smyth_goldstandard

John Stansfield

December 31, 2017

set up

```
library(HiCcompare)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(InteractionSet)
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: stats4
```

```
## 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 objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, cbind, colMeans,
##     colnames, colSums, do.call, duplicated, eval, evalq, Filter,
##     Find, get, grep, grepl, intersect, is.unsorted, lapply,
##     lengths, Map, mapply, match, mget, order, paste, pmax,
##     pmax.int, pmin, pmin.int, Position, rank, rbind, Reduce,
##     rowMeans, rownames, rowSums, sapply, setdiff, sort, table,
##     tapply, union, unique, unsplit, which, which.max, which.min

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:data.table':
##
##     first, second

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:data.table':
##
##     shift

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

## Loading required package: GenomeInfoDb

## Loading required package: SummarizedExperiment

## 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: DelayedArray

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians
```

```
## The following object is masked from 'package:dplyr':
##
##     count
##
## Attaching package: 'DelayedArray'
## The following objects are masked from 'package:matrixStats':
##
##     colMaxs, colMins, colRanges, rowMaxs, rowMins, rowRanges
## The following object is masked from 'package:base':
##
##     apply
```

```
library(MDmisc)
library(ggplot2)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:Biobase':
##
##     combine
## The following object is masked from 'package:BiocGenerics':
##
##     combine
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following objects are masked from 'package:IRanges':
##
##     cov, var
## The following objects are masked from 'package:S4Vectors':
##
##     cov, var
## The following object is masked from 'package:BiocGenerics':
##
##     var
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
options(stringsAsFactors = FALSE)
```

```
dataframe2interactionset <- function(df) {
  df <- as.data.frame(df)
```

```
  set1 <- GRanges(df$chr1, IRanges(start = df$start1, end = df$end1))
  set2 <- GRanges(df$chr2, IRanges(start = df$start2, end = df$end2))
  gi <- GInteractions(set1, set2)
  if (ncol(df) > 6) {
    S4Vectors::values(gi) <- cbind(S4Vectors::values(gi), df[, 7:ncol(df)])
  }
  return(gi)
}
```

read in lib1 data

```
# core_path <- "C:/VM_Shared/DNA_int_analysis/rickman/rickman_lib1/"
# man_path <- "C:/VM_Shared/DNA_int_analysis/manuscript/prostate_analysis/"

core_path <- "D:/3D_DNA/rickman/rickman_lib1/"
man_path <- "D:/3D_DNA/manuscript/prostate_analysis/"

chrs <- paste0('chr', 1:22)
chrs <- c(chrs, 'chrX')

# read in data for 1 mb
ERG_lib1 <- list()
GFP_lib1 <- list()
for(i in 1:22) {
  ERG_lib1[[i]] <- read.table(paste0(core_path, 'ERG_lib1_chr', i, '.1mb.txt'), header = FALSE)
  GFP_lib1[[i]] <- read.table(paste0(core_path, 'GFP_lib1_chr', i, '.1mb.txt'), header = FALSE)
}
ERG_lib1[[23]] <- read.table(paste0(core_path, 'ERG_lib1_chr', 'X', '.1mb.txt'), header = FALSE)
GFP_lib1[[23]] <- read.table(paste0(core_path, 'GFP_lib1_chr', 'X', '.1mb.txt'), header = FALSE)
```

Run hiccompare on data

```
lib1_tables <- mapply(create.hic.table, ERG_lib1, GFP_lib1, chrs, SIMPLIFY = FALSE, MoreArgs = list(scal
                                                                                                      # e
                                                                                                      # e



lib1_tables <- total_sum(lib1_tables)
```

```
## Scaling factor = 2.27073638370632 Total counts IF1: 2376234 Total counts IF2: 5395801
```

```
## chr10 may have a Copy Number Variation. Check the MD plots.
```

```
## chr13 may have a Copy Number Variation. Check the MD plots.
```

```
# pdf(file = paste0(man_path, 'new_data.pdf'))
lib1_tables <- hic_loess(lib1_tables, Plot = FALSE, span = NA)
```

```
## Span for loess: 0.184980512781303
```

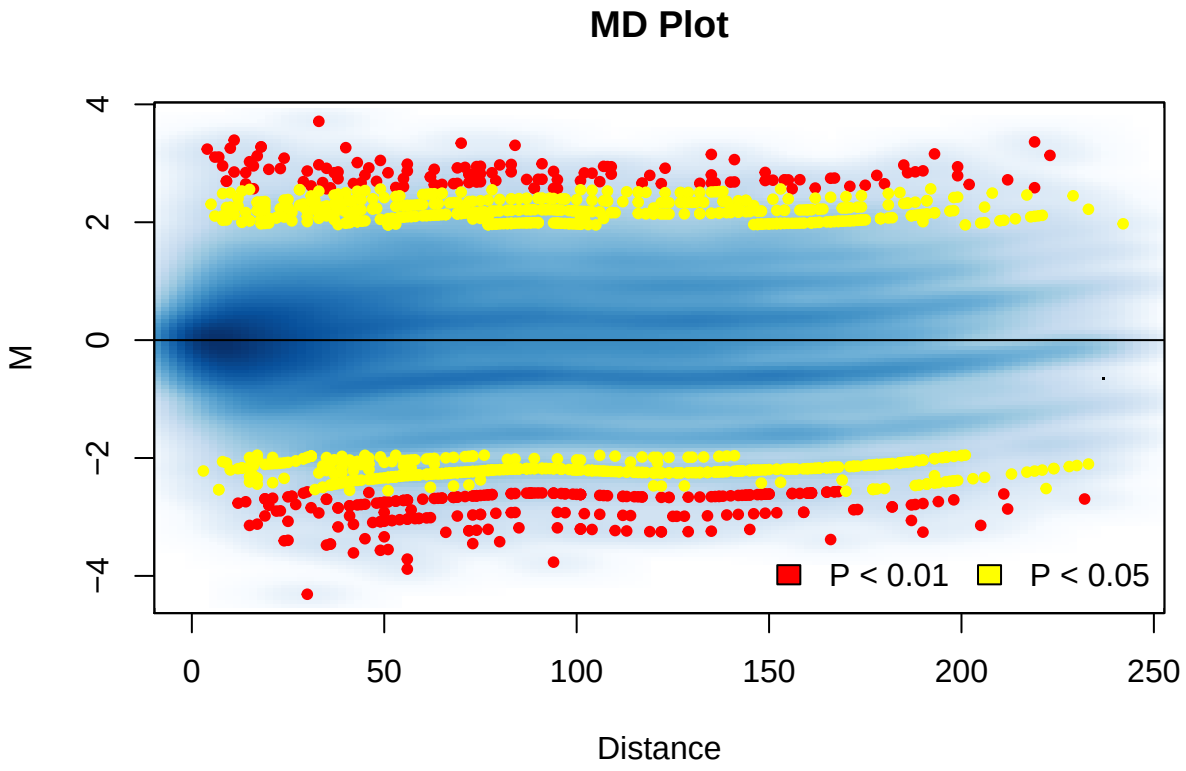
```
## GCV for loess: 4.72818269019694e-05
```

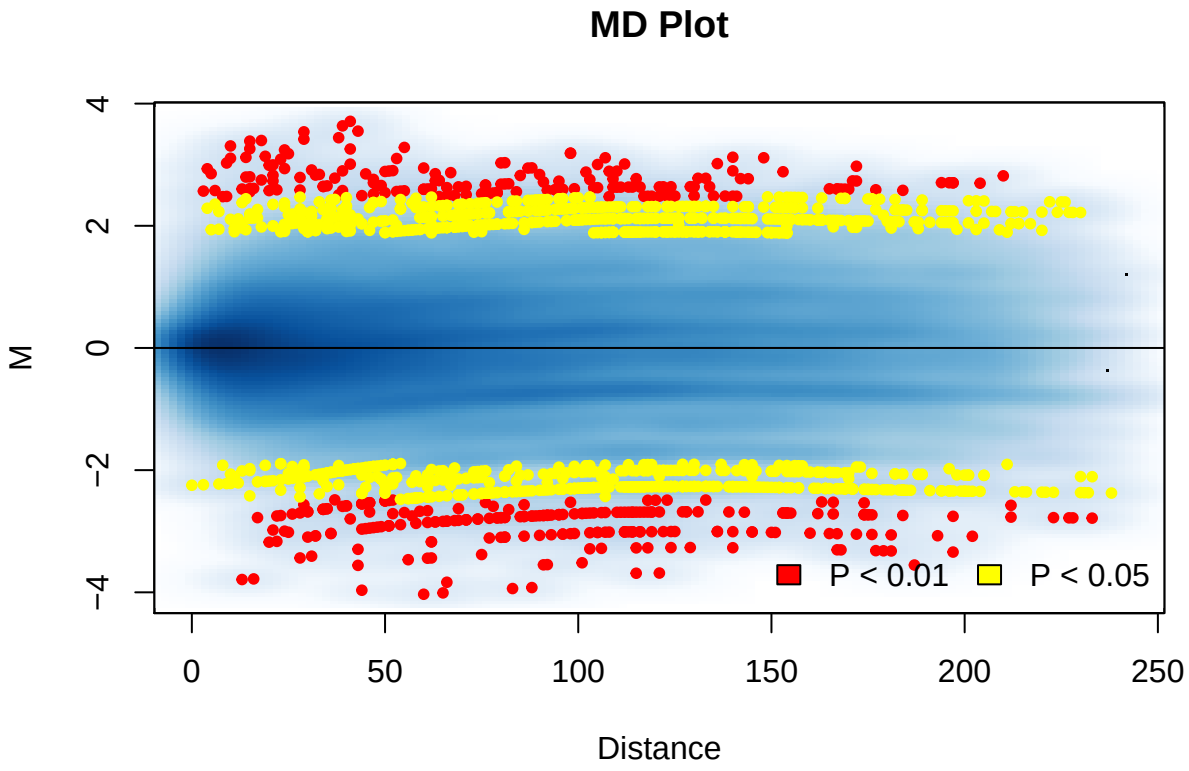
```
## AIC for loess: 0.994311450467231
```

