Methylation HW10

Jessica Murphy

April 17, 2019

```
suppressWarnings(suppressMessages(library("minfi", quietly=T)))
suppressWarnings(suppressMessages(library("wateRmelon", quietly=T)))
suppressWarnings(suppressMessages(library("ChAMP", quietly=T)))
suppressWarnings(suppressMessages(library("RColorBrewer", quietly=T)))

sessionInfo()

## R version 3.5.1 (2018-07-02)
## Platform: x86_64-redhat-linux-gnu (64-bit)
## Running under: CentOS release 6.10 (Final)
##
## Matrix products: default
## BLAS: /usr/lib64/R/lib/libRblas.so
## LAPACK: /usr/lib64/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
##  [1] splines   stats4    parallel  stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] RColorBrewer_1.1-2
##  [2] ChAMP_2.12.4
##  [3] IlluminaHumanMethylationEPICmanifest_0.3.0
##  [4] Illumina450ProbeVariants.db_1.18.0
##  [5] DMRcate_1.18.0
##  [6] DMRcatedata_1.18.0
```

```
##  [7] DSS_2.30.1
##  [8] bsseq_1.18.0
##  [9] FEM_3.10.0
## [10] graph_1.60.0
## [11] impute_1.56.0
## [12] igraph_1.2.4
## [13] corrplot_0.84
## [14] marray_1.60.0
## [15] Matrix_1.2-15
## [16] ChAMPdata_2.14.1
## [17] wateRmelon_1.26.0
## [18] illuminaio_0.24.0
## [19] IlluminaHumanMethylation450kanno.ilmn12.hg19_0.6.0
## [20] ROC_1.58.0
## [21] lumi_2.34.0
## [22] methylumi_2.28.0
## [23] FDb.InfiniumMethylation.hg19_2.2.0
## [24] org.Hs.eg.db_3.7.0
## [25] TxDb.Hsapiens.UCSC.hg19.knownGene_3.2.2
## [26] GenomicFeatures_1.34.3
## [27] AnnotationDbi_1.44.0
## [28] ggplot2_3.1.0
## [29] reshape2_1.4.3
## [30] scales_1.0.0
## [31] limma_3.38.3
## [32] minfi_1.28.3
## [33] bumphunter_1.24.5
## [34] locfit_1.5-9.1
## [35] iterators_1.0.10
## [36] foreach_1.4.4
## [37] Biostrings_2.50.2
## [38] XVector_0.22.0
## [39] SummarizedExperiment_1.12.0
## [40] DelayedArray_0.8.0
## [41] BiocParallel_1.16.6
## [42] matrixStats_0.54.0
## [43] Biobase_2.42.0
## [44] GenomicRanges_1.34.0
## [45] GenomeInfoDb_1.18.2
## [46] IRanges_2.16.0
## [47] S4Vectors_0.20.1
## [48] BiocGenerics_0.28.0
## [49] knitr_1.21
##
## loaded via a namespace (and not attached):
```

```
