

```
## NULL
```

```
## * probeInfo
```

```
## GRanges object with 1725 ranges and 1 metadata column:
```

	seqnames	ranges	strand	name
	<Rle>	<IRanges>	<Rle>	<character>
##	cg00116430	chr1 [ 94188268, 94188268]	+	cg00116430
##	cg00889627	chr1 [ 1959630, 1959630]	-	cg00889627
##	cg01071265	chr1 [160952651, 160952651]	+	cg01071265
##	cg01074104	chr1 [ 41324394, 41324394]	-	cg01074104
##	cg01393939	chr1 [ 87803705, 87803705]	+	cg01393939
##	...	...	...	...
##	cg27584013	chr1 [ 23012439, 23012439]	+	cg27584013
##	cg27589988	chr1 [215147891, 215147891]	+	cg27589988
##	cg27637706	chr1 [ 3472204, 3472204]	-	cg27637706
##	ch.1.131529R	chr1 [ 3283394, 3283394]	+	ch.1.131529R

```
## ch.1.173213985R chr1 [174947362, 174947362] + | ch.1.173213985R
## -----
## seqinfo: 24 sequences from an unspecified genome; no seqlengths
## * geneInfo
## GRanges object with 13741 ranges and 3 metadata columns:
##      seqnames      ranges strand | tx_name      GENEID      SYMBOL
##      <Rle>        <IRanges> <Rle> | <character> <character> <character>
## 10000 chr1 [243651535, 244006584] - | uc001hzz.1 ID10000 AKT3
## 10000 chr1 [243663021, 244006584] - | uc001iab.2 ID10000 AKT3
## 10000 chr1 [243663021, 244006886] - | uc021plu.1 ID10000 AKT3
## 10001 chr14 [ 71050957, 71067384] - | uc001xf.3 ID10001 MED6
## 10001 chr14 [ 71050957, 71067384] - | uc010tth.2 ID10001 MED6
## ...      ...      ...      ...      ...
## 9988 chr7 [86781677, 86825648] + | uc003uik.3 ID9988 DMTF1
## 9988 chr7 [86781677, 86825648] + | uc011khh.2 ID9988 DMTF1
## 9988 chr7 [86781870, 86825648] + | uc003uil.3 ID9988 DMTF1
## 9988 chr7 [86792198, 86809018] + | uc003uim.1 ID9988 DMTF1
## 9988 chr7 [86792198, 86825648] + | uc003uin.3 ID9988 DMTF1
## -----
## seqinfo: 93 sequences (1 circular) from hg19 genome
## ***** End Print (MEE.data) *****
```

## 5 Illustration of ELMER analysis steps

A subset of chromosome 1 data from TCGA LUSC were used as illustration.

### 5.1 Selection of probes within biofeatures

Function, `get.feature.probe`, is used to select probes that are located within biofeatures such as H3K27ac ChIP-seq peaks. As default, the `get.feature.probe` function will automatically select distal enhancer probes on HM450K which are at least 2kb away from the TSS annotated by GENCODE V15 and UCSC-gene and locate within the putative comprehensive enhancers from REMC, ENCODE and FANTOM5.

```
#get distal enhancer probes that are 2kb away from TSS and overlap with REMC and FANTOM5
#enhancers on chromosome 1
Probe <- get.feature.probe(probe=probe, rm.chr=paste0("chr",c(2:22,"X","Y")))
save(Probe,file="./ELMER.example/Result/LUSC/probeInfo_feature.rda")
```

