Wavelet Screening applied to DNA methylation

William R.P.Denault et al.

12/01/2021

## Preparing the data

Loading the data and recoding the phenotype from factor to numeric.

library(WaveletScreening)

## Loading required package: parallel

## Loading required package: snow

##   
## Attaching package: 'snow'

## The following objects are masked from 'package:parallel':  
##   
## clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,  
## clusterExport, clusterMap, clusterSplit, makeCluster, parApply,  
## parCapply, parLapply, parRapply, parSapply, splitIndices,  
## stopCluster

## Loading required package: wavethresh

## Loading required package: MASS

## WaveThresh: R wavelet software, release 4.6.8, installed

## Copyright Guy Nason and others 1993-2016

## Note: nlevels has been renamed to nlevelsWT

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.3

## Loading required package: mvtnorm

## Warning: package 'mvtnorm' was built under R version 3.6.3

##   
##   
## WAVELET  
## SCREENING  
##   
## ----------  
## - -  
## - -  
## | -- -- |  
## | | .| | .| |  
## | -- -- |  
## | |  
## | . . |  
## | ---- |  
## - | | -  
## - | | | | -  
## | | | | | | | |  
## | | | ---- | | |  
## | | | | | |  
## | | ---- | |  
## | | | |  
##   
##   
##   
##

load("simulated\_data.Rdata")  
colon.state[1:4]

## [1] normal tumor normal tumor   
## Levels: normal tumor

pheno <- as.numeric(colon.state)-1  
pheno[1:4]

## [1] 0 1 0 1

The data are a compound of two elements: 1) The base pair position of each CpG, 2) The individual CpG level (M value)

temp.pos[1:10]#the base pair position of each CpG

## [1] 212950 212989 213022 213055 213094 213127 213160 213199 213233 213266

length(temp.pos)#total number of CpGs

## [1] 75069

dim(sim\_methy\_all)#lines= CpG level, column individual

## [1] 75069 26

sim\_methy\_all[1:10,]