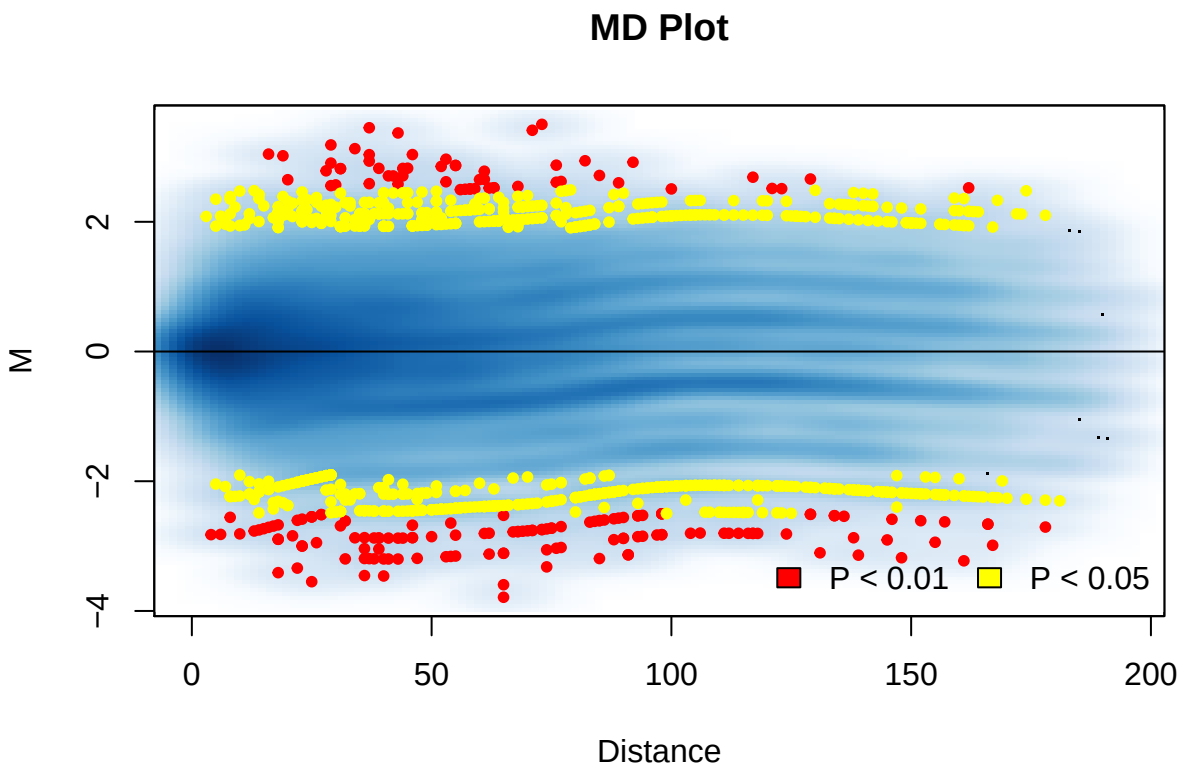
```
## Span for loess: 0.330991350932233
## GCV for loess: 3.58142901142785e-05
## AIC for loess: 0.923107022368063
## Span for loess: 0.217170055865213
## GCV for loess: 5.93747469420197e-05
## AIC for loess: 0.940625144707703
## Span for loess: 0.622326051044541
## GCV for loess: 5.23630661715972e-05
## AIC for loess: 0.814881266863935
## Span for loess: 0.222361697574584
## GCV for loess: 4.84451868144378e-05
## AIC for loess: 0.693829143903295
## Span for loess: 0.245637261272325
## GCV for loess: 6.55476417015176e-05
## AIC for loess: 0.873401374689647
## Span for loess: 0.384641244253762
## GCV for loess: 9.54362483960991e-05
## AIC for loess: 0.958520124253104
## Span for loess: 0.188984494596645
## GCV for loess: 9.0139821392008e-05
## AIC for loess: 0.808710261961436
## Span for loess: 0.168276627027376
## GCV for loess: 0.000139837884112671
## AIC for loess: 0.738081023186783
## Span for loess: 0.319654309512471
## GCV for loess: 0.000112783975317288
## AIC for loess: 0.849649620825071
## Span for loess: 0.774473923713453
## GCV for loess: 0.000104366910633929
## AIC for loess: 0.817432721762632
## Span for loess: 0.117410797614178
## GCV for loess: 0.000102882442352089
## AIC for loess: 0.823567196978535
## Span for loess: 0.570048557620264
## GCV for loess: 0.000167796520321635
## AIC for loess: 0.735110824287097
```

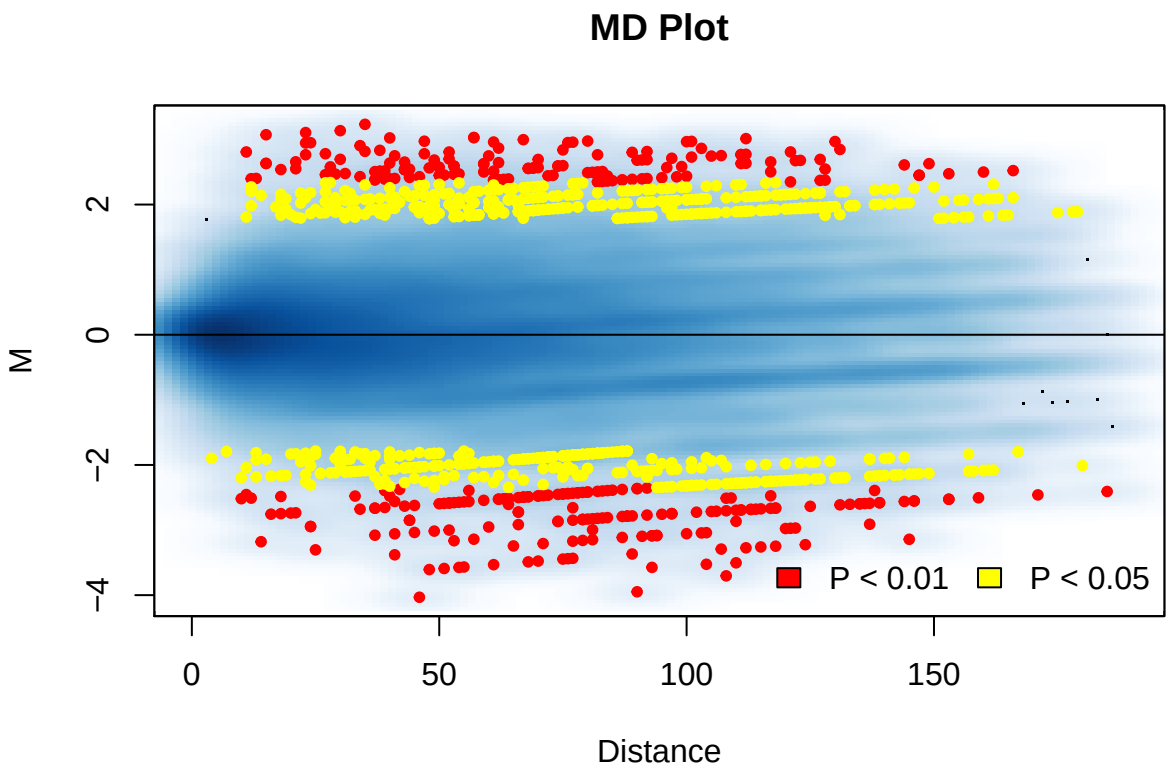
```
## Span for loess: 0.350942491766486
## GCV for loess: 0.000176116533123612
## AIC for loess: 0.595390672426852
## Span for loess: 0.385996667944995
## GCV for loess: 0.000233432323197064
## AIC for loess: 0.703111458235763
## Span for loess: 0.690668169350016
## GCV for loess: 0.000286177851998027
## AIC for loess: 0.772894394556544
## Span for loess: 0.351247228849436
## GCV for loess: 0.000336977521972424
## AIC for loess: 0.943177443308382
## Span for loess: 0.323610900126591
## GCV for loess: 0.00028737049117189
## AIC for loess: 0.760873675703899
## Span for loess: 0.898086317776377
## GCV for loess: 0.000823149659629041
## AIC for loess: 1.01724016836309
## Span for loess: 0.843353262606013
## GCV for loess: 0.000272297855626922
## AIC for loess: 0.295925158014844
## Span for loess: 0.881129049832348
## GCV for loess: 0.000705505157760455
## AIC for loess: 0.137253785145423
## Span for loess: 0.805851060266819
## GCV for loess: 0.00161138159381043
## AIC for loess: 0.889483819181519
## Span for loess: 0.160512388617809
## GCV for loess: 0.00012845482316353
## AIC for loess: 0.957048537338341
```

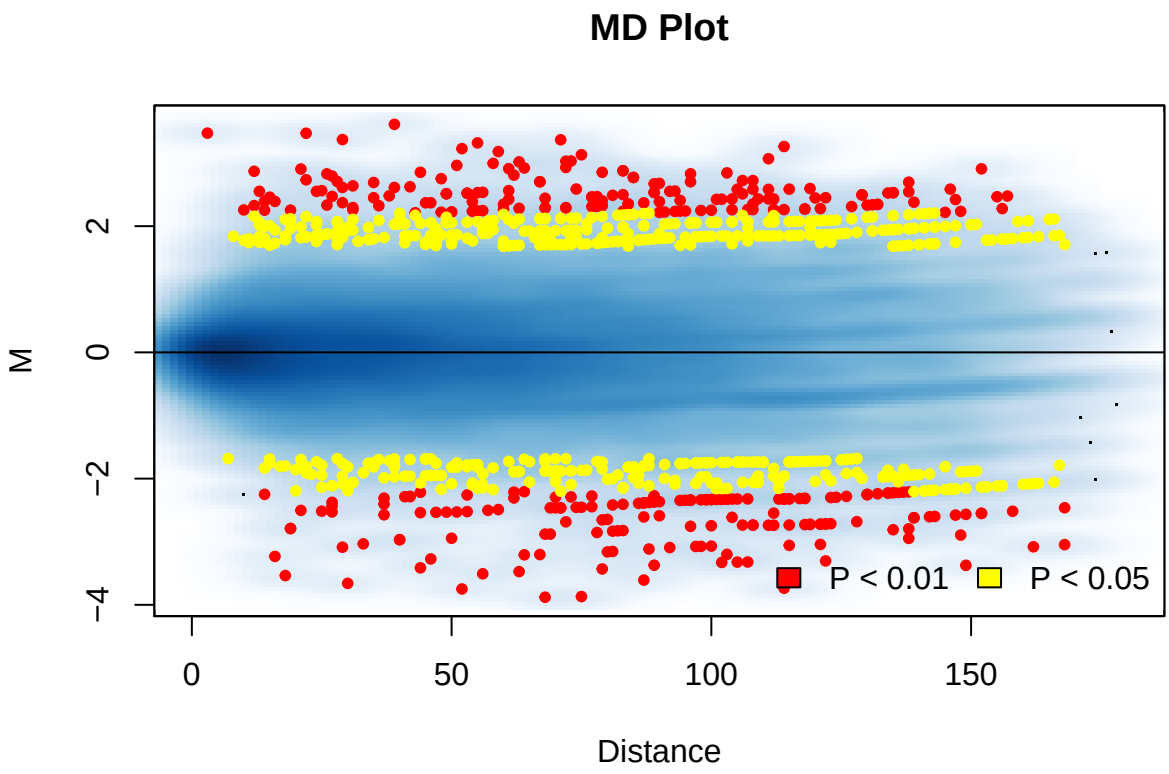
```
# dev.off()

# pdf(file = paste0(man_path, 'new_data_diff.pdf'))
# lib1_tables <- hic_diff(lib1_tables, Plot = TRUE, diff.thresh = NA, iterations = 1000)
# dev.off()

lib1_tables <- hic_compare(lib1_tables, Plot = FALSE, adjust_dist = TRUE, A.quantile = 0.05, p.method =
```