### 5.2 Identifying differentially methylated probes

Function, `get.diff.meth`, will be used to identify differentially methylated probes among the ones within biofeatures, which are selected in the above step.

```
## fetch.mee can take path as input.
mee <- fetch.mee(meth="./ELMER.example/Result/LUSC/LUSC_meth_refined.rda",
  exp="./ELMER.example/Result/LUSC/LUSC_RNA_refined.rda", TCGA=TRUE,
  probeInfo="./ELMER.example/Result/LUSC/probeInfo_feature.rda",
  geneInfo="./ELMER.example/Result/LUSC/geneAnnot.rda")

## ~~~ MEE.data: initializer ~~~
```



```
sig.diff <- get.diff.meth(mee, cores=detectCores()/2, dir.out = "./ELMER.example/Result/LUSC",
                        diff.dir="hypo", pvalue = 0.01)

sig.diff$hypo[1:10,] ## significantly hypomethylated probes

##           probe           pvalue tumorMinNormal      adjust.p
## cg00045114 cg00045114 7.307478e-13      -0.3499588 2.419463e-12
## cg00050294 cg00050294 4.440329e-09      -0.5000920 1.057951e-08
## cg00093522 cg00093522 1.143338e-23      -0.3596944 1.001147e-22
## cg00163018 cg00163018 2.240026e-21      -0.3838195 1.558083e-20
## cg00173804 cg00173804 2.767232e-18      -0.3744557 1.455328e-17
## cg00223046 cg00223046 8.142751e-11      -0.3313773 2.317862e-10
## cg00255699 cg00255699 6.386989e-13      -0.4308994 2.126941e-12
## cg00292636 cg00292636 6.368018e-34      -0.4833375 2.112468e-32
## cg00329272 cg00329272 6.864176e-39      -0.4355332 5.638431e-37
## cg00340127 cg00340127 3.056522e-18      -0.5857140 1.602584e-17

# get.diff.meth automatically save output files.
# getMethdiff.hypo.probes.csv contains statistics for all the probes.
# getMethdiff.hypo.probes.significant.csv contains only the significant probes.
dir(path = "./ELMER.example/Result/LUSC", pattern = "getMethdiff")

## [1] "getMethdiff.hypo.probes.csv" "getMethdiff.hypo.probes.significant.csv"
```

### 5.3 Identifying putative probe-gene pairs

Function, `get.pair` function, will be used to identify putative target genes for selected probes. This step is the most time consuming step since it requires a large amount calculations for permutation. The greater the permutation time is, the longer it will take. It is recommended to use multiple cores for this step. Default permutation time is 1000 which may need 12 hrs by 4 cores. However 10,000 permutations is recommended which may cost 2 days, if high confidence results are desired.

```
### identify target gene for significantly hypomethylated probes.

Sig.probes <- read.csv("./ELMER.example/Result/LUSC/getMethdiff.hypo.probes.significant.csv",
                      stringsAsFactors=FALSE)[,1]
head(Sig.probes) # significantly hypomethylated probes

## [1] "cg00045114" "cg00050294" "cg00093522" "cg00163018" "cg00173804" "cg00223046"

## Collect nearby 20 gene for Sig.probes
nearGenes <- GetNearGenes(TRange=getProbeInfo(mee,probe=Sig.probes),
                        geneAnnot=getGeneInfo(mee),cores=detectCores()/2)

## Identify significant probe-gene pairs
Hypo.pair <- get.pair(mee=mee,probes=Sig.probes,nearGenes=nearGenes,
                    permu.dir="./ELMER.example/Result/LUSC/permu",permu.size=300,Pe = 0.01,
                    dir.out="./ELMER.example/Result/LUSC",cores=detectCores()/2,label= "hypo")

## Calculate empirical P value.
head(Hypo.pair) ## significant probe-gene pairs

##           Probe      GeneID      Symbol Distance Sides      Raw.p      Pe
## cg20701183.ID8543 cg20701183 ID8543      LM04      2563      L2 7.453984e-14 0.003322259
## cg19403323.ID255928 cg19403323 ID255928 SYT14      87477      R1 1.671937e-12 0.003322259
## cg12213388.ID84451 cg12213388 ID84451 KIAA1804 993548      L4 2.527644e-12 0.003322259
```

```
## cg26607897.ID55811 cg26607897 ID55811 ADCY10 292476 R4 4.593610e-12 0.003322259
## cg10574861.ID8543 cg10574861 ID8543 LM04 4715 L2 4.770162e-12 0.003322259
## cg26607897.ID23432 cg26607897 ID23432 GPR161 563308 R7 8.048248e-12 0.003322259

# get.pair automatically save output files.
# getPair.hypo.all.pairs.statistic.csv contains statistics for all the probe-gene pairs.
# getPair.hypo.pairs.significant.csv contains only the significant probes.
dir(path = "./ELMER.example/Result/LUSC", pattern = "getPair")

## [1] "getPair.hypo.all.pairs.statistic.csv" "getPair.hypo.pairs.significant.csv"
```