## [,1] [,2] [,3] [,4] [,5] [,6]  
## [1,] 0.01789069 -0.5979471 -0.26861179 -0.7450518 0.006154367 -0.2316303  
## [2,] -0.12295093 -0.1793961 -0.72841170 -1.3086674 -0.515000732 -0.8252050  
## [3,] -0.60850539 -1.1254358 -1.43378666 -0.2566904 -1.407540420 -1.5996424  
## [4,] -0.13233239 -0.5206948 -0.33446328 -0.9355251 0.378571786 -0.1911608  
## [5,] -0.52356683 -0.1193659 -1.08203326 0.8466020 -0.301928248 -0.4684455  
## [6,] -0.17583869 0.1366015 -0.08845688 0.2800046 -0.349583008 0.1841946  
## [7,] -0.27027873 -0.5149496 -0.98621691 -0.4734249 -0.622979968 -0.8448085  
## [8,] -0.85012038 -1.2079693 -0.80243886 -1.0941048 -0.059525344 -0.7331130  
## [9,] -0.63962465 -0.5387749 0.11230321 -1.0197601 0.155452984 0.4797858  
## [10,] -0.38126215 -0.5939619 -1.74599669 -0.2232314 -1.414829111 -1.2352218  
## [,7] [,8] [,9] [,10] [,11] [,12]  
## [1,] -1.01687372 1.6369072 1.29521677 0.09090859 -0.7362546 0.568777789  
## [2,] -1.56503759 -0.6358605 -0.54430514 -0.25168886 -0.6505482 -0.228435296  
## [3,] -1.26149751 -2.1215118 -0.95027779 0.29285588 -1.6455165 -1.072475068  
## [4,] -1.41769035 0.7114587 0.48849978 0.05401856 -0.4647886 0.687934467  
## [5,] -0.56899677 -1.1046892 -0.62454157 0.01643283 -0.5484838 -0.500839924  
## [6,] -0.09056637 -0.3646430 -0.05112869 0.02459357 0.4292049 -0.175917892  
## [7,] -0.87285199 -0.6049488 -0.70557733 -0.11578385 -0.5465851 -0.005313975  
## [8,] -1.83983349 0.3766095 -0.04748010 -0.73728525 -1.2546551 -0.044575542  
## [9,] -0.93833347 -0.2119520 -0.65619931 -0.38297017 -0.4674312 0.122974041  
## [10,] -1.78175983 0.4887372 0.13874454 -0.69501087 -1.1251894 -0.907240611  
## [,13] [,14] [,15] [,16] [,17] [,18]  
## [1,] 1.2214266 0.3094245 -1.2098631 0.25370798 1.21585292 0.58778925  
## [2,] 0.5364375 -0.3858237 -0.6960881 0.28675925 -0.05952001 0.07478925  
## [3,] -0.9592571 -2.1566811 -1.0850309 -0.44432612 -0.74814482 -0.71138026  
## [4,] 0.6022066 0.1325970 -0.3215502 -0.45979378 -0.10102253 -0.17319446  
## [5,] -0.5828068 -1.4841920 -0.7454910 0.19456725 -0.81984088 -0.12222264  
## [6,] 0.2634174 0.5321023 0.0494421 0.08552505 0.15979661 0.60939812  
## [7,] 0.4065701 -0.1772014 -0.4156729 0.19408731 0.06841280 0.07764693  
## [8,] 0.3057424 0.1136516 -0.0663459 -0.30617165 0.26984676 0.13134646  
## [9,] 0.5872839 0.7095553 0.1274825 -0.06578879 -0.42908901 -0.08656222  
## [10,] -0.3546422 0.1036027 -0.3450238 -0.47024317 0.11120745 -0.62208133  
## [,19] [,20] [,21] [,22] [,23] [,24]  
## [1,] 1.35841308 0.8473610 0.87437071 -0.2797963 1.0240464 -0.0474233  
## [2,] 0.97049115 0.3249609 -0.14503151 0.3692742 0.5834905 -0.1544694  
## [3,] 0.56541391 1.1907114 0.67110972 -0.7161315 0.5871834 -0.2251210  
## [4,] 1.21976445 0.9521819 0.28127674 0.2228654 0.8944737 0.4141170  
## [5,] 0.31635396 0.1429107 -0.70967252 -0.9236785 0.2908476 -1.2761696  
## [6,] -0.24168640 0.1983878 -0.08392748 -0.2183991 0.3677756 -0.4101177  
## [7,] 0.13559124 0.2970654 -0.33136362 -0.3886523 0.7319181 -0.3486615  
## [8,] 0.76040412 0.1583277 -0.73506247 -0.3577664 -0.2540041 -0.2940470  
## [9,] 0.01964559 -0.4792784 -0.77114336 0.1333364 0.0914977 -0.4515665  
## [10,] 0.54613471 0.9843494 0.06375385 -1.2929332 1.6758987 -0.5591644  
## [,25] [,26]  
## [1,] -0.75230347 -0.43835517  
## [2,] -0.05120867 -0.13882907  
## [3,] -0.44052168 -0.37305521  
## [4,] 0.81779179 0.27872728  
## [5,] -0.23627018 -1.10941609  
## [6,] 0.19845081 -0.40186991  
## [7,] 0.42894590 -0.56942409  
## [8,] 0.51154905 0.09127244  
## [9,] 0.34227517 -0.45180062  
## [10,] 0.84985374 0.37805003

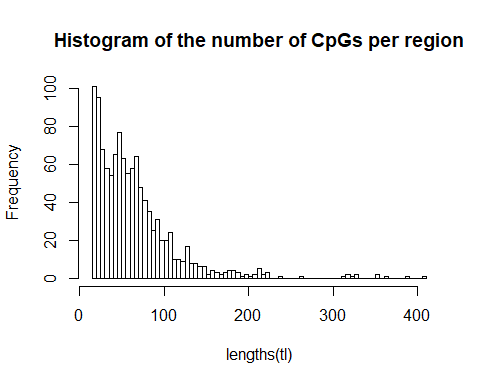
### Defining slice

Here, we divide the data into regions that have at least 17 CpGs, separated by a maximum distance of 500 bp. This results in a missingness rate of 1.89%.