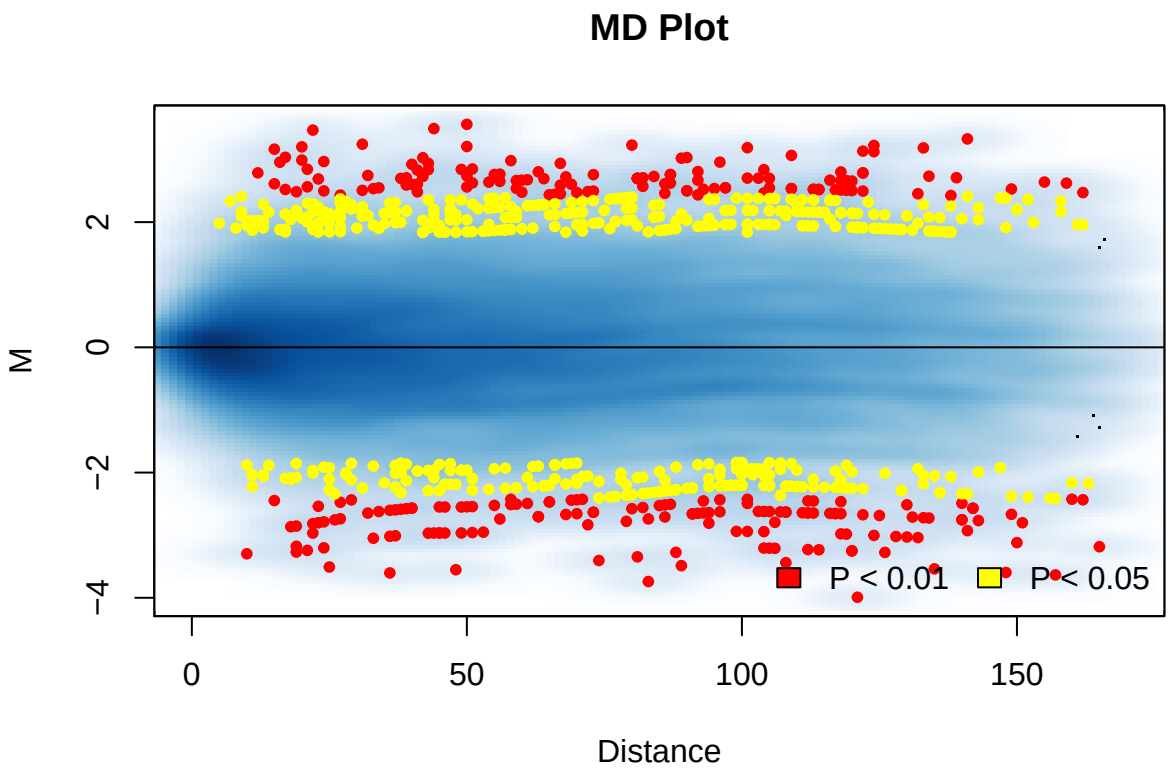

MD Plot
M
4
2
0
−2
−4
0
50
100
150
200
250
Distance
P < 0.01
P < 0.05

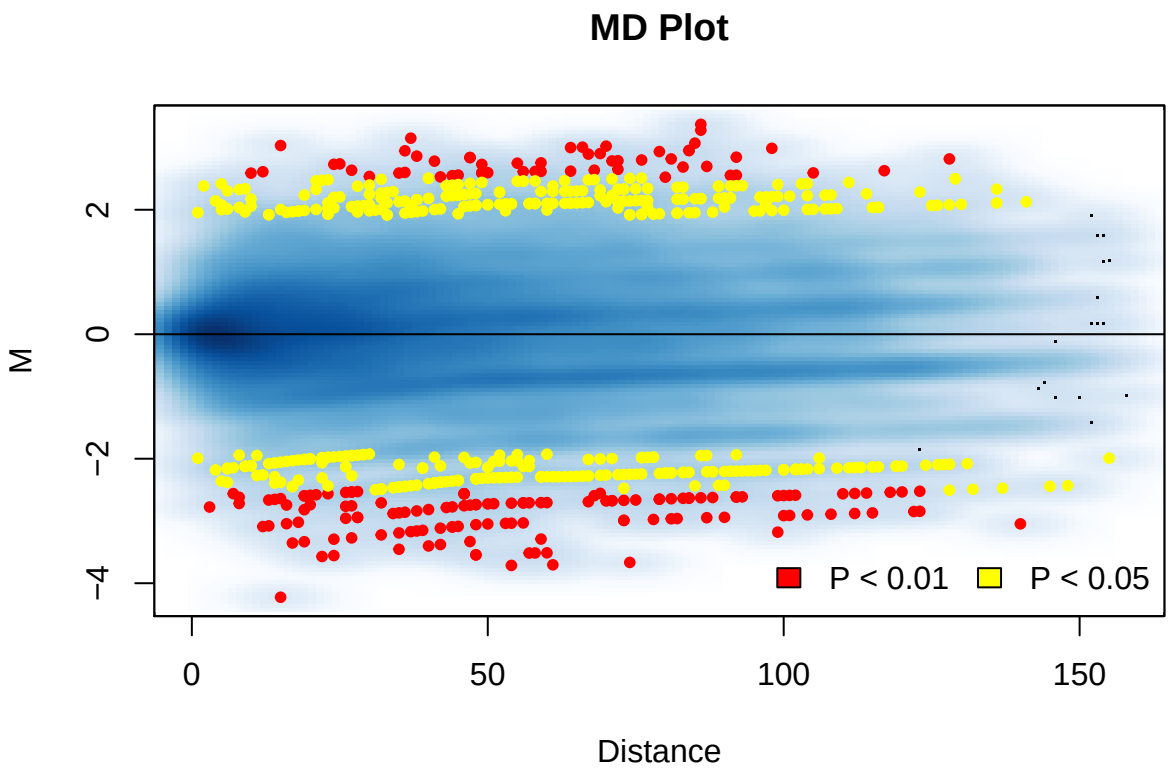

MD Plot
M
4
2
0
−2
−4
0
50
100
150
200
250
Distance
P < 0.01
P < 0.05

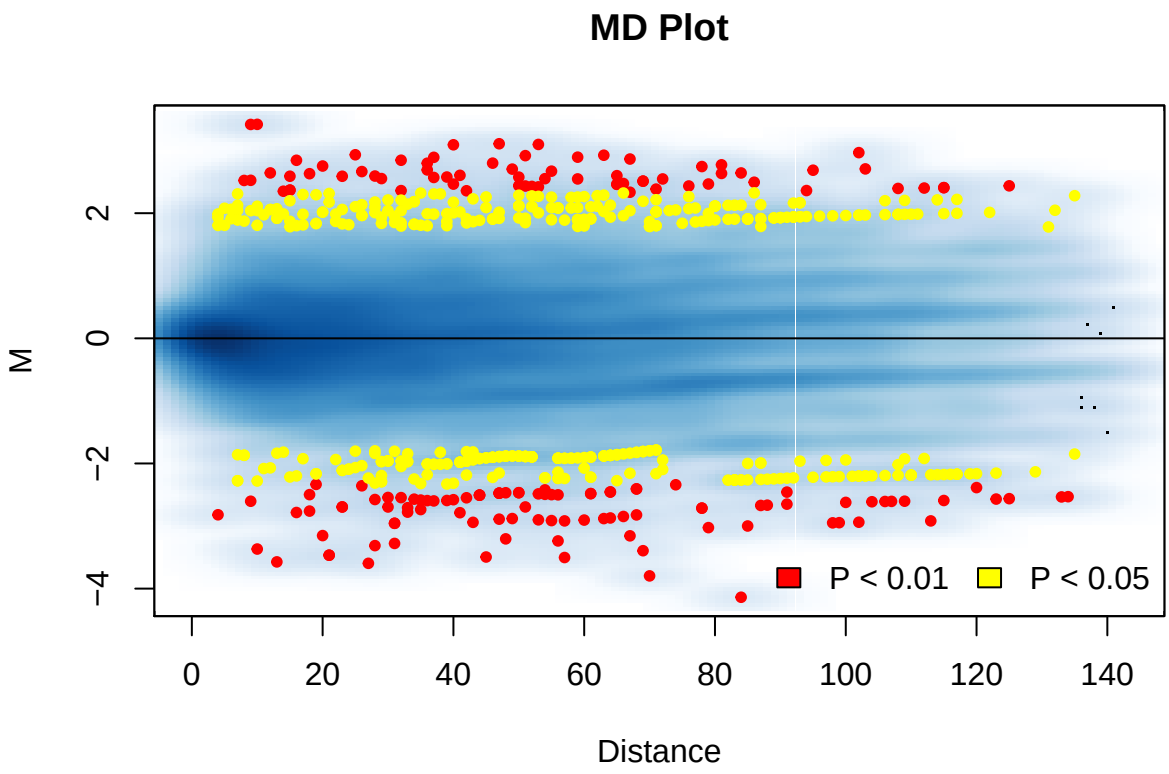

MD Plot
M
2
0
−2
−4
0
50
100
150
200
Distance
P < 0.01
P < 0.05