## 5.4 Motif enrichment analysis on the selected probes

Function, `get.enriched.motif`, will be used to calculate enrichment of the motifs from factorbook and JASPER for the selected probes. Odds Ratio is used to assess the enrichment levels and 95% confidence interval of Odds Ratio is calculated.

```
### identify enriched motif for significantly hypomethylated probes which
### have putative target genes.

Sig.probes.paired <- read.csv("./ELMER.example/Result/LUSC/getPair.hypo.pairs.significant.csv",
                             stringsAsFactors=FALSE)[,1]
head(Sig.probes.paired) # significantly hypomethylated probes with putative target genes
## [1] "cg20701183" "cg19403323" "cg12213388" "cg26607897" "cg10574861" "cg26607897"

enriched.motif <- get.enriched.motif(probes=Sig.probes.paired,
                                     dir.out="./ELMER.example/Result/LUSC", label="hypo",
                                     min.incidence = 10, lower.OR = 1.1)

## 6 motifs are enriched.

names(enriched.motif) # enriched motifs
## [1] "AP1" "BARHL2" "IRF" "PRDM1" "TCF7L2" "TP53"

# get.enriched.motif automatically save output files.
# getMotif.hypo.enriched.motifs.rda contains enriched motifs and the probes with the motif.
# getMotif.hypo.motif.enrichment.csv contains summary of enriched motifs.
dir(path = "./ELMER.example/Result/LUSC", pattern = "getMotif")

## [1] "getMotif.hypo.enriched.motifs.rda" "getMotif.hypo.motif.enrichment.csv"

# motif enrichment figure will be automatically generated.
dir(path = "./ELMER.example/Result/LUSC", pattern = "motif.enrichment.pdf")

## [1] "hypo.motif.enrichment.pdf"
```

## 5.5 Identifying regulatory TF

Function, `get.TFs`, will use the anti-correlation of a particular TF and the level of demethylation at its binding sites to predict the regulatory TF.

```
### identify regulatory TF for the enriched motifs

load("./ELMER.example/Result/LUSC/getMotif.hypo.enriched.motifs.rda")
TF <- get.TFs(mee=mee, enriched.motif=enriched.motif, dir.out="./ELMER.example/Result/LUSC",
              cores=detectCores()/2, label= "hypo")
```

```
# get.TFs automatically save output files.
# getTF.hypo.TFs.with.motif.pvalue.rda contains statistics for all TF with average
# DNA methylation at sites with the enriched motif.
# getTF.hypo.significant.TFs.with.motif.summary.csv contains only the significant probes.
dir(path = "./ELMER.example/Result/LUSC", pattern = "getTF")

## [1] "getTF.hypo.significant.TFs.with.motif.summary.csv"
## [2] "getTF.hypo.TFs.with.motif.pvalue.rda"

# TF ranking plot based on statistics will be automatically generated.
dir(path = "./ELMER.example/Result/LUSC/TFrankPlot", pattern = "pdf")

## [1] "AP1.TFrankPlot.pdf"      "BARHL2.TFrankPlot.pdf"  "IRF.TFrankPlot.pdf"
## [4] "PRDM1.TFrankPlot.pdf"    "TCF7L2.TFrankPlot.pdf"  "TP53.TFrankPlot.pdf"
```

## 6 Generating figures

---

### 6.1 Scatter plots

#### 6.1.1 Scatter Plot of 20 nearby genes

Generate scatter plots for one probes' nearby 20 gene expression vs DNA methylation at this probe. Figure 1

```
scatter.plot(mee, byProbe=list(probe=c("cg19403323"), geneNum=20),
             category="TN", save=FALSE)

## cg19403323
```

#### 6.1.2 Scatter Plot of One Pair

Generate a scatter plot for one probe-gene pair. Figure 2

```
scatter.plot(mee, byPair=list(probe=c("cg19403323"), gene=c("ID255928")),
             category="TN", save=FALSE, lm_line=TRUE)
```

#### 6.1.3 TF expression vs. average DNA methylation

Generate scatter plot for TF expression vs average DNA methylation of the sites with certain motif. Figure 3

```
load("ELMER.example/Result/LUSC/getMotif.hypo.enriched.motifs.rda")
scatter.plot(mee, byTF=list(TF=c("TP53", "TP63", "TP73"),
                             probe=enriched.motif[["TP53"]]), category="TN",
             save=FALSE, lm_line=TRUE)
```

### 6.2 Schematic plot

Schematic plot shows a brief view of linkages between genes and probes. To make a schematic plot, "Pair" object should be generated first.

```
# Make a "Pair" object for schematic.plot
pair <- fetch.pair(pair="./ELMER.example/Result/LUSC/getPair.hypo.pairs.significant.withmotif.csv",
                  probeInfo = "./ELMER.example/Result/LUSC/probeInfo_feature.rda",
```