thresh <- 500  
tl <- split(temp.pos , cumsum(c(1, diff(temp.pos) > thresh) ) ) #Remove cases that are too far apart  
  
#Level of analysis 4  
tl <- tl[-which(lengths(tl)< 17)]#List of the positions of CpGs that we will analyze later  
sum(lengths(tl))/length( temp.pos)#percent missingness

## [1] 0.981124

hist( lengths(tl),nclass = 100, main="Histogram of the number of CpGs per region")



## Running Wavelet Screening region by region

In this section, we run Wavelet Screening on each region separately. This takes about 3 to 4 minutes on a single CPU.

library(WaveletScreening)  
tt <- proc.time()  
res <- list()  
for ( i in 1 :length(tl))  
{  
 reg\_mat <- sim\_methy\_all[which( temp.pos %in% tl[[i]]),] #Select CpGs in region i  
 bp <- temp.pos[which( temp.pos %in% tl[[i]])]#Correspond to the base position of the CpG in region i  
 res[[i]] <- Wavelet\_screening( Y=pheno,  
 loci=reg\_mat,  
 lev\_res=4,  
 coeftype = "c",  
 bp = bp,  
 base\_shrink = 1/26,  
 sigma\_b =200,  
 verbose = FALSE  
 )  
  
  
}  
  
  
proc.time() -tt #total run time

## user system elapsed   
## 49.93 1.94 51.87

We concatenate the results of Wavelet Screening into one data frame.

res <- do.call( rbind, res)  
res[1:10,]