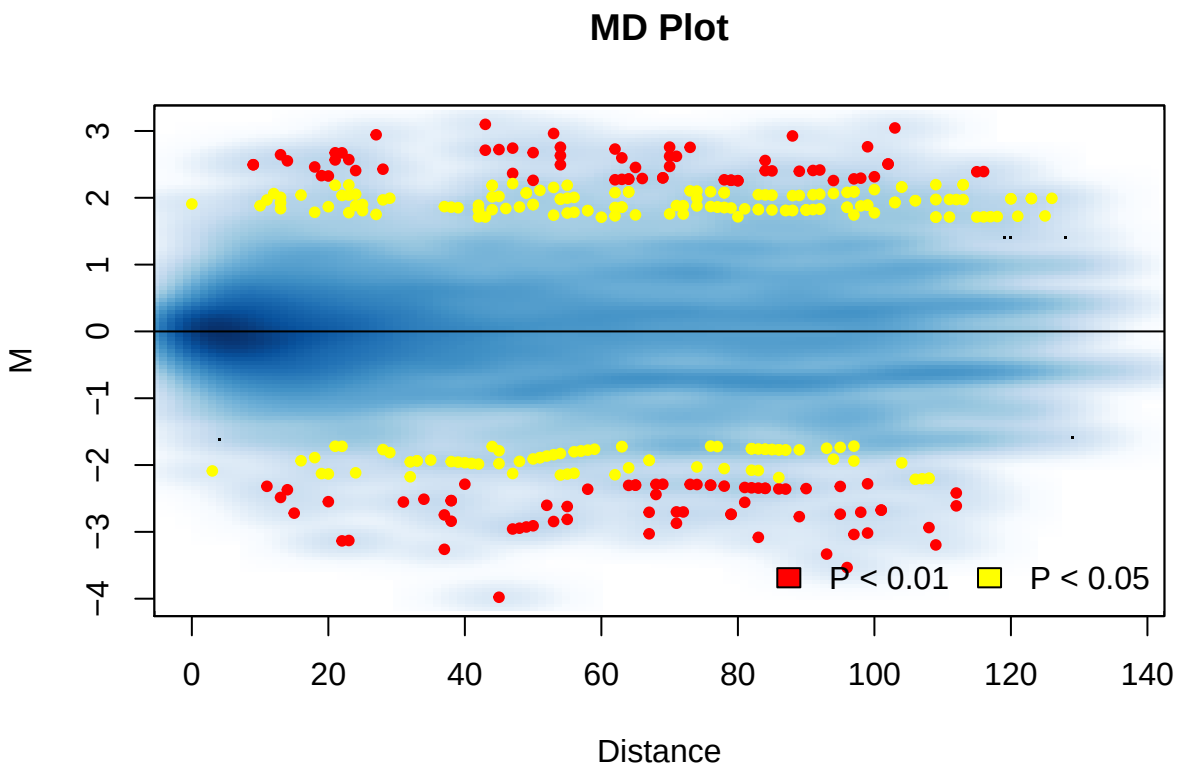

MD Plot
M
2
0
−2
−4
0
50
100
150
Distance
P < 0.01
P < 0.05

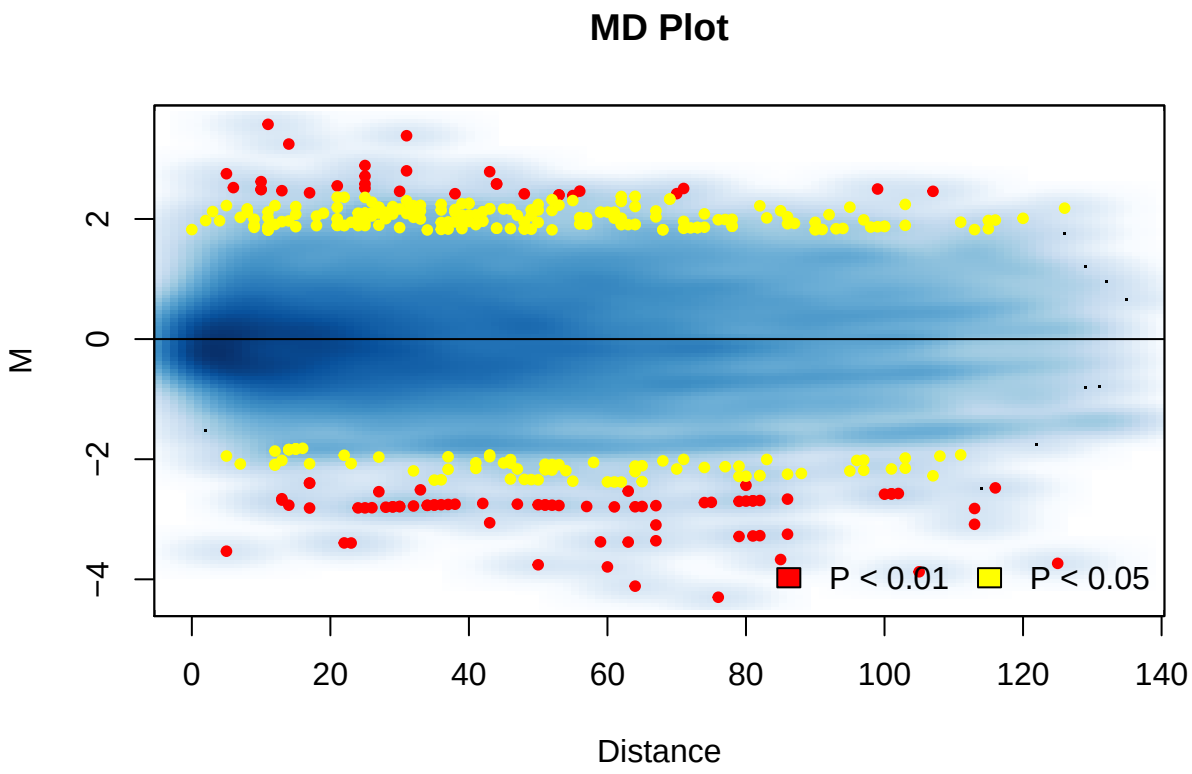

MD Plot
M
2
0
−2
−4
0
50
100
150
Distance
P < 0.01
P < 0.05

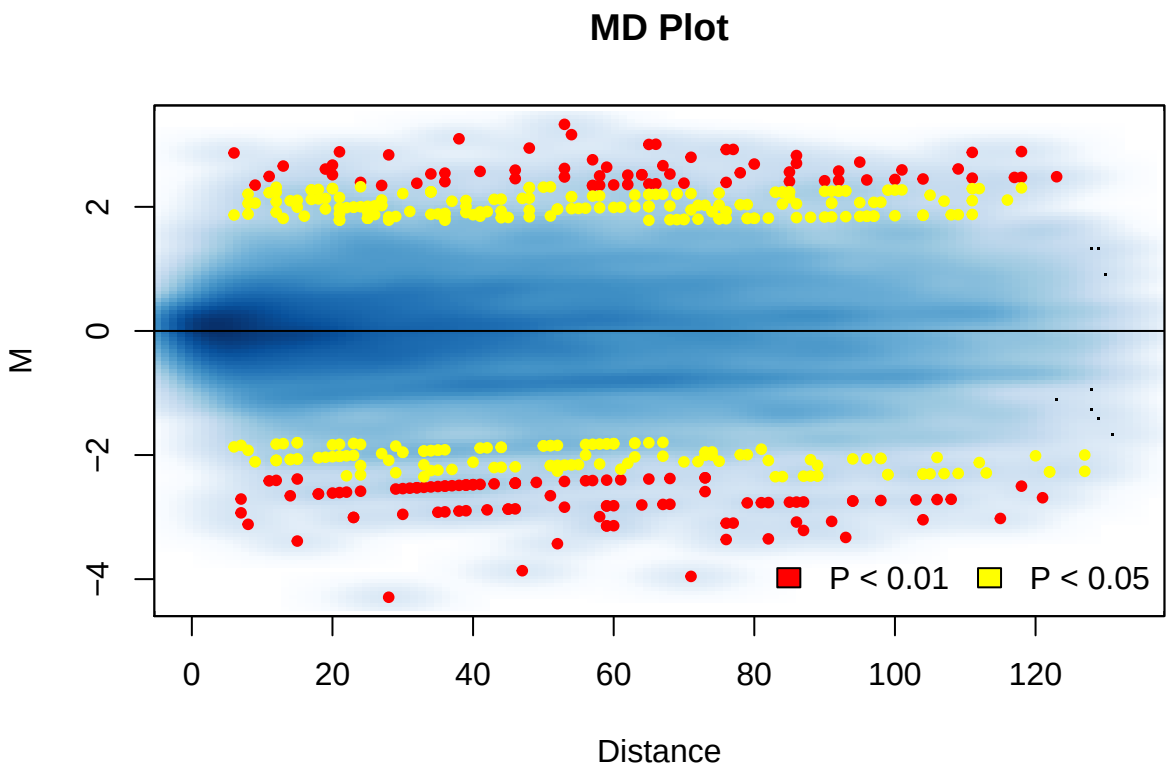

MD Plot
M
2
0
−2
−4
0
50
100
150
Distance
P < 0.01
P< 0.05