##   [1] rtracklayer_1.42.1
##   [2] prabclus_2.2-7
##   [3] R.methodsS3_1.7.1
##   [4] pkgmaker_0.27
##   [5] tidyr_0.8.2
##   [6] acepack_1.4.1
##   [7] bit64_0.9-7
##   [8] R.utils_2.8.0
##   [9] data.table_1.12.0
##  [10] rpart_4.1-13
##  [11] RCurl_1.95-4.11
##  [12] GEOquery_2.50.5
##  [13] AnnotationFilter_1.6.0
##  [14] doParallel_1.0.14
##  [15] preprocessCore_1.44.0
##  [16] RSQLite_2.1.1
##  [17] combinat_0.0-8
##  [18] bit_1.1-14
##  [19] xml2_1.2.0
##  [20] httpuv_1.4.5.1
##  [21] assertthat_0.2.0
##  [22] IlluminaHumanMethylation450kmanifest_0.4.0
##  [23] IlluminaHumanMethylationEPICanno.ilm10b4.hg19_0.6.0
##  [24] viridis_0.5.1
##  [25] isva_1.9
##  [26] xfun_0.5
##  [27] hms_0.4.2
##  [28] evaluate_0.13
##  [29] missMethyl_1.16.0
##  [30] DNAcopy_1.56.0
##  [31] promises_1.0.1
##  [32] DEoptimR_1.0-8
##  [33] progress_1.2.0
##  [34] dendextend_1.9.0
##  [35] DBI_1.0.0
##  [36] htmlwidgets_1.3
##  [37] reshape_0.8.8
##  [38] purrr_0.3.0
##  [39] dplyr_0.8.0.1
##  [40] backports_1.1.3
##  [41] permute_0.9-4
##  [42] trimcluster_0.1-2.1
##  [43] annotate_1.60.0
##  [44] biomaRt_2.38.0
##  [45] ensembldb_2.6.6
```

```
##  [46] withr_2.1.2
##  [47] globaltest_5.36.0
##  [48] Gviz_1.26.5
##  [49] BSgenome_1.50.0
##  [50] robustbase_0.93-3
##  [51] checkmate_1.9.1
##  [52] GenomicAlignments_1.18.1
##  [53] prettyunits_1.0.2
##  [54] mclust_5.4.2
##  [55] cluster_2.0.7-1
##  [56] RPMM_1.25
##  [57] lazyeval_0.2.1
##  [58] crayon_1.3.4
##  [59] genefilter_1.64.0
##  [60] pkgconfig_2.0.2
##  [61] nlme_3.1-137
##  [62] ProtGenerics_1.14.0
##  [63] nnet_7.3-12
##  [64] rlang_0.3.1
##  [65] diptest_0.75-7
##  [66] nleqslv_3.3.2
##  [67] registry_0.5
##  [68] affyio_1.52.0
##  [69] dichromat_2.0-0
##  [70] rngtools_1.3.1
##  [71] base64_2.0
##  [72] Rhdf5lib_1.4.2
##  [73] base64enc_0.1-3
##  [74] geneLenDataBase_1.18.0
##  [75] whisker_0.3-2
##  [76] viridisLite_0.3.0
##  [77] bitops_1.0-6
##  [78] R.oo_1.22.0
##  [79] KernSmooth_2.23-15
##  [80] blob_1.1.1
##  [81] DelayedMatrixStats_1.4.0
##  [82] doRNG_1.7.1
##  [83] stringr_1.4.0
##  [84] qvalue_2.14.1
##  [85] nor1mix_1.2-3
##  [86] readr_1.3.1
##  [87] memoise_1.1.0
##  [88] magrittr_1.5
##  [89] plyr_1.8.4
##  [90] bibtex_0.4.2
```

```