## L\_h min\_ph\_pv Beta\_0\_0 Beta\_1\_1 Beta\_1\_2 Beta\_2\_1  
## [1,] -10.892168 0.00000000 -0.102042450 0.04501653 -0.258261702 -0.007768737  
## [2,] -18.604068 0.00000000 -0.065289572 -0.09134897 0.008579152 -0.072190861  
## [3,] -13.846078 0.00000000 0.002647418 -0.12631846 -0.013546850 -0.116194972  
## [4,] -1.198495 0.56447538 -1.187995204 -0.12056095 -1.066627320 -0.056044266  
## [5,] -2.143019 0.50878409 0.442588878 0.05376592 1.349828866 -0.044537698  
## [6,] -13.089435 0.00000000 0.021123125 0.17531647 0.042197850 0.158239504  
## [7,] -13.568792 0.00000000 0.014476640 -0.09048563 0.062370918 0.079588710  
## [8,] -13.849895 0.00000000 0.029041037 0.06711068 -0.035738053 0.037775293  
## [9,] -14.853967 0.00057376 0.050233649 0.06462273 -0.060672061 0.013432911  
## [10,] -17.021486 0.00000000 -0.033038633 -0.02648544 -0.133385878 0.164956585  
## Beta\_2\_2 Beta\_2\_3 Beta\_2\_4 Beta\_3\_1 Beta\_3\_2  
## [1,] 0.088332075 -0.15203469 -0.20503552 -0.092637647 -0.117277337  
## [2,] -0.137122155 0.03608488 0.01026217 -0.023809994 -0.170441860  
## [3,] -0.016480948 -0.09295459 -0.08253614 -0.172936285 -0.071946969  
## [4,] -0.002380679 -0.20000163 -1.42509364 -0.016128843 0.022366887  
## [5,] 0.042817339 0.63393221 1.42795535 0.080362565 -0.001589849  
## [6,] 0.079666124 -0.04982345 0.14564684 0.032341217 0.120852594  
## [7,] -0.065703523 0.08324567 0.17594130 0.014029584 0.061469274  
## [8,] -0.066585843 -0.04808553 -0.08508384 0.056929503 -0.082214380  
## [9,] 0.134344867 0.18450133 0.01217245 0.017206579 -0.024305417  
## [10,] -0.157588803 -0.09290005 0.03708296 -0.009319208 -0.034280441  
## Beta\_3\_3 Beta\_3\_4 Beta\_3\_5 Beta\_3\_6 Beta\_3\_7  
## [1,] 0.084547354 -0.05320393 0.18384288 -0.005326181 -0.208057402  
## [2,] -0.006136308 -0.02454137 -0.07730636 0.098099846 -0.038548160  
## [3,] -0.038021643 -0.11308398 -0.03113610 -0.065787316 -0.283247728  
## [4,] 0.021937342 -0.04483561 -0.04140766 -0.377358581 -1.268700517  
## [5,] 0.045760578 0.04434851 -0.13523566 0.903318165 1.235070571  
## [6,] 0.141281447 0.06853662 -0.11795842 0.060423955 0.136656581  
## [7,] -0.050443223 0.12435722 0.07777224 -0.150042380 0.094573416  
## [8,] 0.114591294 -0.24244020 0.05692418 -0.023193496 -0.107334018  
## [9,] -0.069862815 0.14286652 0.07817662 0.105112064 -0.001078996  
## [10,] -0.050858478 -0.05984341 0.02482959 -0.006167439 0.033004111  
## Beta\_3\_8 Beta\_4\_1 Beta\_4\_2 Beta\_4\_3 Beta\_4\_4  
## [1,] -0.07461941 -0.01464610 -0.16820349 -0.053142068 -0.014245945  
## [2,] 0.13609893 -0.02163298 -0.02921719 -0.082472762 -0.046653266  
## [3,] -0.02194699 -0.13088182 -0.23126059 0.036338308 -0.100955799  
## [4,] -1.48859546 0.01646537 -0.02728531 0.006489706 0.013225125  
## [5,] 0.47163535 0.05538576 0.04030405 0.036007704 0.005819308  
## [6,] 0.10358619 0.06303715 0.02529086 -0.044374998 0.192392254  
## [7,] 0.18092008 0.25941078 -0.12363889 -0.038197795 0.171832268  
## [8,] -0.16726898 0.15307352 0.07797846 -0.075562471 0.062965252  
## [9,] 0.06253307 -0.05764775 -0.16447445 -0.024459971 0.235646923  
## [10,] 0.18642920 -0.05719605 0.06974371 0.042365751 -0.099600599  
## Beta\_4\_5 Beta\_4\_6 Beta\_4\_7 Beta\_4\_8 Beta\_4\_9  
## [1,] -0.0006937623 -0.005510857 0.18686177 -0.126395317 -0.005538995  
## [2,] -0.0749148909 -0.074093580 -0.02715454 0.094663118 -0.067051441  
## [3,] -0.0407742892 -0.008943335 -0.17677823 -0.173849257 -0.014064267  
## [4,] 0.0239572853 0.014879601 0.07807602 -0.056631896 -0.133261729  
## [5,] -0.0822104062 -0.029329917 0.15800747 -0.035028643 -0.002949024  
## [6,] 0.0243976796 0.205666413 0.04932730 0.141013721 -0.038574644  
## [7,] 0.0150905159 -0.206363078 0.03097133 -0.002041579 0.092776620  
## [8,] 0.2084587429 0.136766469 -0.16618183 -0.202412165 -0.121512473  
## [9,] -0.2821275317 0.051720461 0.05437987 -0.009138282 0.086415335  
## [10,] -0.0320992846 -0.021053570 -0.08497257 -0.110306688 -0.204373012  
## Beta\_4\_10 