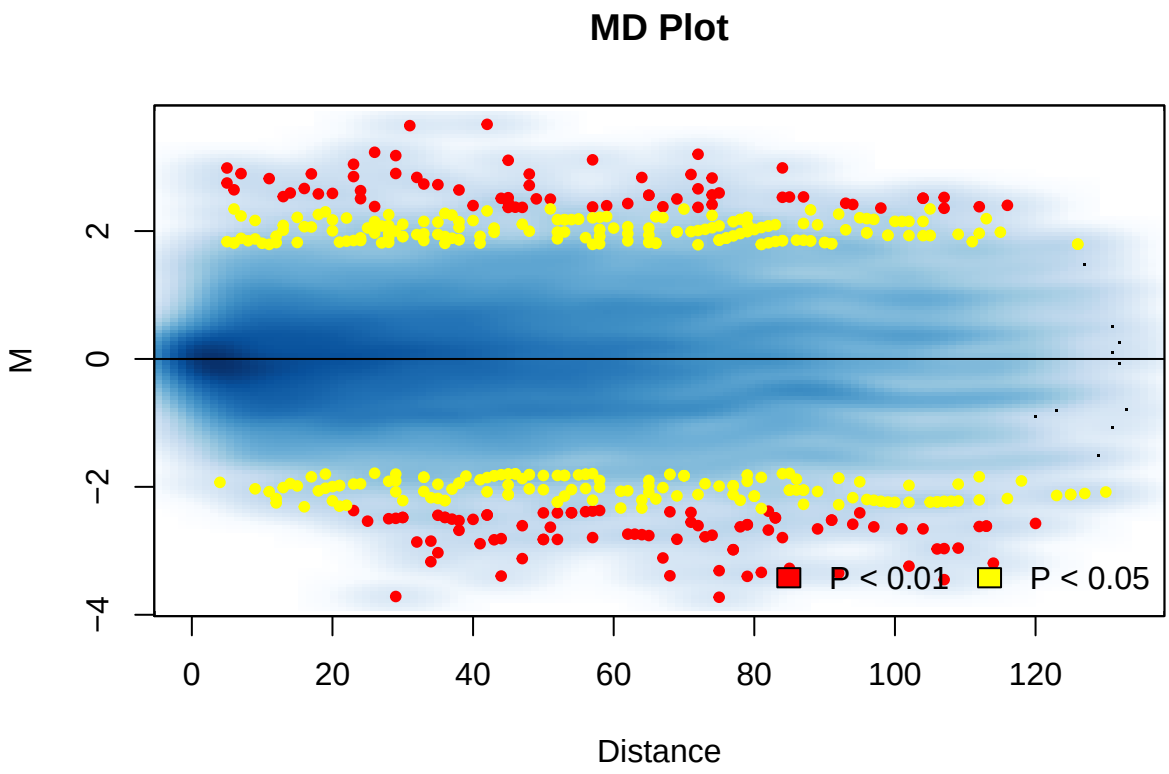

MD Plot
M
2
0
−2
−4
0
50
100
150
Distance
P < 0.01
P < 0.05

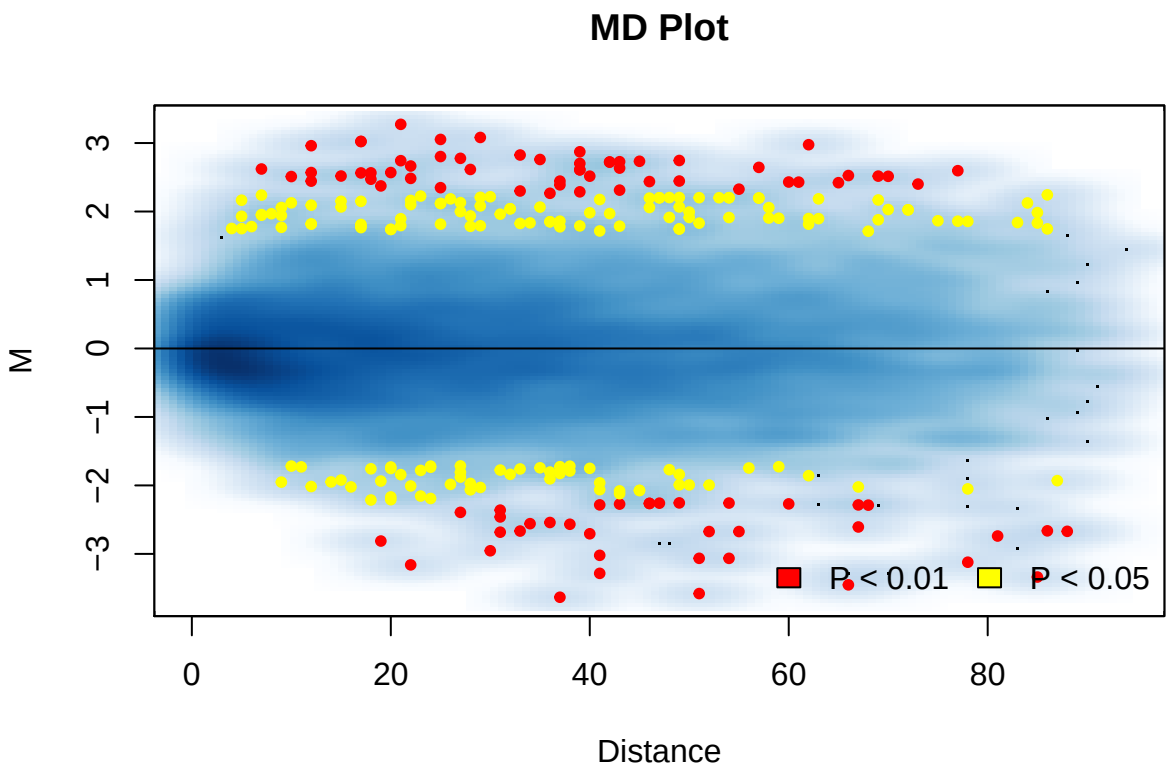

MD Plot
M
2
0
−2
−4
0
20
40
60
80
100
120
140
Distance
P < 0.01
P < 0.05

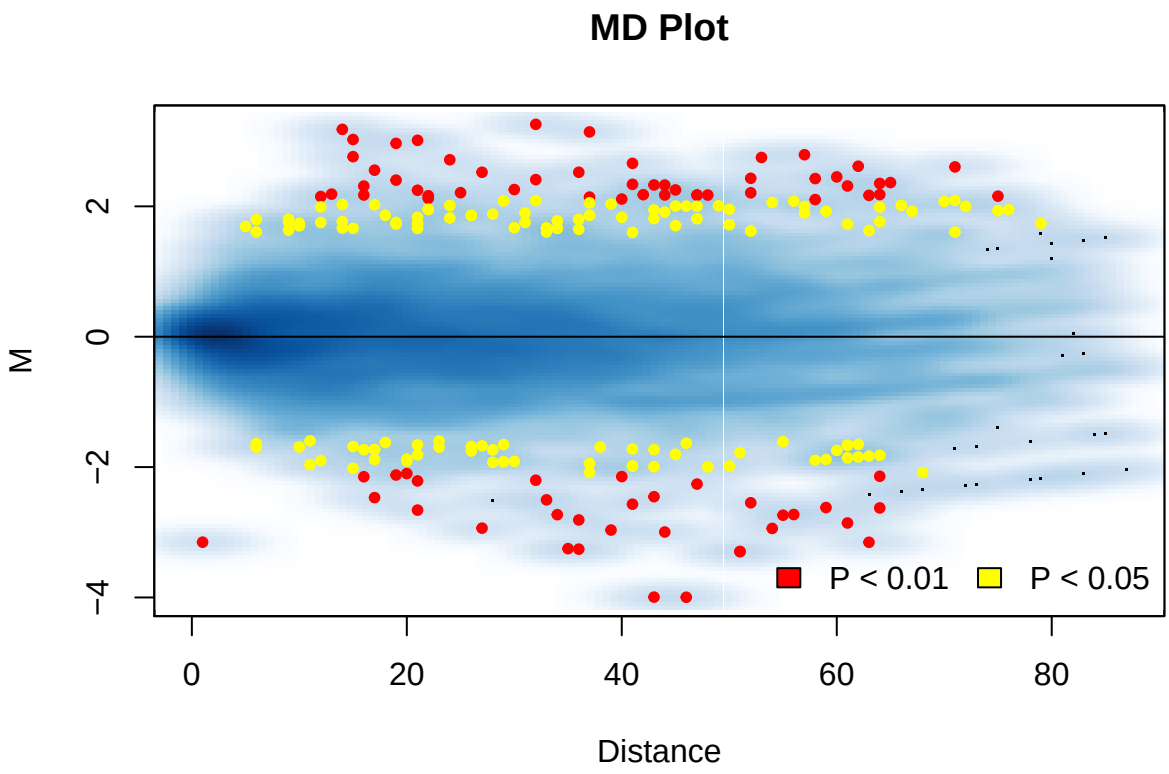

MD Plot
M
3
2
1
0
−1
−2
−3
−4
0
20
40
60
80
100
120
140
Distance
P < 0.01
P < 0.05