##  [91] zlibbioc_1.28.0
##  [92] compiler_3.5.1
##  [93] clue_0.3-56
##  [94] Rsamtools_1.34.1
##  [95] affy_1.60.0
##  [96] JADE_2.0-1
##  [97] IlluminaHumanMethylationEPICanno.ilm10b2.hg19_0.6.0
##  [98] htmlTable_1.13.1
##  [99] Formula_1.2-3
## [100] MASS_7.3-51.1
## [101] mgcv_1.8-27
## [102] tidyselect_0.2.5
## [103] stringi_1.3.1
## [104] askpass_1.1
## [105] latticeExtra_0.6-28
## [106] grid_3.5.1
## [107] VariantAnnotation_1.28.11
## [108] tools_3.5.1
## [109] ruv_0.9.7
## [110] rstudioapi_0.9.0
## [111] foreign_0.8-71
## [112] gridExtra_2.3
## [113] digest_0.6.18
## [114] BiocManager_1.30.4
## [115] shiny_1.2.0
## [116] quadprog_1.5-5
## [117] fpc_2.1-11.1
## [118] Rcpp_1.0.0
## [119] siggenes_1.56.0
## [120] later_0.8.0
## [121] httr_1.4.0
## [122] biovizBase_1.30.1
## [123] kernlab_0.9-27
## [124] colorspace_1.4-0
## [125] XML_3.98-1.17
## [126] statmod_1.4.30
## [127] kpmt_0.1.0
## [128] multtest_2.38.0
## [129] shinythemes_1.1.2
## [130] flexmix_2.3-15
## [131] plotly_4.8.0
## [132] xtable_1.8-3
## [133] jsonlite_1.6
## [134] modeltools_0.2-22
## [135] R6_2.4.0
```

```
## [136] Hmisc_4.2-0
## [137] pillar_1.3.1
## [138] htmltools_0.3.6
## [139] mime_0.6
## [140] glue_1.3.0
## [141] class_7.3-15
## [142] beanplot_1.2
## [143] codetools_0.2-16
## [144] mvtnorm_1.0-8
## [145] lattice_0.20-38
## [146] tibble_2.0.1
## [147] sva_3.30.1
## [148] curl_3.3
## [149] BiasedUrn_1.07
## [150] gtools_3.8.1
## [151] GO.db_3.7.0
## [152] openssl_1.2.1
## [153] survival_2.43-3
## [154] rmarkdown_1.11
## [155] fastICA_1.2-1
## [156] munsell_0.5.0
## [157] rhdf5_2.26.2
## [158] GenomeInfoDbData_1.2.0
## [159] goseq_1.34.1
## [160] HDF5Array_1.10.1
## [161] gtable_0.2.0
```

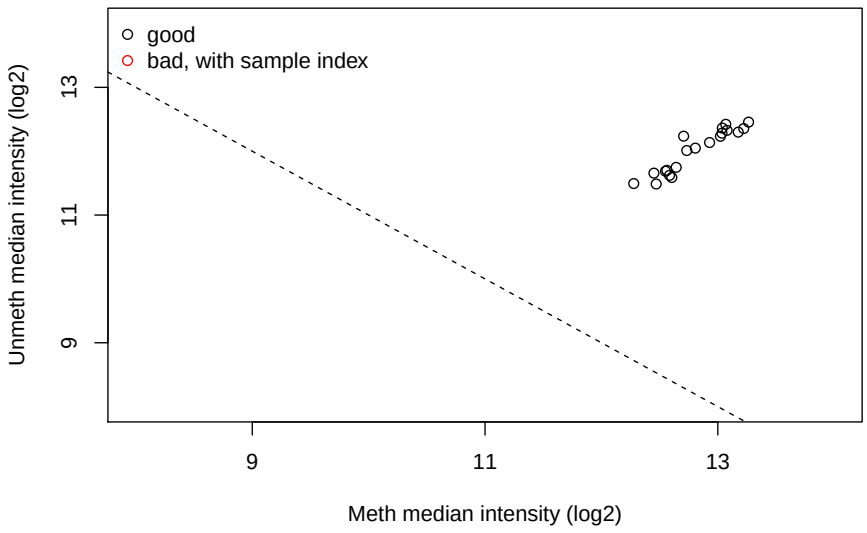
1 Read Data

```
baseDir1 = "/BIOS6660/Methylation/plate1"
targets1 = read.metharray.sheet(baseDir1)

## [read.metharray.sheet] Found the following CSV files:

## [1] "/BIOS6660/Methylation/plate1/selected_plate1.csv"

baseDir2 = "/BIOS6660/Methylation/plate2"
targets2 = read.metharray.sheet(baseDir2)

## [read.metharray.sheet] Found the following CSV files:

## [1] "/BIOS6660/Methylation/plate2/selected_plate2.csv"
```

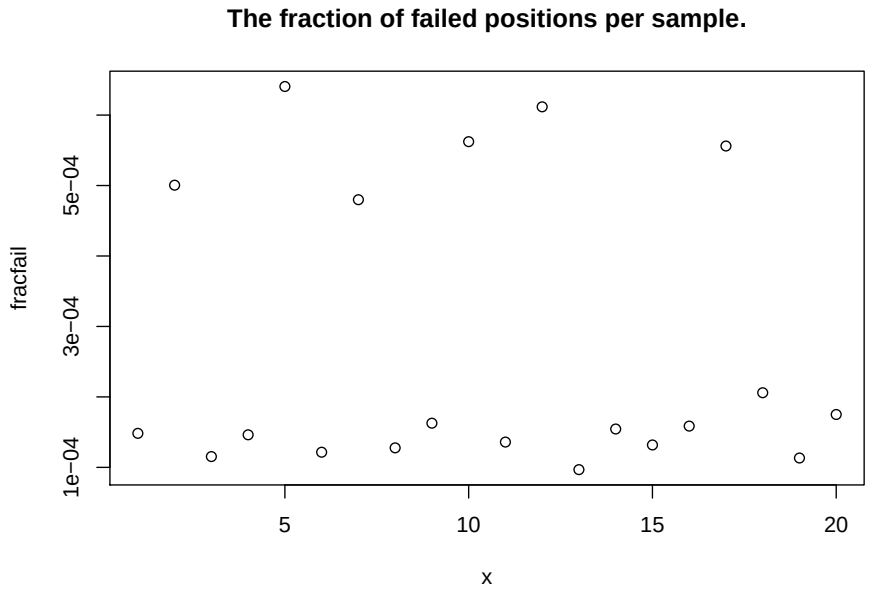
```
targets = rbind(targets1, targets2)

rgSet = read.metharray.exp(targets=targets, extended=T)
sampleNames(rgSet) = rgSet[[1]]
rgSet

## class: RGChannelSetExtended
## dim: 622399 20
## metadata(0):
## assays(5): Green Red GreenSD RedSD NBeads
## rownames(622399): 10600313 10600322 ... 74810490 74810492
## rowData names(0):
## colnames(20): 20209 20216 ... 20075 20071
## colData names(9): Sample_Name Sample_Well ... Basename filenames
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19