Beta\_4\_11 Beta\_4\_12 Beta\_4\_13 Beta\_4\_14  
## [1,] 0.03842183 0.01011882 -0.073560826 -0.160486951 -0.22410702  
## [2,] -0.14921376 -0.01261691 0.022529673 -0.047425253 -0.07255507  
## [3,] 0.05984873 0.09377850 0.017680163 -0.097109513 -0.14995590  
## [4,] 0.02458634 -0.21927567 -0.484112696 -0.903645054 -1.44744974  
## [5,] -0.02700205 0.10679908 1.044407877 1.141422916 1.37712678  
## [6,] 0.09098012 -0.12242111 0.079736246 0.086830260 0.09820484  
## [7,] 0.08073425 0.11650759 -0.104927683 0.171880322 0.03350407  
## [8,] -0.04331760 -0.07934445 -0.016353456 -0.107880039 -0.09184273  
## [9,] 0.05692418 0.16781793 0.120539783 -0.005819308 -0.03328204  
## [10,] -0.03541582 0.03717163 0.006855928 -0.007559164 0.05408091  
## Beta\_4\_15 Beta\_4\_16 Pi\_0\_0 Pi\_1\_1 Pi\_1\_2 Pi\_2\_1 Pi\_2\_2  
## [1,] -0.070337279 -0.040200533 0.000000 0 0.0000000 0 0  
## [2,] -0.011745857 -0.005583968 0.000000 0 0.0000000 0 0  
## [3,] -0.135556963 -0.003208116 0.000000 0 0.0000000 0 0  
## [4,] -1.312412158 -1.372184966 0.568959 0 0.4468279 0 0  
## [5,] 0.618535582 -0.056229201 0.000000 0 0.6804662 0 0  
## [6,] 0.094028079 0.103586194 0.000000 0 0.0000000 0 0  
## [7,] -0.075361652 0.014599394 0.000000 0 0.0000000 0 0  
## [8,] -0.039797386 -0.227857196 0.000000 0 0.0000000 0 0  
## [9,] 0.009215396 0.006480055 0.000000 0 0.0000000 0 0  
## [10,] -0.036566646 0.019731506 0.000000 0 0.0000000 0 0  
## Pi\_2\_3 Pi\_2\_4 Pi\_3\_1 Pi\_3\_2 Pi\_3\_3 Pi\_3\_4 Pi\_3\_5 Pi\_3\_6  
## [1,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [2,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [3,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [4,] 0.00000000 0.8098353 0 0 0 0 0 0.0266292  
## [5,] 0.06078344 0.7658341 0 0 0 0 0 0.2130935  
## [6,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [7,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [8,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [9,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## [10,] 0.00000000 0.0000000 0 0 0 0 0 0.0000000  
## Pi\_3\_7 Pi\_3\_8 Pi\_4\_1 Pi\_4\_2 Pi\_4\_3 Pi\_4\_4 Pi\_4\_5 Pi\_4\_6  
## [1,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## [2,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## [3,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## [4,] 0.6884904 0.85590029 0 0 0 0 0.000000000 0  
## [5,] 0.5703146 0.02934114 0 0 0 0 0.000000000 0  
## [6,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## [7,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## [8,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## [9,] 0.0000000 0.00000000 0 0 0 0 0.009180161 0  
## [10,] 0.0000000 0.00000000 0 0 0 0 0.000000000 0  
## Pi\_4\_7 Pi\_4\_8 Pi\_4\_9 Pi\_4\_10 Pi\_4\_11 Pi\_4\_12 Pi\_4\_13  
## [1,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [2,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [3,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [4,] 0.000000000 0 0.004389187 0 0.011358931 0.05562903 0.2972256  
## [5,] 0.002818485 0 0.000000000 0 0.000307418 0.35155514 0.4628667  
## [6,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [7,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [8,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [9,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## [10,] 0.000000000 0 0.000000000 0 0.000000000 0.00000000 0.0000000  
## Pi\_4\_14 Pi\_4\_15 Pi\_4\_16 null\_sd\_start\_EM  
## [1,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [2,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [3,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [4,] 0.8377691 0.73526969 0.7853552 0.3922313  
## [5,] 0.7327671 0.06952741 0.0000000 0.3922313  
## [6,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [7,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [8,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [9,] 0.0000000 0.00000000 0.0000000 0.3922313  
## [10,] 0.0000000 0.00000000 0.0000000 0.3922313