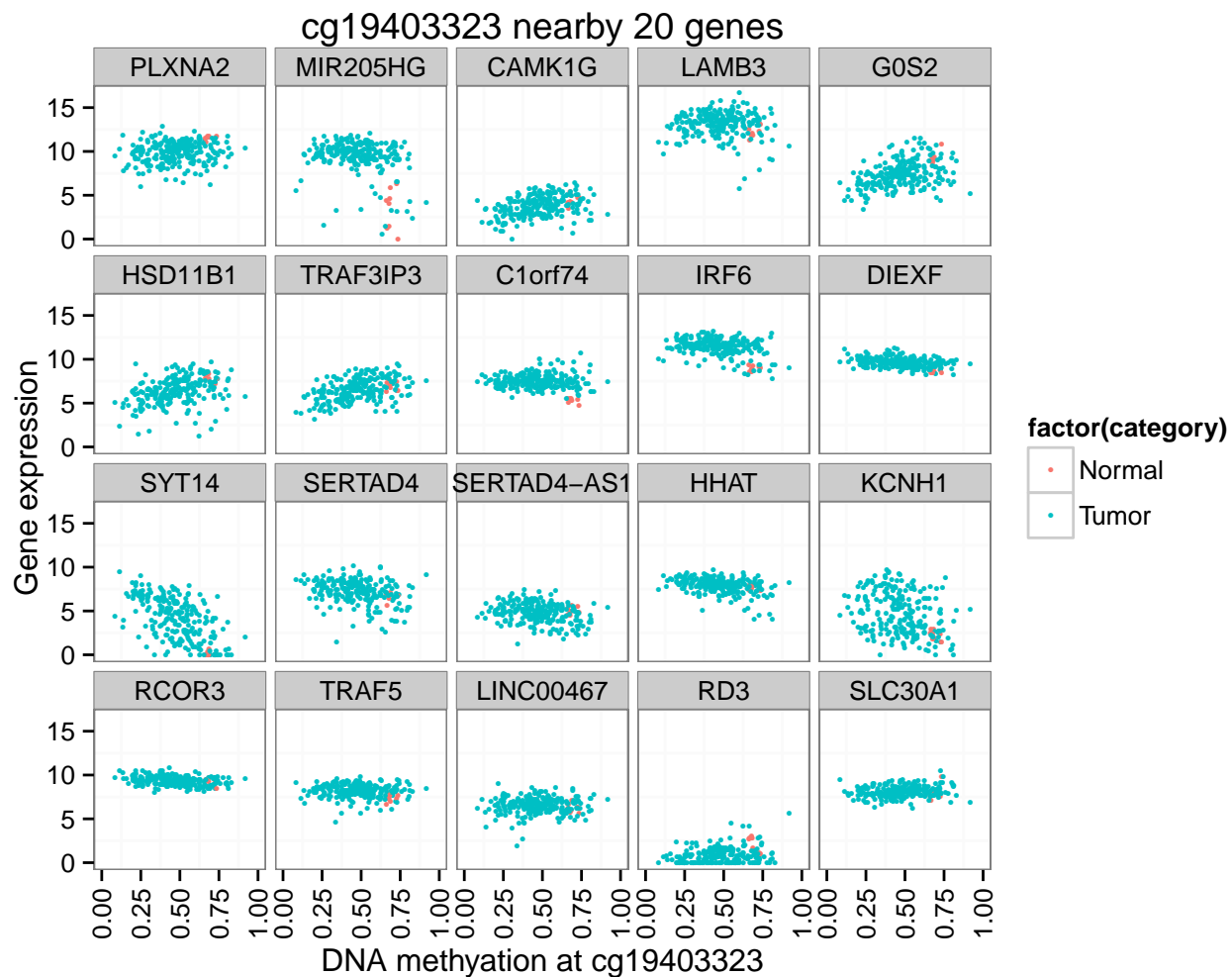


Figure 1: Each scatter plot shows the methylation level of an example probe cg19403323 in all LUSC samples plotted against the expression of one of 20 adjacent genes.

```
geneInfo = "./ELMER.example/Result/LUSC/geneAnnot.rda")
## ~~~ Pair:  initializer ~~~
```

### 6.2.1 Nearby Genes

Generate schematic plot for one probe with 20 nearby genes and label the gene significantly linked with the probe in red. Figure 4

```
grid::grid.newpage()
schematic.plot(pair=pair, byProbe="cg19403323", save=FALSE)
## cg19403323
```