getManifest(rgSet)

## Loading required package:  IlluminaHumanMethylation450kmanifest

## IlluminaMethylationManifest object
## Annotation
##   array: IlluminaHumanMethylation450k
## Number of type I probes: 135476
## Number of type II probes: 350036
## Number of control probes: 850
## Number of SNP type I probes: 25
## Number of SNP type II probes: 40

clindat = read.table("/BIOS6660/Methylation/demographic.txt",
                     sep="\t", header=T)
table(clindat$Exposure, clindat$child_sex)

##
##     F M
##   0 3 7
##   1 3 7

stopifnot(all(clindat$Sample_Name==rgSet$Sample_Name))
pData(rgSet)$Sample_Group = clindat$Exposure
pData(rgSet)$child_sex = clindat$child_sex
```

2 Detection P value

```
mset = preprocessRaw(rgSet)
mset

## class: MethylSet
## dim: 485512 20
## metadata(0):
## assays(2): Meth Unmeth
## rownames(485512): cg00050873 cg00212031 ... ch.22.47579720R
##   ch.22.48274842R
## rowData names(0):
## colnames(20): 20209 20216 ... 20075 20071
## colData names(11): Sample_Name Sample_Well ... Sample_Group
##   child_sex
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19
## Preprocessing
##   Method: Raw (no normalization or bg correction)
##   minfi version: 1.28.3
##   Manifest version: 0.4.0

qc = getQC(mset)
plotQC(qc)
```




```
detP = detectionP(rgSet)
detPcut = 0.05

failed = detP > detPcut
fracfail = colMeans(failed)
main = paste("The fraction of failed positions per sample.")
x = seq(1, length(fracfail), 1)
plot(x, fracfail, main=main)
```




```
removeDetP = 0.1
badProbes = rowMeans(failed) > removeDetP

mset.f = mset[!badProbes,]
mset.f

## class: MethylSet
## dim: 485139 20
## metadata(0):
## assays(2): Meth Unmeth
## rownames(485139): cg00213748 cg00455876 ... ch.22.47579720R
##   ch.22.48274842R
## rowData names(0):
## colnames(20): 20209 20216 ... 20075 20071
## colData names(11): Sample_Name Sample_Well ... Sample_Group
##   child_sex
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19
## Preprocessing
##   Method: Raw (no normalization or bg correction)
##   minfi version: 1.28.3
##   Manifest version: 0.4.0

mset = mset.f
```

```
message("There are ", sum(badProbes),
        " bad probes with high detection P values removed.")