## Simulation of the null distribution

To assess the significance of each region, we simulate the null distribution of the test statistic. Below we display how to simulate this distribution.

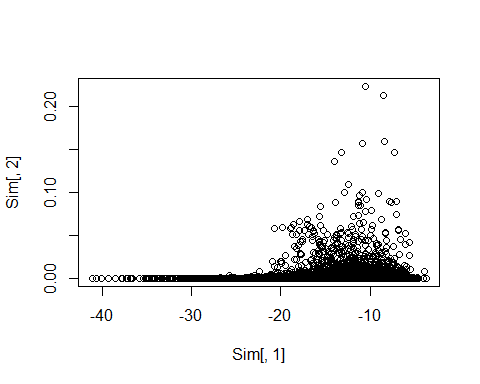
set.seed(1)  
#Simulation function using the same parameter as used in the Wavelet\_screening function  
tt <- proc.time()  
Sim <- Simu\_null\_emp(res=res,  
 coeftype="c",  
 lev\_res = 4,  
 base\_shrink = 1/26,  
 size=100000#Number of simulations required  
)

## [1] "Simulation of test statistics"  
## [1] "10000 simulations performed"  
## [1] "20000 simulations performed"  
## [1] "30000 simulations performed"  
## [1] "40000 simulations performed"  
## [1] "50000 simulations performed"  
## [1] "60000 simulations performed"  
## [1] "70000 simulations performed"  
## [1] "80000 simulations performed"  
## [1] "90000 simulations performed"  
## [1] "1e+05 simulations performed"

proc.time() -tt #total running time

## user system elapsed   
## 42.11 0.78 42.93

plot( Sim[,1], Sim[,2])



## Performing the Box-Cox transform

As explained in the article, the distribution of Lh might not be Gaussian when analyzing data using low depth. Below, we display how to perform a Box-Cox transform on the simulations of the test statistic under the H~0.

Here, we have an additionnal issue: the observed test statistics contains some positive and some negative values. However the Box-Cox transform requires only positive values. We therefore shift the distribution and then apply the Box-Cox transform.

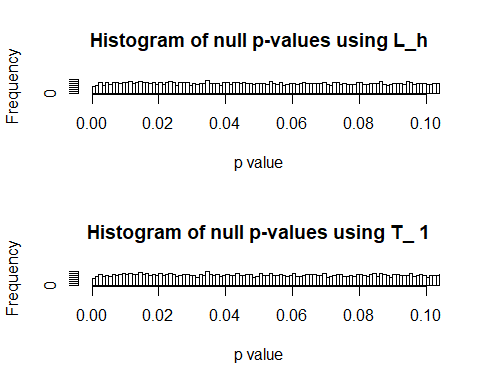
library(EnvStats)  
shifting <- max( 0,1.0001\*max( Sim[,1],res[,1]))  
bc\_lambda <- boxcox(-Sim[,1] + shifting, optimize = TRUE)  
Sim[,1] <- - ( ( (-Sim[,1] + shifting)^bc\_lambda$lambda) -1 )/bc\_lambda$lambda  
res[,1] <- - ( ( (-res[,1] + shifting)^bc\_lambda$lambda) -1 )/bc\_lambda$lambda

## Computing the p-values

When the Box-Cox transform is done, we can then compute the p-value of each region as follows.

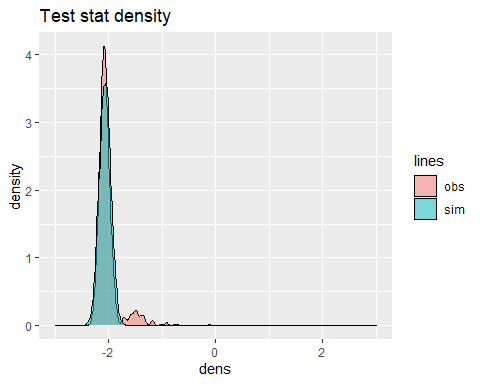
lambda <- Search\_lambda(Sim,plot=TRUE)#optimizing the lambda\* value