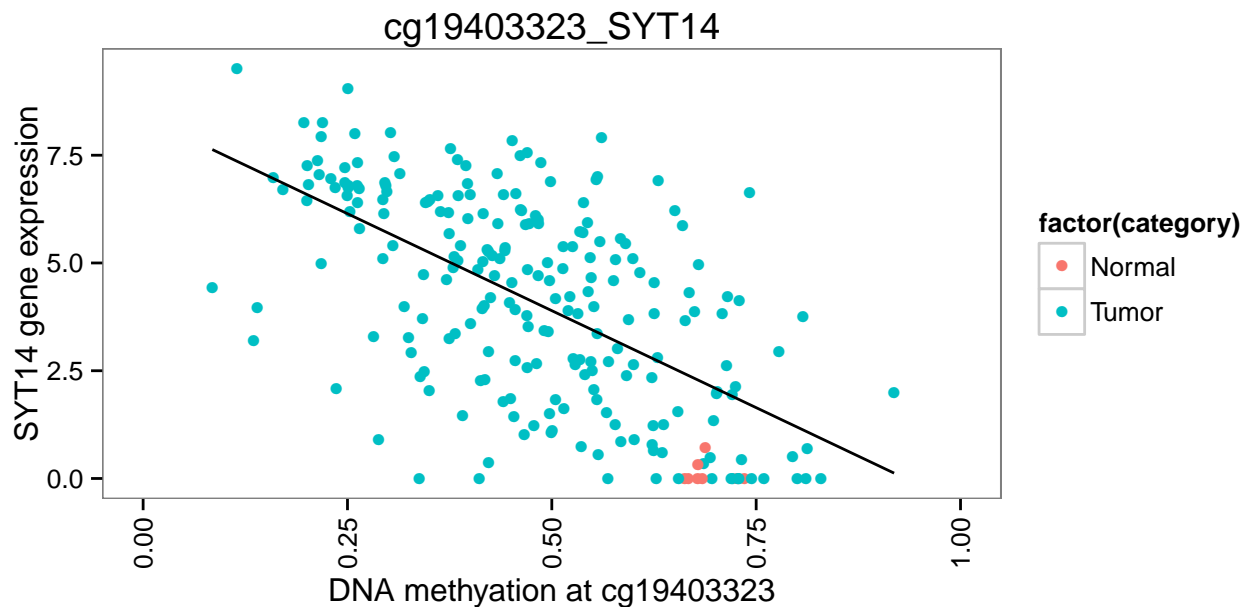


Figure 2: Scatter plot shows the methylation level of an example probe cg19403323 in all LUSC samples plotted against the expression of the putative target gene SYT14.

### 6.2.2 Nearby Probes

Generate schematic plot for one gene with the probes which the gene is significantly linked to. Figure 5

```
grid::grid.newpage()
schematic.plot(pair=pair, byGene="ID255928", save=FALSE)
```

## 6.3 Motif enrichment plot

Motif enrichment plot shows the enrichment levels for the selected motifs. Figure 6

```
motif.enrichment.plot(motif.enrichment="./ELMER.example/Result/LUSC/getMotif.hypo.motif.enrichment.csv",
                      significant=list(OR=1.3, lowerOR=1.3), label="hypo", save=FALSE)
```

## 6.4 TF ranking plot

TF ranking plot shows statistic  $-\log_{10}(P \text{ value})$  assessing the anti-correlation level of TFs expression level with average DNA methylation level at sites with a given motif. Figure 7

```
load("./ELMER.example/Result/LUSC/getTF.hypo.TFs.with.motif.pvalue.rda")
TF.rank.plot(motif.pvalue=TF.meth.cor, motif="TP53", TF.label=list(TP53=c("TP53", "TP63", "TP73")),
             save=FALSE)
```

```
## $TP53
```