```

```
## There are 373 bad probes with high detection P values removed.
```

3 Check bead count

```
beadCutoff = 0.1
bc = beadcount(rgSet)
quantile(bc, na.rm=T)
```

```
##   0%  25%  50%  75% 100%
##    3   11   14   17  108
```

```
bc2 = bc[rowSums(is.na(bc)) < beadCutoff*(ncol(bc)), ]
mset.f2 = mset[featureNames(mset) %in% row.names(bc2), ]
mset.f2
```

```
## class: MethylSet
## dim: 484561 20
## metadata(0):
## assays(2): Meth Unmeth
## rownames(484561): cg00213748 cg00455876 ... ch.22.47579720R
##   ch.22.48274842R
## rowData names(0):
## colnames(20): 20209 20216 ... 20075 20071
## colData names(11): Sample_Name Sample_Well ... Sample_Group
##   child_sex
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19
## Preprocessing
##   Method: Raw (no normalization or bg correction)
##   minfi version: 1.28.3
##   Manifest version: 0.4.0
```

```
message("Filtering probes with a beadcount <3 in at least ",
        beadCutoff*100, "% of samples has removed ", dim(mset)[1]-
          dim(mset.f2)[1], " from the analysis.")
```

```
## Filtering probes with a beadcount <3 in at least 10% of samples
has removed 578 from the analysis.
```

```
mset = mset.f2
```

4 Check non-CG probes

```
mset.cg = dropMethylationLoci(mset, dropCH=T)
mset.cg
```

```
## class: MethylSet
## dim: 481471 20
## metadata(0):
## assays(2): Meth Unmeth
## rownames(481471): cg00213748 cg00455876 ... cg27662611 cg27665648
## rowData names(0):
## colnames(20): 20209 20216 ... 20075 20071
## colData names(11): Sample_Name Sample_Well ... Sample_Group
##   child_sex
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19
## Preprocessing
##   Method: Raw (no normalization or bg correction)
##   minfi version: 1.28.3
##   Manifest version: 0.4.0
```

```
message("There are ", dim(mset)[1]-dim(mset.cg)[1],
        " non-CG probes. Keep them in the final analysis dataset.")
```

```
## There are 3090 non-CG probes.  Keep them in the final analysis dataset.
```

5 Map to the genome

```
gset = mapToGenome(mset)
gset
```

```
## class: GenomicMethylSet
## dim: 484561 20
## metadata(0):
## assays(2): Meth Unmeth
## rownames(484561): cg13869341 cg14008030 ... cg21106100 cg08265308
## rowData names(0):
## colnames(20): 20209 20216 ... 20075 20071
## colData names(11): Sample_Name Sample_Well ... Sample_Group
##   child_sex
## Annotation
##   array: IlluminaHumanMethylation450k
```

```
##   annotation: ilmn12.hg19
## Preprocessing
##   Method: Raw (no normalization or bg correction)
##   minfi version: 1.28.3
##   Manifest version: 0.4.0
```

```
annotation = getAnnotation(gset, dropNonMapping=F)
names(annotation)
```

```
##  [1] "chr"                      "pos"
##  [3] "strand"                   "Name"
##  [5] "AddressA"                 "AddressB"
##  [7] "ProbeSeqA"                "ProbeSeqB"
##  [9] "Type"                     "NextBase"
## [11] "Color"                    "Probe_rs"
## [13] "Probe_maf"                "CpG_rs"
## [15] "CpG_maf"                  "SBE_rs"
## [17] "SBE_maf"                  "Islands_Name"
## [19] "Relation_to_Island"       "Forward_Sequence"
## [21] "SourceSeq"                "Random_Loci"
## [23] "Methyl27_Loci"            "UCSC_RefGene_Name"
## [25] "UCSC_RefGene_Accession"   "UCSC_RefGene_Group"
## [27] "Phantom"                  "DMR"
## [29] "Enhancer"                 "HMM_Island"
## [31] "Regulatory_Feature_Name"  "Regulatory_Feature_Group"
## [33] "DHS"
```

```
table(annotation$chr)
```

```
##
##  chr1 chr10 chr11 chr12 chr13 chr14 chr15 chr16 chr17 chr18 chr19  chr2
## 46792 24360 28760 24497 12268 15053 15246 21941 27832  5915 25486 34769
## chr20 chr21 chr22  chr3  chr4  chr5  chr6  chr7  chr8  chr9  chrX  chrY
## 10363  4240  8526 25114 20433 24291 36523 29972 20915  9853 11216   196
```

```
dim(annotation)
```

```
## [1] 484561     33
```

```
annotation2 = getAnnotation(gset, dropNonMapping=T)
dim(annotation2)
```

```
## [1] 484561     33
```

```
message("There are ", dim(annotation)[1]-dim(annotation2)[1],
        " non-mapping probes.")