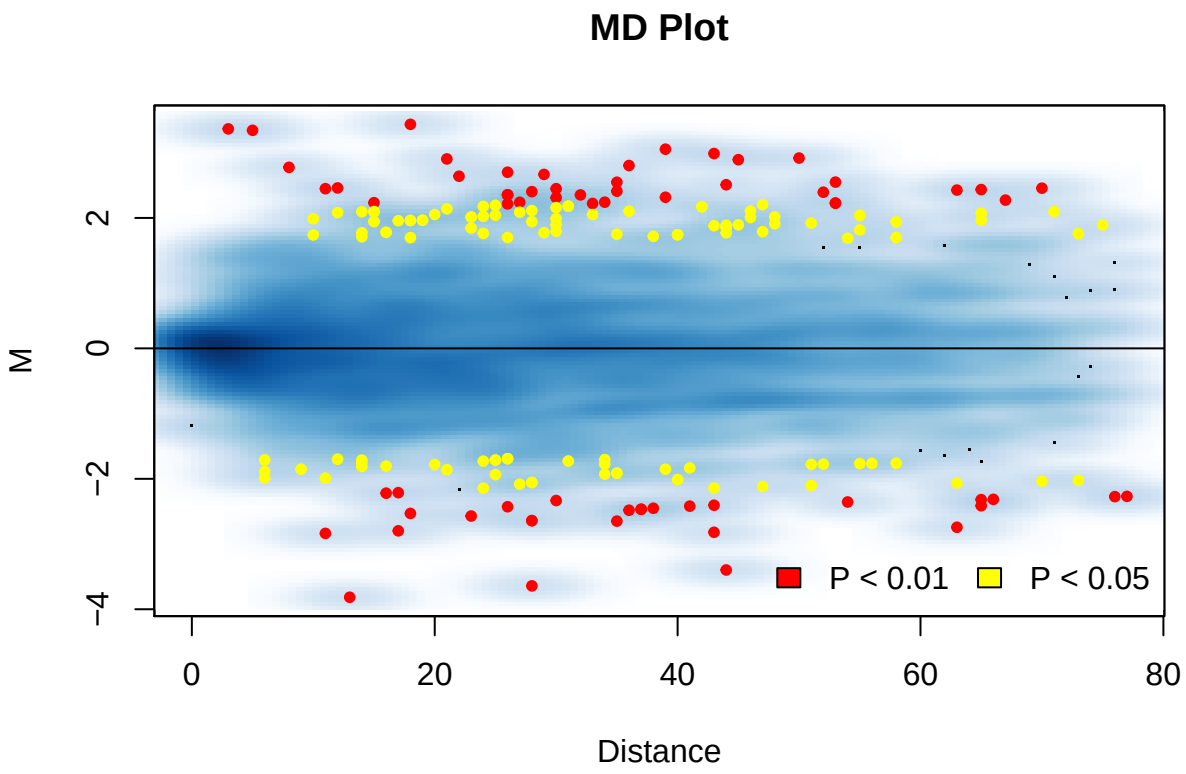

MD Plot
M
2
0
−2
−4
0
20
40
60
80
100
120
140
Distance
P < 0.01
P < 0.05

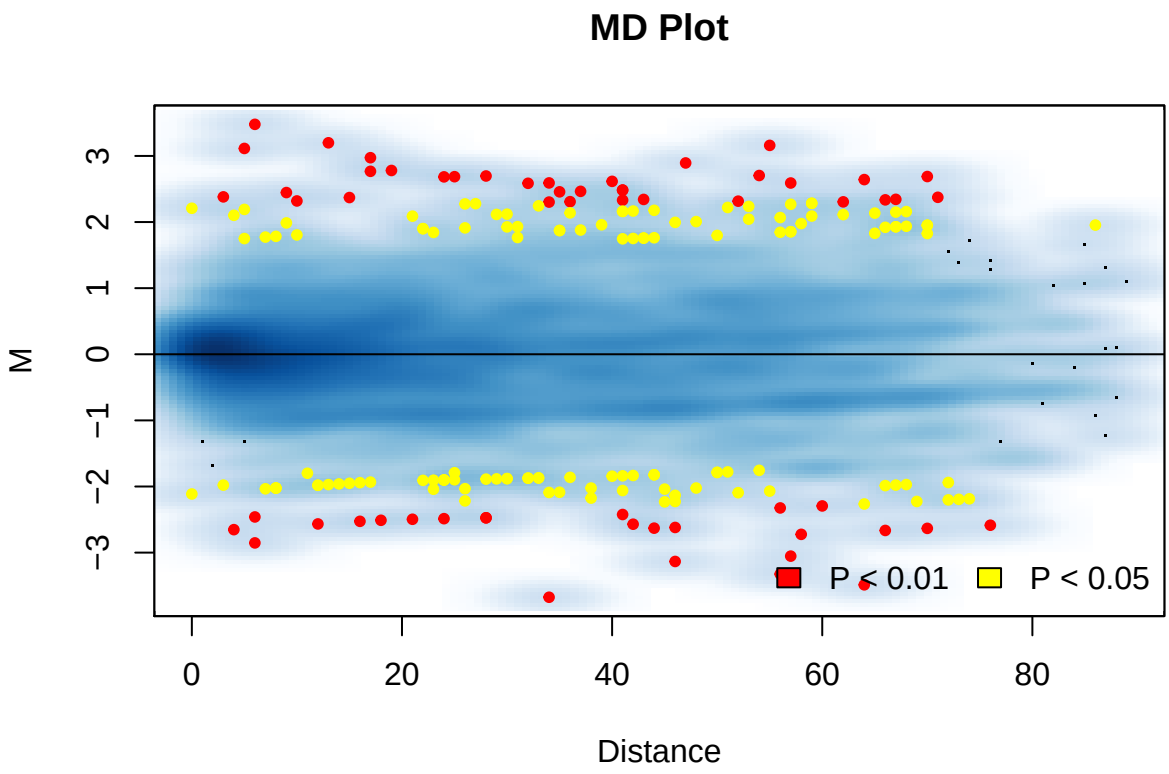

MD Plot
M
2
0
−2
−4
0
20
40
60
80
100
120
Distance
P < 0.01
P < 0.05

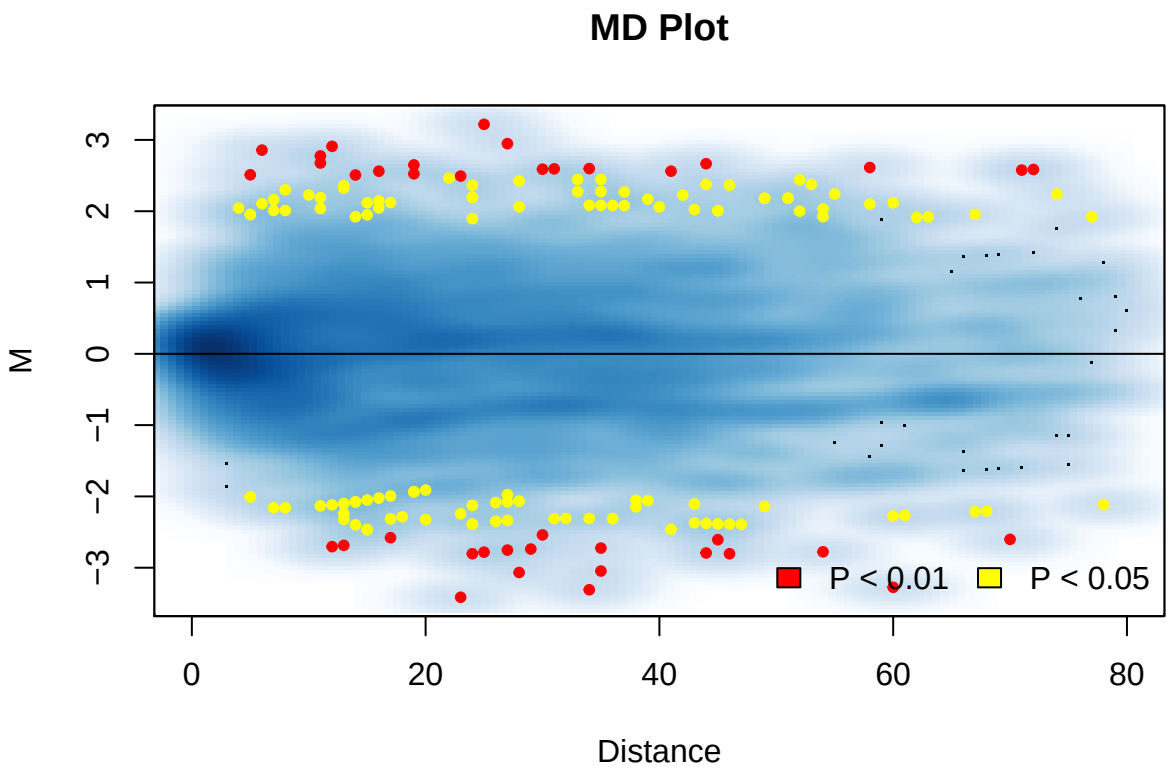

MD Plot
M
2
0
−2
−4
0
20
40
60
80
100
120
Distance
P < 0.01
P < 0.05

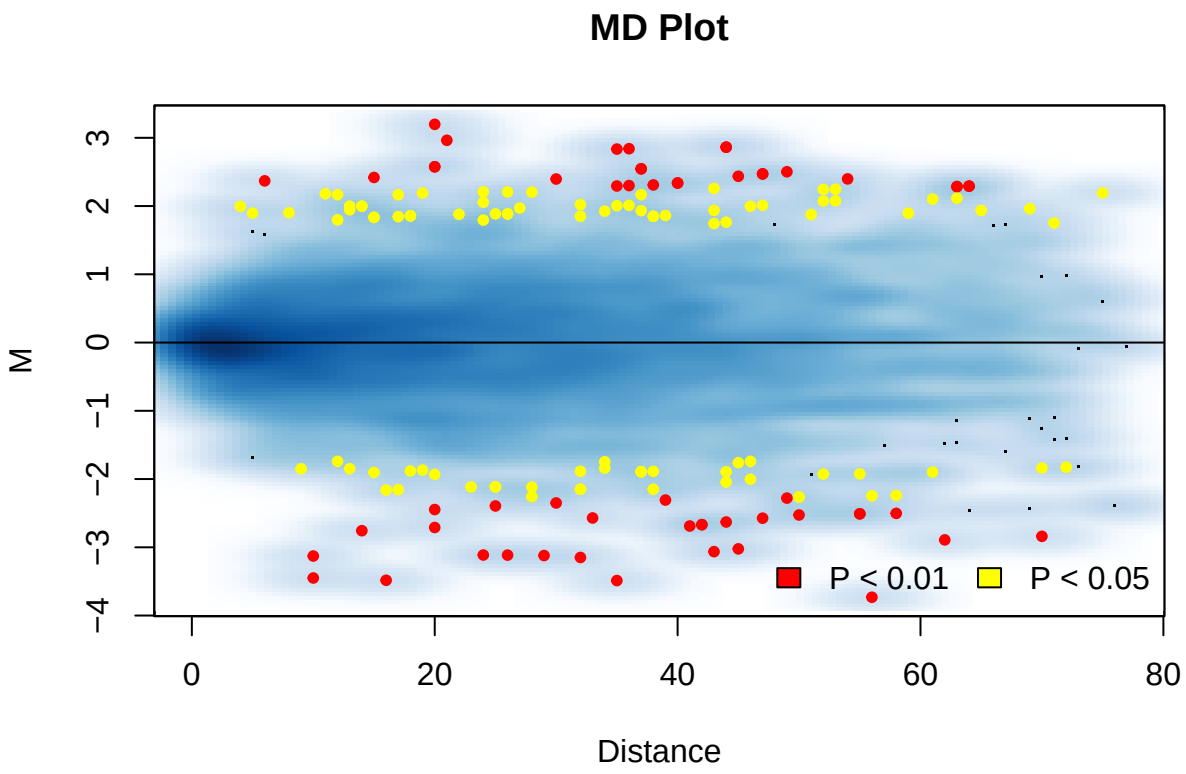

MD Plot
M
3
2
1
0
−1
−2
−3
0
20
40
60
80
Distance
P < 0.01
P < 0.05

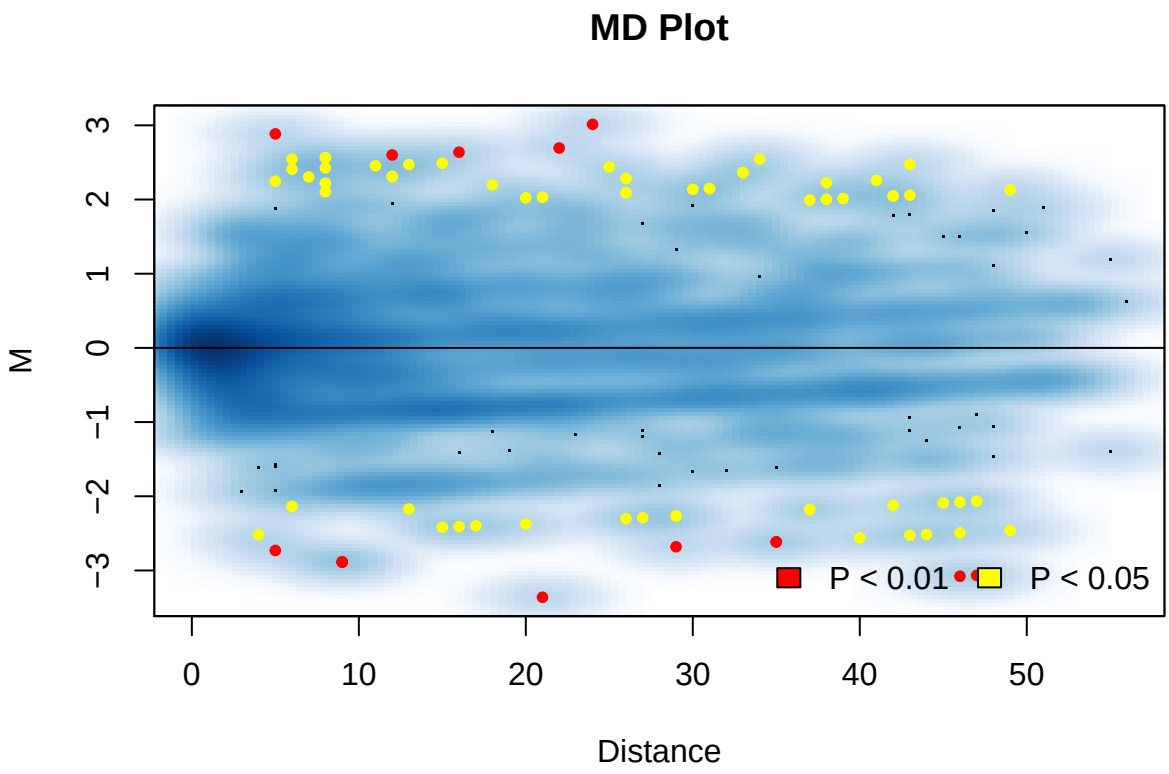

MD Plot
M
2
0
−2
−4
0
20
40
60
80
Distance
P < 0.01
P < 0.05

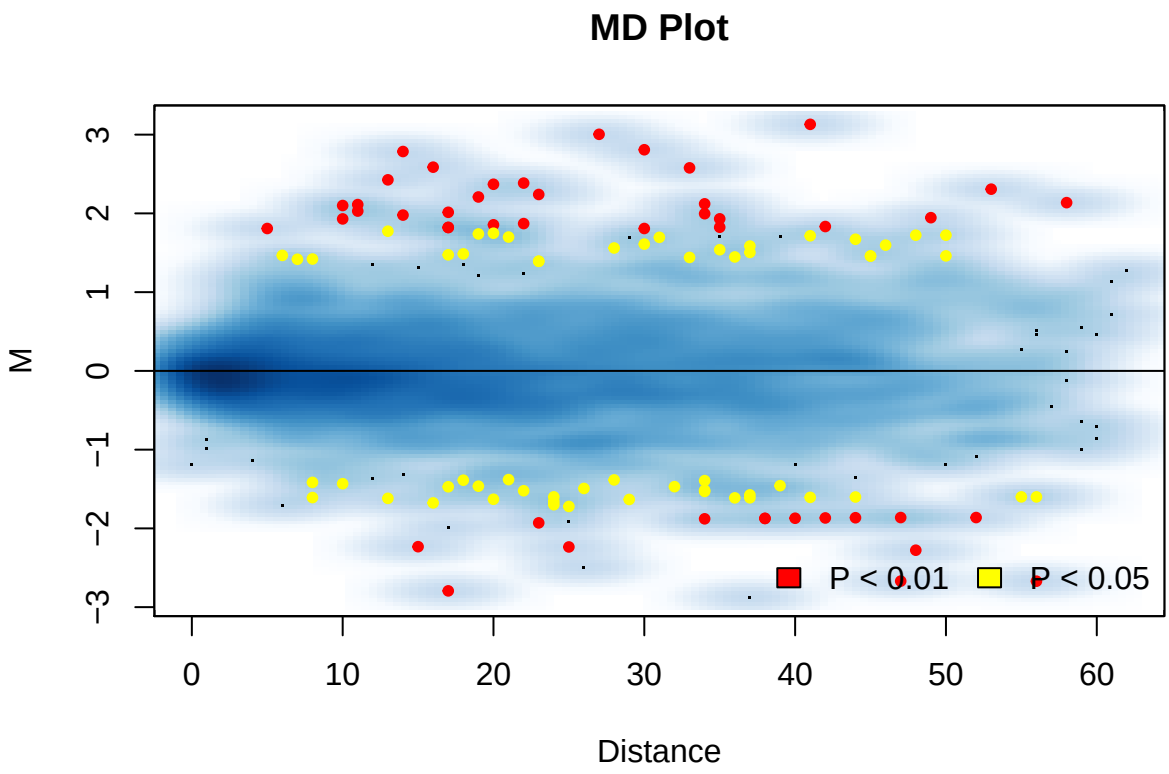

MD Plot
M
2
0
−2
−4
0
20
40
60
80
Distance
P < 0.01
P < 0.05

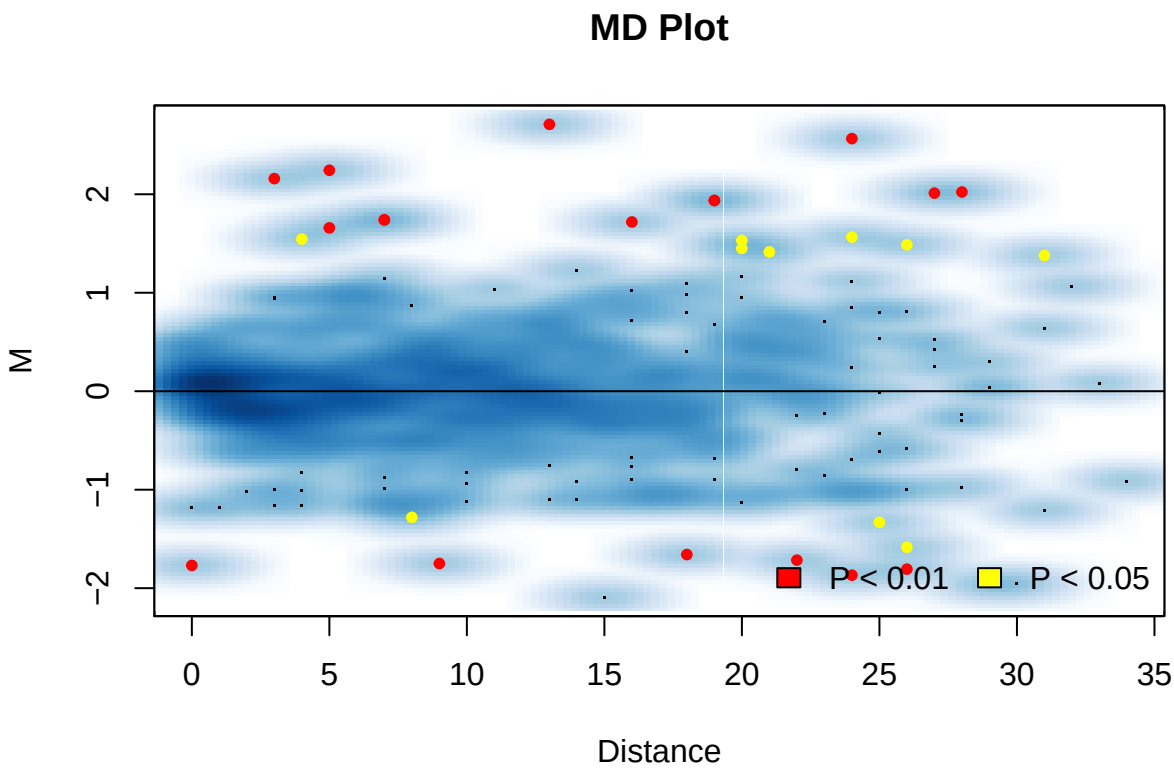

MD Plot
M
3
2
1
0
−1
−2
−3
0
20
40
60
80
Distance
P < 0.01
P < 0.05

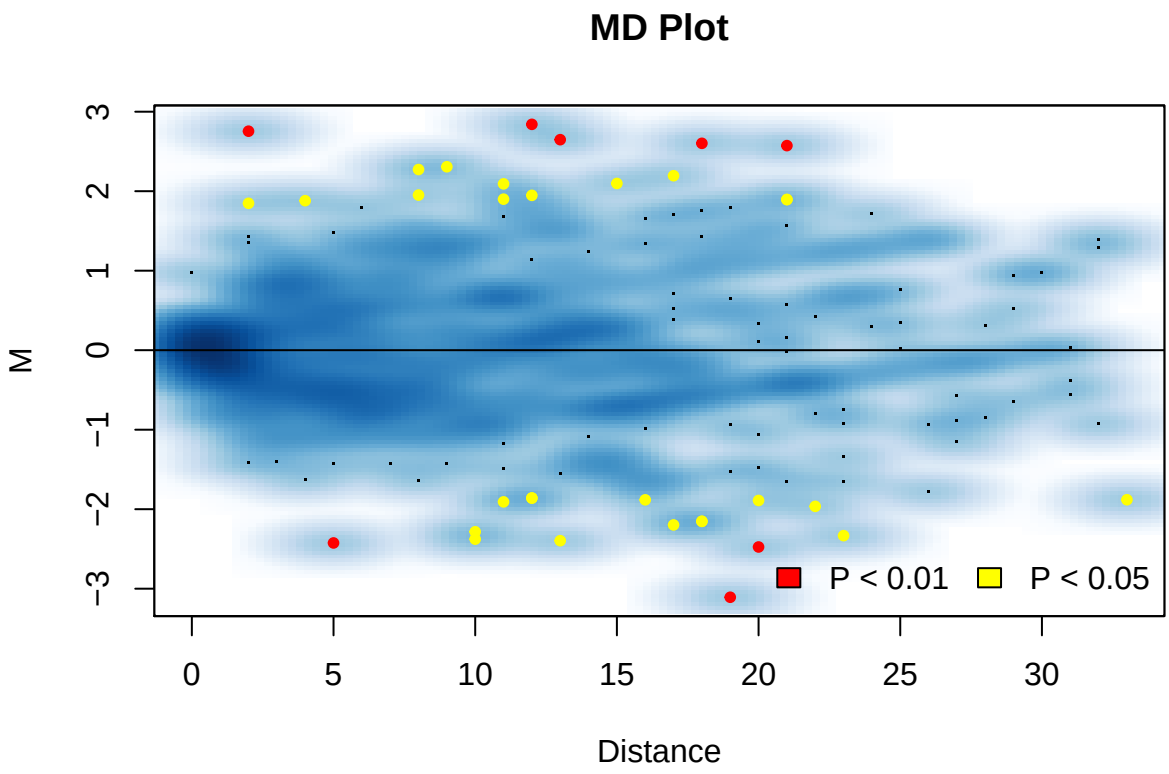

MD Plot
M
3
2
1
0
−1
−2
−3
0
20
40
60
80
Distance
P < 0.01
P < 0.05

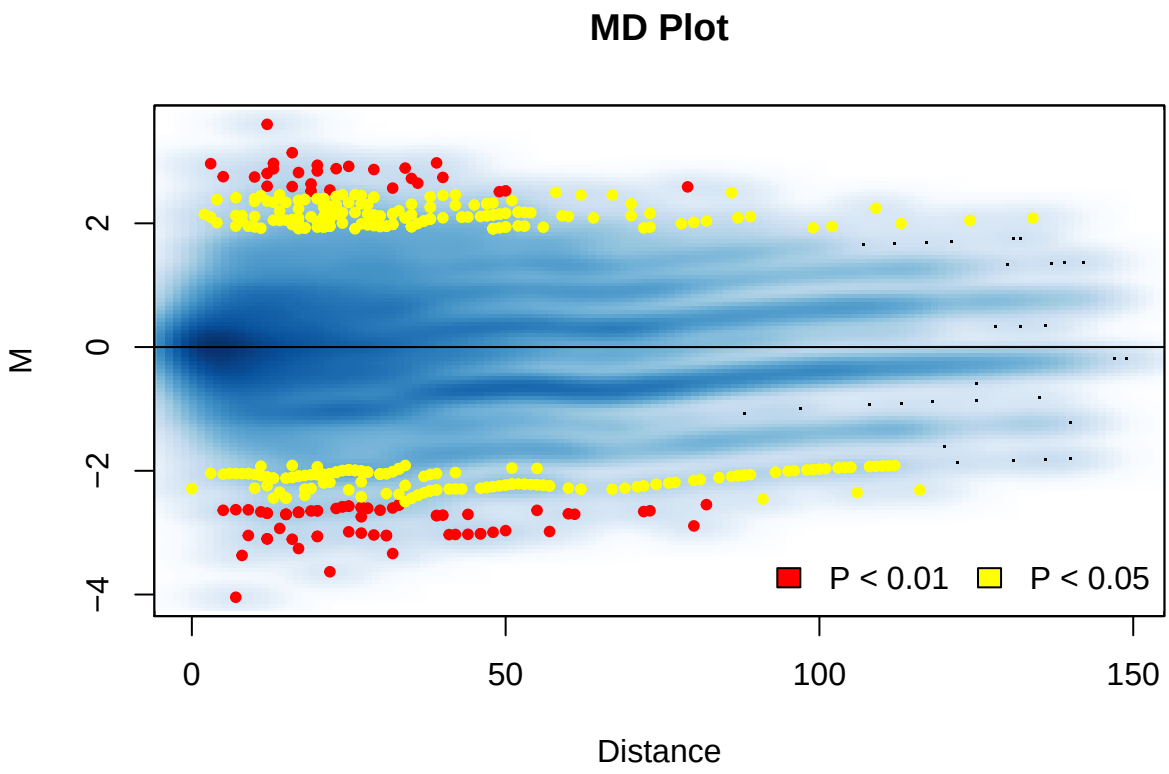

MD Plot
M
3
2
1
0
−1
−2
−3
−4
0
20
40
60
80
Distance
P < 0.01
P < 0.05


MD Plot
M
3
2
1
0
−1
−2
−3
0
10
20
30
40
50
Distance
P < 0.01
P < 0.05


MD Plot
M
3
2
1
0
−1
−2
−3
0
10
20
30
40
50
60
Distance
P < 0.01
P < 0.05


MD Plot
M
2
1
0
−1
−2
0
5
10
15
20
25
30
35
Distance
P < 0.01
P < 0.05


MD Plot
M
3
2
1
0
−1
−2
−3
0
5
10
15
20
25
30
Distance
P < 0.01
P < 0.05



Read in smyth data

```
# set up their data for overlap analysis
# core_path <- "C:/VM_Shared/DNA_int_analysis/prostate_data/"
core_path <- "D:/3D_DNA/prostate_data/"
library(readr)
their.results <- read_tsv(paste0(core_path, 'results.tsv'))
```

```