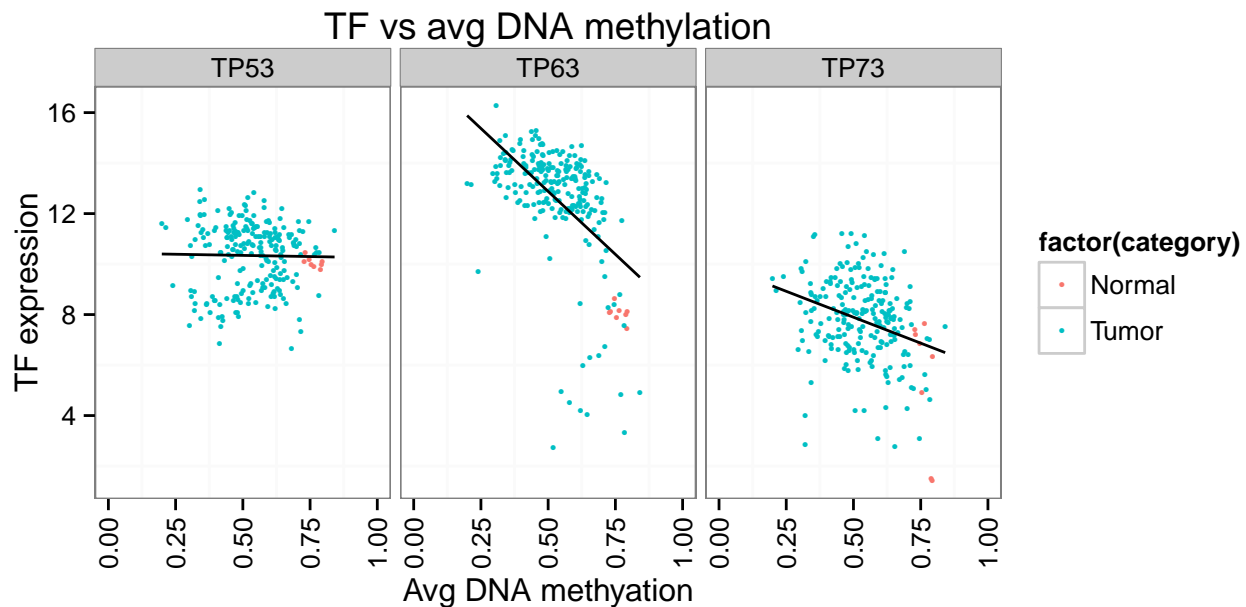


Figure 3: Each scatter plot shows the average methylation level of sites with the TP53 motif in all LUSC samples plotted against the expression of the transcription factor TP53, TP63, TP73 respectively.

```
sessionInfo()

## R version 3.1.3 (2015-03-09)
## Platform: x86_64-apple-darwin14.1.0 (64-bit)
## Running under: OS X 10.10.2 (Yosemite)
##
## 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] stats4      parallel    stats       graphics    grDevices   utils       datasets    methods
## [9] base
##
## other attached packages:
## [1] ELMER_0.99.13
## [2] IlluminaHumanMethylation450kanno.ilmn12.hg19_0.2.1
## [3] minfi_1.12.0
## [4] bumphunter_1.6.0
## [5] locfit_1.5-9.1
## [6] iterators_1.0.7
## [7] foreach_1.4.2
## [8] Biostrings_2.34.1
## [9] XVector_0.6.0
## [10] GenomicRanges_1.18.4
## [11] GenomeInfoDb_1.2.4
## [12] IRanges_2.0.1
## [13] S4Vectors_0.4.0
## [14] lattice_0.20-30
```