```

```
## There are 0 non-mapping probes.
```

```
autosomes = annotation[!annotation$chr %in% c("chrX", "chrY"), ]
allosomes = annotation[annotation$chr %in% c("chrX", "chrY"), ]
```

6 Identify probes with SNP

```
gset = addSnpInfo(gset)

getAnnotationObject(gset)

## IlluminaMethylationAnnotation object
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12
##   genomeBuild: hg19
## Available annotation
##   Islands.UCSC
##   Locations
##   Manifest
##   Other
##   SNPs.132CommonSingle
##   SNPs.135CommonSingle
##   SNPs.137CommonSingle
##   SNPs.138CommonSingle
##   SNPs.141CommonSingle
##   SNPs.142CommonSingle
##   SNPs.144CommonSingle
##   SNPs.146CommonSingle
##   SNPs.147CommonSingle
##   SNPs.Illumina
## Defaults
##   Locations
##   Manifest
##   SNPs.137CommonSingle
##   Islands.UCSC
##   Other

gset.f = dropLociWithSnps(gset, snps=c("SBE", "CpG"), maf=0)

message("The number of probes with snps is ", dim(gset)[1]-
          dim(gset.f)[1], ". Keep them for now.")

## The number of probes with snps is 17438.  Keep them for now.
```

7 Plot raw β and M values

```r
beta.raw = getBeta(gset)
M.raw = getM(gset)
colnames(beta.raw) = sampleNames(mset)
colnames(M.raw) = sampleNames(mset)

champ.SVD(beta=beta.raw, pd=pData(rgSet),
          resultsDir = paste(getwd(), "resultsChamp1", sep = "/"))

## [============================]
## [<<< ChAMP.SVD START >>>]
## ----------------
## champ.SVD Results will be saved in /home/murphjes/BIOS6660/Homework_10/resultsChamp1
.
## 35 NA are detected in your beta Data Set, which may cause fail or
uncorrect of SVD analysis.  You may want to impute NA with champ.impute()
function first.
## [SVD analysis will be proceed with 484561 probes and 20 samples.]
##
## [ champ.SVD() will only check the dimensions between data and pd,
instead if checking if Sample_Names are correctly matched (because
some user may have no Sample_Names in their pd file),thus please make
sure your pd file is in accord with your data sets (beta) and (rgSet).]
## « Following Factors in your pd(sample_sheet.csv) will be analysised:
»
## <Sample_Well>(character):D01, D02, D04, D05, D06, D07, D08, D10,
D11, D12, F01, F02, F04, F05, F06, F07, F08, F10, F11, F12
## <Sample_Plate>(character):Plate 1, Plate 2
## <Array>(character):R01C01, R02C01, R04C01, R05C01, R06C01, R01C02,
R02C02, R04C02, R05C02, R06C02
## <Slide>(character):9721366035, 9992576163
## <Sample_Group>(integer):1, 0
## <child_sex>(factor):M, F
## [champ.SVD have automatically select ALL factors contain at least
two different values from your pd(sample_sheet.csv), if you don't want
to analysis some of them, please remove them manually from your pd
variable then retry champ.SVD().]
##
## « Following Factors in your pd(sample_sheet.csv) will not be analysis:
»
## <Sample_Name>
## <Sample.Group>
## <Pool_ID>
## <Basename>
## <filenames>
```

```
## [Factors are ignored because they only indicate Name or Project,
or they contain ONLY ONE value across all Samples.]
##
## « PhenoTypes.lv generated successfully.  »
## Error in while (step == 0) {:  missing value where TRUE/FALSE needed
```

```
gset.auto = gset[featureNames(gset) %in% row.names(autosomes), ]
beta.raw.auto = getBeta(gset.auto)
M.raw.auto = getM(gset.auto)

champ.SVD(beta=beta.raw.auto, pd=pData(rgSet),
          resultsDir = paste(getwd(), "resultsChamp1", sep = "/"))
```

```
## [==========================]
## [<<< ChAMP.SVD START >>>]
## ---------------
## champ.SVD Results will be saved in /home/murphjes/BIOS6660/Homework_10/resultsChamp1
.