## Parsed with column specification:
## cols(
##   anchor = col_character(),
##   anchor.start = col_integer(),
##   anchor.end = col_integer(),
##   target = col_character(),
##   target.start = col_integer(),
##   target.end = col_integer(),
##   logFC = col_double(),
##   logCPM = col_double(),
##   F = col_double(),
##   PValue = col_double(),
##   FDR = col_double(),
##   anchor.anno = col_character(),
##   target.anno = col_character()
## )
```

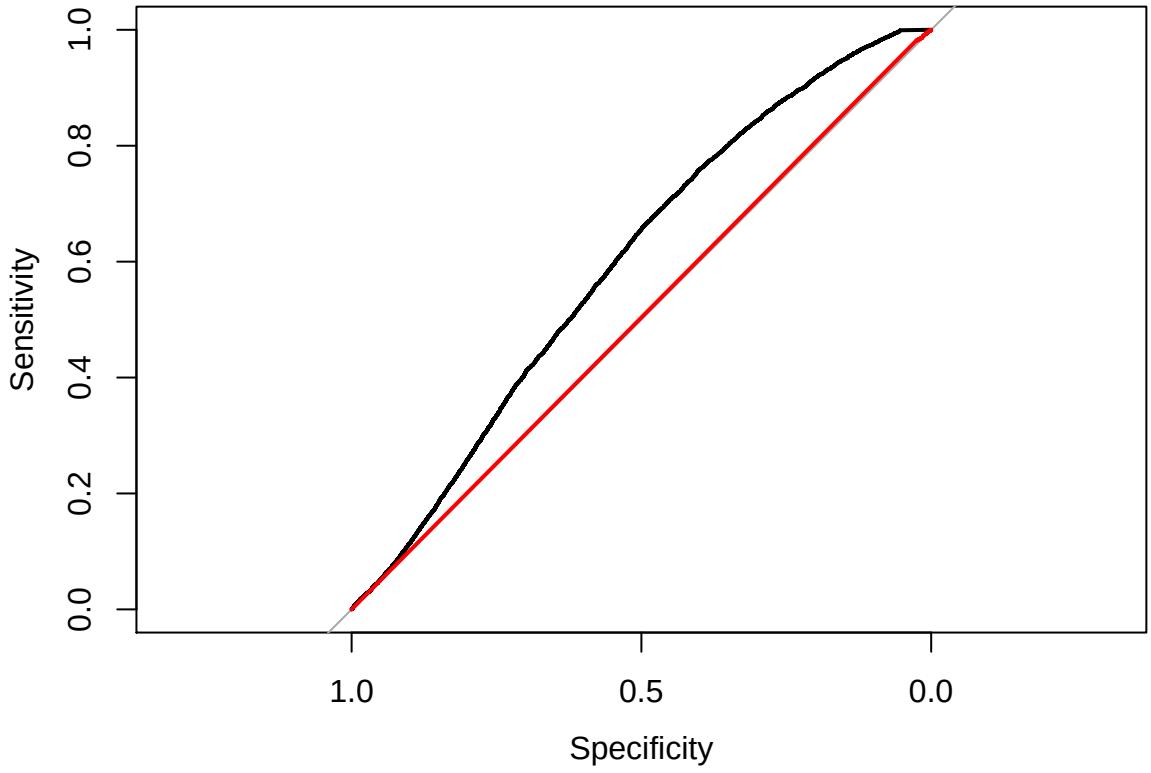
```
# their.results <- subset(their.results, FDR < 0.05)
# they used the lower triangle of the matrix for most pairs of regions while we only use upper triangle
new_start1 <- ifelse(their.results$anchor.start >= their.results$target.start, their.results$target.sta
new_end1 <- ifelse(their.results$anchor.end >= their.results$target.end, their.results$target.end, thei
new_start2 <- ifelse(their.results$anchor.start >= their.results$target.start, their.results$anchor.sta
new_end2 <- ifelse(their.results$anchor.end >= their.results$target.end, their.results$anchor.end, thei
new.their.results <- data.frame(anchor = their.results$anchor, anchor.start = new_start1, anchor.end = n
                                target = their.results$target, target.start = new_start2, target.end = n
new.their.results <- cbind(new.their.results, their.results[, 7:11])
their.results <- new.their.results
s1 <- GRanges(their.results$anchor, IRanges(start = their.results$anchor.start, end = their.results$anch
s2 <- GRanges(their.results$target, IRanges(start = their.results$target.start, end = their.results$targ