## [1] "Rough search"  
## [1] "Dichotomized search"



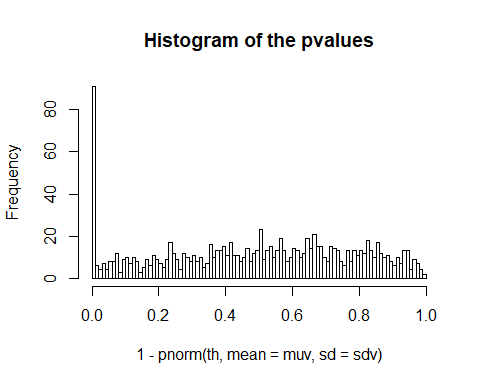
#lambda <- 15  
Th <- Sim[,c("L\_h")]+lambda\*Sim[,c("min\_ph\_pv")] #Computing the null distribution of the test statistic   
muv <- median(Th,na.rm = TRUE) #estimating the parameter of the null distribution  
sdv <- mad(Th,na.rm = TRUE)  
####################################  
##Test value of the loci to be tested  
####################################  
th <- res[,1]+lambda\*res[,2]#Computing the test statistic  
  
##########################  
##Plot of two distributions  
##########################  
dat <- data.frame(dens = c(c(th),c(Th[1:2000]))#To have the same size on the plot  
 , lines = c(rep("obs", length(c(th))), rep( "sim", length(c(Th[1:2000]))) ))  
  
ggplot(dat, aes(x = dens, fill = lines)) +  
 geom\_density(alpha = 0.5)+  
 xlim(c(-3,3))+#Some of the associated regions have test statistics very far from the null; this results in poor visualization if they are not excluded.  
 ggtitle("Test stat density")

## Warning: Removed 1 rows containing non-finite values (stat\_density).

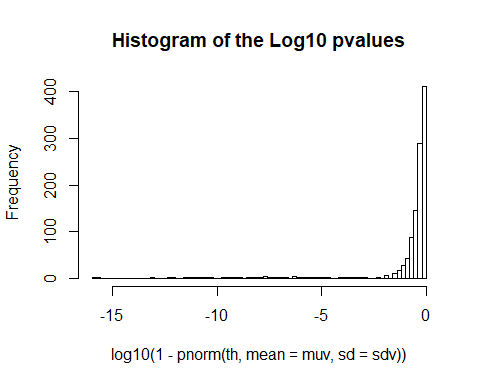


Here, we see the null distribution of the test statistic, and the associated regions on the right side of the plot.

#Computing the p-value  
pv <- 1-pnorm(th,mean=muv,sd=sdv)  
hist(1-pnorm(th,mean=muv,sd=sdv),nclass=100, main="Histogram of the pvalues")



hist(log10(1-pnorm(th,mean=muv,sd=sdv)),nclass=100,main ="Histogram of the Log10 pvalues")



## Loading data from Irizzary et al. 2009

The data can be found at <https://www.nature.com/articles/ng.298> as supplementary material.

library(readxl)  
tt <- read\_excel("41588\_2009\_BFng298\_MOESM18\_ESM.xls")  
sub <- tt[which(tt[,2]=="chr3"),]#Only selecting Chromosome 3  
sub <- as.data.frame(sub)  
head(sub) #data frame with the start and end positions of each DMR we are trying to detect.

## ...1 chr start end delta M colon.tumor colon.normal fdr  
## 1 44705 chr3 193607217 193607552 -0.9357281 0.8858763 1.8981316 0.00e+00  
## 2 45816 chr3 82629728 82630444 -0.8377335 0.9777084 1.7687336 0.00e+00  
## 3 43646 chr3 129687777 129688211 0.7653712 1.0458169 0.2482736 1.99e-09  
## 4 44692 chr3 191521264 191521946 -0.7363103 0.3052087 0.9414630 2.29e-09  
## 5 44954 chr3 28591265 28591843 0.6876208 0.8209596 0.1174106 6.39e-08  
## 6 43642 chr3 129686645 129687266 0.6589586 1.6160071 0.9675025 3.87e-07  
## state name annotation region distance island distToIsla  
## 1 Less methylation FGF12 NM\_021032 inside 1153 Far 960  
## 2 Less methylation GBE1 NM\_000158 upstream 736294 Shore 309227  
## 3 Some methylation GATA2 NM\_032638 inside 6506 Shore 0  
## 4 No methylation CLDN1 NM\_021101 inside 962 Far 562  
## 5 Some methylation ZCWPW2 NM\_001040432 upstream 49632 Shore 0  
## 6 More methylation GATA2 NM\_032638 inside 7451 Shore 916

lstemp <- list() #  
  
for (i in 1: dim(sub)[1])  
{  
   
 lstemp[[i]] <- temp.pos[ which( (sub$start[i]-1) < temp.pos & temp.pos < (sub$end[i] +1 ))]  
   