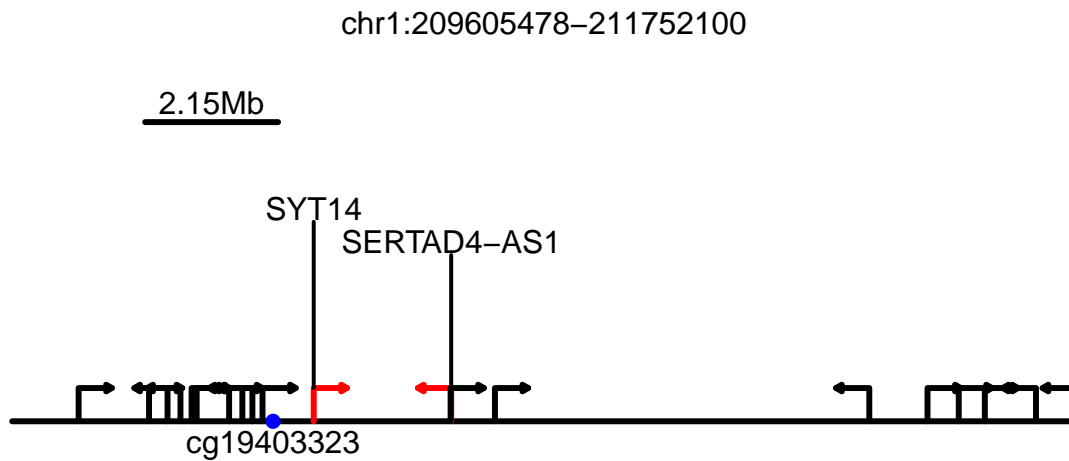


Figure 4: The schematic plot shows probe colored in blue and the location of nearby 20 genes. The genes significantly linked to the probe were shown in red.

```
## [15] Biobase_2.26.0
## [16] BiocGenerics_0.12.1
##
## loaded via a namespace (and not attached):
## [1] annotate_1.44.0      AnnotationDbi_1.28.1 base64_1.1
## [4] beanplot_1.2        BiocStyle_1.4.1     codetools_0.2-11
## [7] colorspace_1.2-6    DBI_0.3.1           digest_0.6.8
## [10] doRNG_1.6           evaluate_0.5.5      formatR_1.0
## [13] genefilter_1.48.1   ggplot2_1.0.1       grid_3.1.3
## [16] gtable_0.1.2        highr_0.4           illuminaio_0.8.0
## [19] knitr_1.9           labeling_0.3         limma_3.22.7
## [22] MASS_7.3-39         matrixStats_0.14.0  mclust_4.4
## [25] multtest_2.22.0     munsell_0.4.2       nlme_3.1-120
## [28] nor1mix_1.2-0       pkgmaker_0.22       plyr_1.8.1
## [31] preprocessCore_1.28.0 proto_0.3-10        quadprog_1.5-5
## [34] RColorBrewer_1.1-2  Rcpp_0.11.5         registry_0.2
## [37] reshape_0.8.5       reshape2_1.4.1      rngtools_1.2.4
## [40] RSQLite_1.0.0       scales_0.2.4        siggenes_1.40.0
## [43] snow_0.3-13         splines_3.1.3       stringr_0.6.2
## [46] survival_2.38-1     tools_3.1.3         XML_3.98-1.1
## [49] xtable_1.7-4        zlibbioc_1.12.0
```

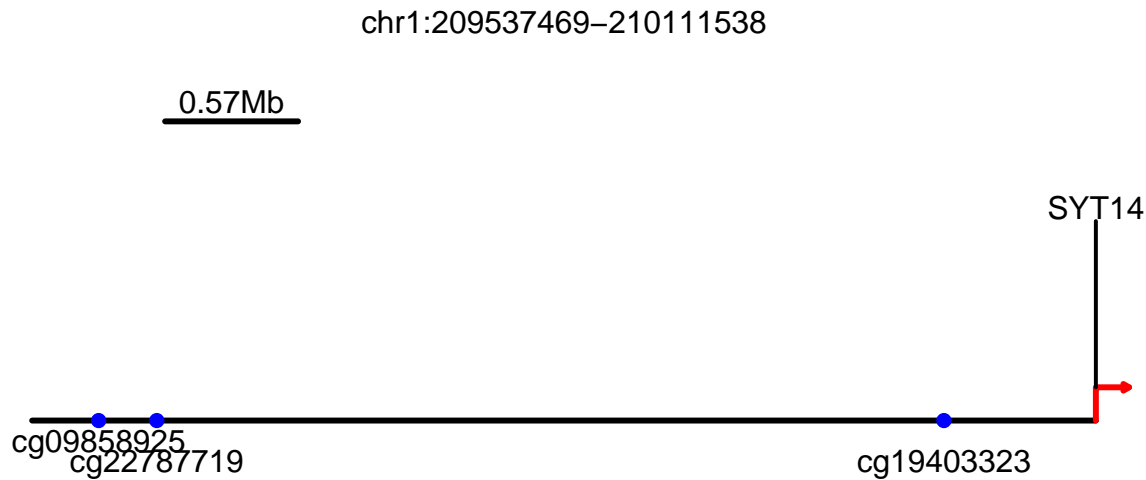


Figure 5: The schematic plot shows the gene colored in red and all blue colored probes, which are significantly linked to the expression of this gene.