## 35 NA are detected in your beta Data Set, which may cause fail or
uncorrect of SVD analysis.  You may want to impute NA with champ.impute()
function first.
## [SVD analysis will be proceed with 473149 probes and 20 samples.]
##
## [ champ.SVD() will only check the dimensions between data and pd,
instead if checking if Sample_Names are correctly matched (because
some user may have no Sample_Names in their pd file),thus please make
sure your pd file is in accord with your data sets (beta) and (rgSet).]
## « Following Factors in your pd(sample_sheet.csv) will be analysised:
»
## <Sample_Well>(character):D01, D02, D04, D05, D06, D07, D08, D10,
D11, D12, F01, F02, F04, F05, F06, F07, F08, F10, F11, F12
## <Sample_Plate>(character):Plate 1, Plate 2
## <Array>(character):R01C01, R02C01, R04C01, R05C01, R06C01, R01C02,
R02C02, R04C02, R05C02, R06C02
## <Slide>(character):9721366035, 9992576163
## <Sample_Group>(integer):1, 0
## <child_sex>(factor):M, F
## [champ.SVD have automatically select ALL factors contain at least
two different values from your pd(sample_sheet.csv), if you don't want
to analysis some of them, please remove them manually from your pd
variable then retry champ.SVD().]
##
## « Following Factors in your pd(sample_sheet.csv) will not be analysis:
»
## <Sample_Name>
## <Sample.Group>
```

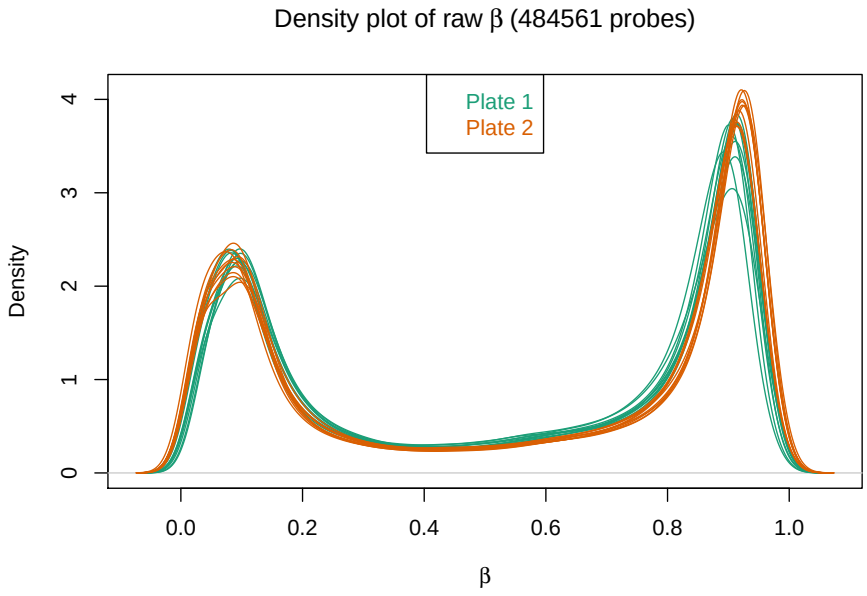
```
## <Pool_ID>
## <Basename>
## <filenames>
## [Factors are ignored because they only indicate Name or Project,
or they contain ONLY ONE value across all Samples.]