}  
  
sum( lengths(lstemp)) # number of CpGs to detect

## [1] 1901

pos\_to\_detect <- do.call( c,lstemp)#List of the CpGs to detect

## Performing a slicing

Ensure the same slicing as before.

thresh <- 500  
  
tl <- split(temp.pos , cumsum(c(1, diff(temp.pos) > thresh) ) ) #Remove cases that are too far apart  
tl <- tl[-which(lengths(tl) <17)]#Keep only long-enough sequences of CPGs  
  
pos\_to\_detect <- pos\_to\_detect [which(pos\_to\_detect %in% do.call( c , tl))] #filtering out the CpGs when slicing

## Power

If a region contains more than one CpG that is differentially methylated, then it has to be detected.

#Number of CpGs per regions  
n\_CpG\_per\_region <- list()  
  
for ( i in 1:length(tl))  
{  
 n\_CpG\_per\_region[[i]] <- pos\_to\_detect[which(pos\_to\_detect %in% tl[[i]])]  
  
}  
  
  
nCpG\_reg <- lengths(n\_CpG\_per\_region)  
indx\_to\_detect <- which( nCpG\_reg >0)#List of the regions to detect  
indx\_to\_detect

## [1] 4 5 14 18 21 56 58 80 100 120 130 134 139 140 143  
## [16] 157 168 189 190 221 232 249 327 337 375 413 416 417 448 479  
## [31] 508 520 521 527 587 610 614 642 647 670 676 677 689 725 731  
## [46] 741 748 760 771 774 775 777 778 795 801 803 804 806 814 817  
## [61] 821 822 832 849 858 859 878 879 900 914 915 935 939 941 947  
## [76] 977 995 997 1007 1008 1014 1019 1020 1021 1022 1024 1025 1045 1066

Regions detected

type <- c("pv", "pv", "fdr", "fdr")  
threshold <- c("e-5", "e-6", "5%", "1%")  
detected\_reg <- c()  
tru\_reg <- c()  
#Regions detected  
  
#When using p-values 10^-5  
indx\_WS <- which(pv < 10^(-5))  
detected\_reg <- c( detected\_reg ,length(indx\_WS))  
tru\_reg <- c( tru\_reg , length(which(indx\_WS %in% indx\_to\_detect) ) )  
  
#When using p-values 10^-6  
indx\_WS <- which(pv < 10^(-6))  
detected\_reg <- c( detected\_reg ,length(indx\_WS))  
tru\_reg <- c( tru\_reg , length(which(indx\_WS %in% indx\_to\_detect) ) )  
fdr\_regions <- p.adjust(pv, method ="BH")  
  
#When using FDR at 5%  
indx\_WS <- which(fdr\_regions <0.05)  
detected\_reg <- c( detected\_reg ,length(indx\_WS))  
tru\_reg <- c( tru\_reg , length(which(indx\_WS %in% indx\_to\_detect) ) )  
  
#When using FDR at 1%  
indx\_WS <- which(fdr\_regions <0.01)  
detected\_reg <- c( detected\_reg ,length(indx\_WS))  
tru\_reg <- c( tru\_reg , length(which(indx\_WS %in% indx\_to\_detect) ) )  
  
#Summary   
df\_out <- data.frame(type=type, threshold = threshold, nb\_detected\_region= detected\_reg, nb\_true\_regions= tru\_reg )

df\_out

## type threshold nb\_detected\_region nb\_true\_regions  
## 1 pv e-5 74 74  
## 2 pv e-6 63 63  
## 3 fdr 5% 90 88  
## 4 fdr 1% 86 84

df\_out summarizes the number of regions detected using different criteria and the number of truly associated regions.