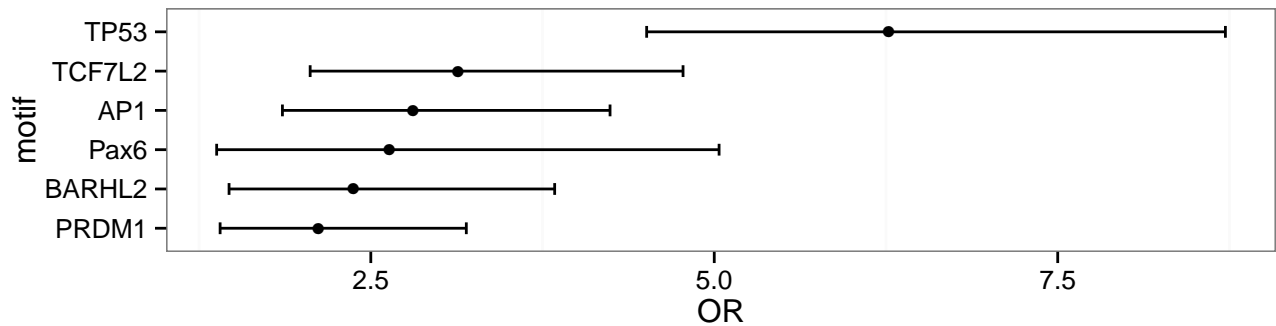


Figure 6: The plot shows the Odds Ratio (x axis) for the selected motifs with OR above 1.3 and lower boundary of OR above 1.3. The range shows the 95% confidence interval for each Odds Ratio.



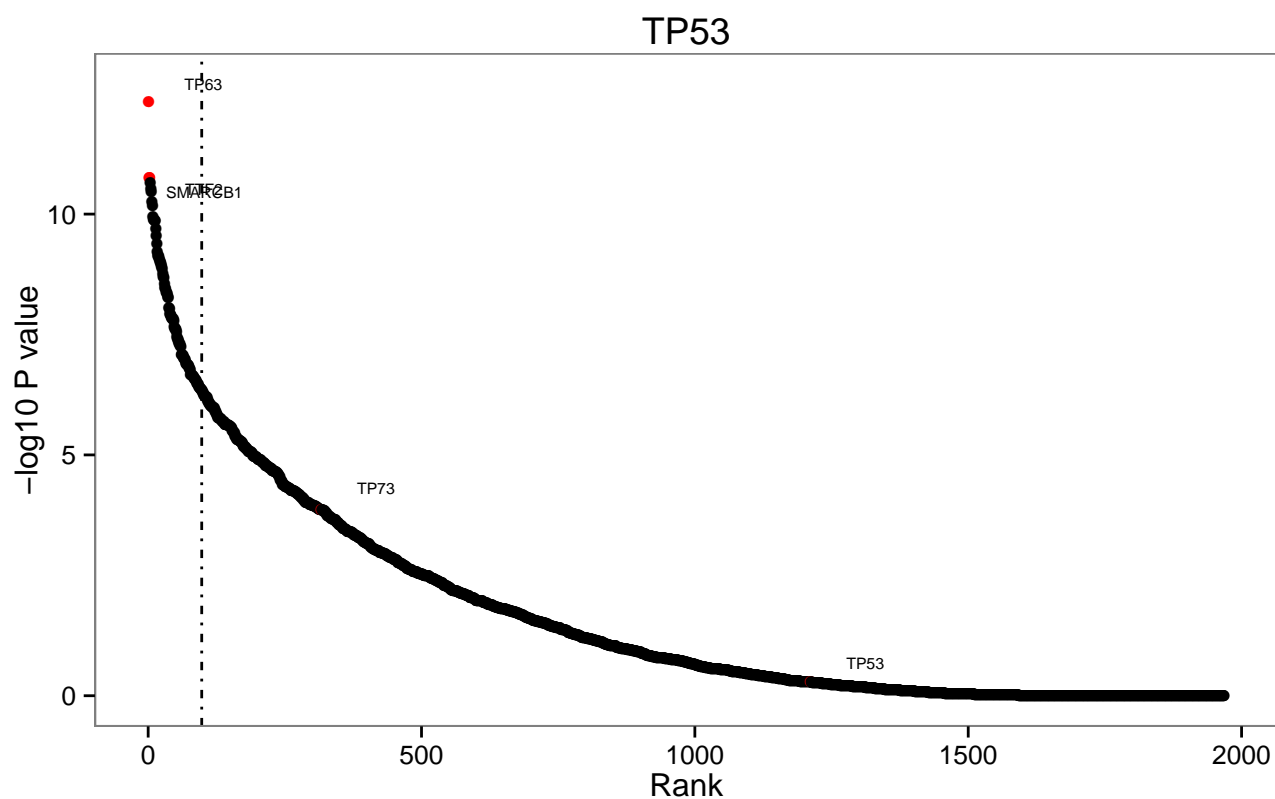


Figure 7: Shown are TF ranking plots based on the score ( $-\log(P \text{ value})$ ) of association between TF expression and DNA methylation of the TP53 motif in the LUSC cancer type. The dashed line indicates the boundary of the top 5% association score. The top 3 associated TFs and the TF family members (dots in red) that are associated with that specific motif are labeled in the plot.