##
## « PhenoTypes.lv generated successfully.  »
## Error in while (step == 0) {:  missing value where TRUE/FALSE needed
```

```
champ.SVD(beta=M.raw.auto, pd=pData(rgSet),
          resultsDir = paste(getwd(), "resultsChamp1", sep = "/"))
```

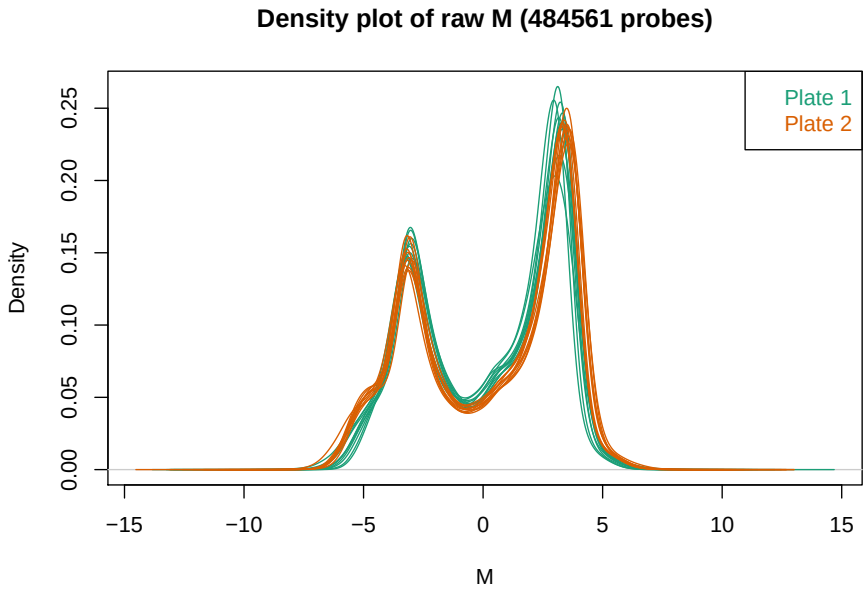
```
## [============================]
## [«« ChAMP.SVD START »»]
## ----------------
## champ.SVD Results will be saved in /home/murphjes/BIOS6660/Homework_10/resultsChamp1
.
## 35 NA are detected in your beta Data Set, which may cause fail or
uncorrect of SVD analysis.  You may want to impute NA with champ.impute()
function first.
## [SVD analysis will be proceed with 473149 probes and 20 samples.]
##
## [ champ.SVD() will only check the dimensions between data and pd,
instead if checking if Sample_Names are correctly matched (because
some user may have no Sample_Names in their pd file),thus please make
sure your pd file is in accord with your data sets (beta) and (rgSet).]
## « Following Factors in your pd(sample_sheet.csv) will be analysised:
»
## <Sample_Well>(character):D01, D02, D04, D05, D06, D07, D08, D10,
D11, D12, F01, F02, F04, F05, F06, F07, F08, F10, F11, F12
## <Sample_Plate>(character):Plate 1, Plate 2
## <Array>(character):R01C01, R02C01, R04C01, R05C01, R06C01, R01C02,
R02C02, R04C02, R05C02, R06C02
## <Slide>(character):9721366035, 9992576163
## <Sample_Group>(integer):1, 0
## <child_sex>(factor):M, F
## [champ.SVD have automatically select ALL factors contain at least
two different values from your pd(sample_sheet.csv), if you don't want
to analysis some of them, please remove them manually from your pd
variable then retry champ.SVD().]
##
## « Following Factors in your pd(sample_sheet.csv) will not be analysis:
»
## <Sample_Name>
## <Sample.Group>
## <Pool_ID>
```

```
## <Basename>
## <filenames>
## [Factors are ignored because they only indicate Name or Project,
or they contain ONLY ONE value across all Samples.]
##
## « PhenoTypes.lv generated successfully.  »
## Error in while (step == 0) {:  missing value where TRUE/FALSE needed
```

```
totalProbes = dim(beta.raw)[1]
main1 = bquote("Density plot of raw"~beta~"("*.(totalProbes)~"probes)")
pal = brewer.pal(8, "Dark2")
densityPlot(beta.raw, sampGroups=pData(gset)$Sample_Plate,
            main=main1, xlab=expression(beta), legend=F)
legend("top", legend=levels(as.factor(pData(gset)$Sample_Plate)),
       text.col=pal)
```




```
main2 <- paste("Density plot of raw M (", totalProbes,
               " probes)", sep="")
densityPlot(M.raw, sampGroups=pData(gset)$Sample_Plate,
            main=main2, xlab="M", legend=F)
legend("topright", legend=levels(as.factor(pData(gset)$Sample_Plate)),
       text.col=pal)
```


Density plot of raw M (484561 probes)
Plate 1
Plate 2
Density
0.00
0.05
0.10
0.15
0.20
0.25
−15
−10
−5
0
5
10
15
M