smyth.gi <- GInteractions(s1, s2)
meta <- cbind(their.results$logFC, their.results$logCPM, their.results$F, their.results$PValue, their.re
meta <- as.data.frame(meta)
colnames(meta) <- c('logFC', 'logCPM', 'F', 'PValue', 'FDR')
S4Vectors::values(smyth.gi) <- cbind(S4Vectors::values(smyth.gi), meta)
```

overlap significant smyth regions with our regions

```
# combine out results into single object & convert to interaction set
ours <- rbindlist(lib1_tables)
ours <- dataframe2interactionset(ours)

# get significant smyth interactions
smyth.sig <- smyth.gi[smyth.gi$FDR < 0.05,]
smyth.sig <- sort(smyth.sig)

# get overlaps
olaps <- InteractionSet::findOverlaps(smyth.sig, ours, use.region = 'same', minoverlap = 500000)
# make truth vector
truth <- rep(0, length(ours))
truth[unique(olaps@to)] <- 1
```

ROC

```
pval_roc <- roc(response = truth, predictor = ours@elementMetadata$p.value)
adjpval_roc <- roc(response = truth, predictor = ours@elementMetadata$p.adj)


plot(pval_roc)
plot(adjpval_roc, add = TRUE, col = 'red')
```



look at how smyth corresponds to our results

```
sum(truth == 1)
```

```
## [1] 5494
```

```
sum(truth == 1 & ours@elementMetadata$p.adj < 0.05)
```

```
## [1] 11
```

```
sum(truth == 1 & ours@elementMetadata$p.value < 0.05)
```

```
## [1] 347
```

```
sum(ours@elementMetadata$p.value < 0.05)
```

```
## [1] 11065
```

```
sum(ours@elementMetadata$p.adj < 0.05)
```

```
## [1] 420
```

smyth significant regions and our significant regions barely overlap.