missMethyl: Analysing Illumina HumanMethylation450 BeadChip Data

Belinda Phipson and Jovana Maksimovic

Modified: 4 September, 2014. Compiled: October 13, 2014

Contents

1 Introduction

The *missMethyl* package contains functions to analyse methylation data from Illumina's HumanMethylation450 beadchip. These arrays are a cost-effective alternative to whole genome bisulphite sequencing, and as such are widely used to profile DNA methylation. Specifically, missMethyl contains functions to perform SWAN normalisation [6] and differential variability analysis [7]. As our lab's research into specialised analyses of these arrays continues we anticipate that the package will be continuously updated with new functions.

Raw data files are in IDAT format, which can be read into R using the *minfi* package [1]. Statistical analyses are usually performed on M-values, and β values are used for visualisation, both of which can be extracted from *MethylSet* objects, which is a class of object created by *minfi*. For detecting differentially variable CpGs we recommend that the analysis is performed on M-values. All analyses described here are performed at the CpG site level.

2 Reading data into R

We will use the data in the *minfiData* package to demonstrate the functions in *missMethyl*. The example dataset has 6 samples across two slides. The sample information is in the targets file. An essential column in the targets file is the "Basename" column which tells *minfi* where the idat files to be read in are located. The R commands to read in the data

are taken from the *minfi* Users Guide. For additional details on how to read the IDAT files into R, as well as information regarding quality control please refer to the *minfi* Userś Guide.

```
> library(missMethyl)
> library(limma)
> library(minfi)
> library(minfiData)
> baseDir <- system.file("extdata", package = "minfiData")
> targets <- read.450k.sheet(baseDir)
```

```
[read.450k.sheet] Found the following CSV files:
[1] "/Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/SampleSheet.csv"
```

```
> targets[,1:9]
```

```
  Sample_Name Sample_Well Sample_Plate Sample_Group Pool_ID person age sex status
1    GroupA_3          H5           NA       GroupA      NA    id3  83   M normal
2    GroupA_2          D5           NA       GroupA      NA    id2  58   F normal
3    GroupB_3          C6           NA       GroupB      NA    id3  83   M cancer
4    GroupB_1          F7           NA       GroupB      NA    id1  75   F cancer
5    GroupA_1          G7           NA       GroupA      NA    id1  75   F normal
6    GroupB_2          H7           NA       GroupB      NA    id2  58   F cancer
```

```
> targets[,10:12]
```

```
   Array      Slide
1 R02C02 5723646052
2 R04C01 5723646052
3 R05C02 5723646052
4 R04C02 5723646053
5 R05C02 5723646053
6 R06C02 5723646053
                                                                                                    Base
1 /Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/5723646052/5723646052_R0
2 /Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/5723646052/5723646052_R0
3 /Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/5723646052/5723646052_R0
4 /Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/5723646053/5723646053_R0
5 /Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/5723646053/5723646053_R0
6 /Library/Frameworks/R.framework/Versions/3.1/Resources/library/minfiData/extdata/5723646053/5723646053_R0
```

```
> rgSet <- read.450k.exp(base = baseDir,targets = targets)
```

The data is now an *RGChannelSet* object and needs to be normalised and converted to a *MethylSet* object.

3 Subset-quantile within array normalization (SWAN)

SWAN (subset-quantile within array normalization) is a within-array normalization method for Illumina 450k BeadChips. Technical differencs have been demonstrated to exist between the Infinium I and Infinium II assays on a single 450K array [2, 3]. Using the SWAN method substantially reduces the technical variability between the assay designs whilst maintaining the important biological differences. The SWAN method makes the assumption that the number of CpGs within the 50bp probe sequence reflects the underlying biology of the region being interrogated. Hence, the overall distribution of intensities of probes with the same number of CpGs in the probe body should be the same regardless of assay type. The method then uses a subset quantile normalization approach to adjust the intensities of each array [6].

SWAN can take a *MethylSet*, *RGChannelSet* or *MethyLumiSet* as input. It should be noted that, in order to create the normalization subset, SWAN randomly selects Infinium I and II probes that have one, two and three underlying CpGs;

as such, we recommend using `set.seed` before `SWAN` to ensure that the normalized intensities will be identical, if the normalization is repeated.

The technical differences between Infinium I and II assay designs can result in aberrant beta value distributions (Figure 1, panel "Raw"). Using SWAN corrects for the technical differences between the Infinium I and II assay designs and produces a smoother overall β value distribution (Figure 1, panel "SWAN").

```
> mSet <- preprocessRaw(rgSet)
> mSetSw <- SWAN(mSet,verbose=TRUE)
```

```
[SWAN] Preparing normalization subset
[SWAN] Normalizing methylated channel
[SWAN] Normalizing array 1 of 6
[SWAN] Normalizing array 2 of 6
[SWAN] Normalizing array 3 of 6
[SWAN] Normalizing array 4 of 6
[SWAN] Normalizing array 5 of 6
[SWAN] Normalizing array 6 of 6
[SWAN] Normalizing unmethylated channel
[SWAN] Normalizing array 1 of 6
[SWAN] Normalizing array 2 of 6
[SWAN] Normalizing array 3 of 6
[SWAN] Normalizing array 4 of 6
[SWAN] Normalizing array 5 of 6
[SWAN] Normalizing array 6 of 6
```

```
> par(mfrow=c(1,2), cex=1.25)
> densityByProbeType(mSet[,1], main = "Raw")
> densityByProbeType(mSetSw[,1], main = "SWAN")
```

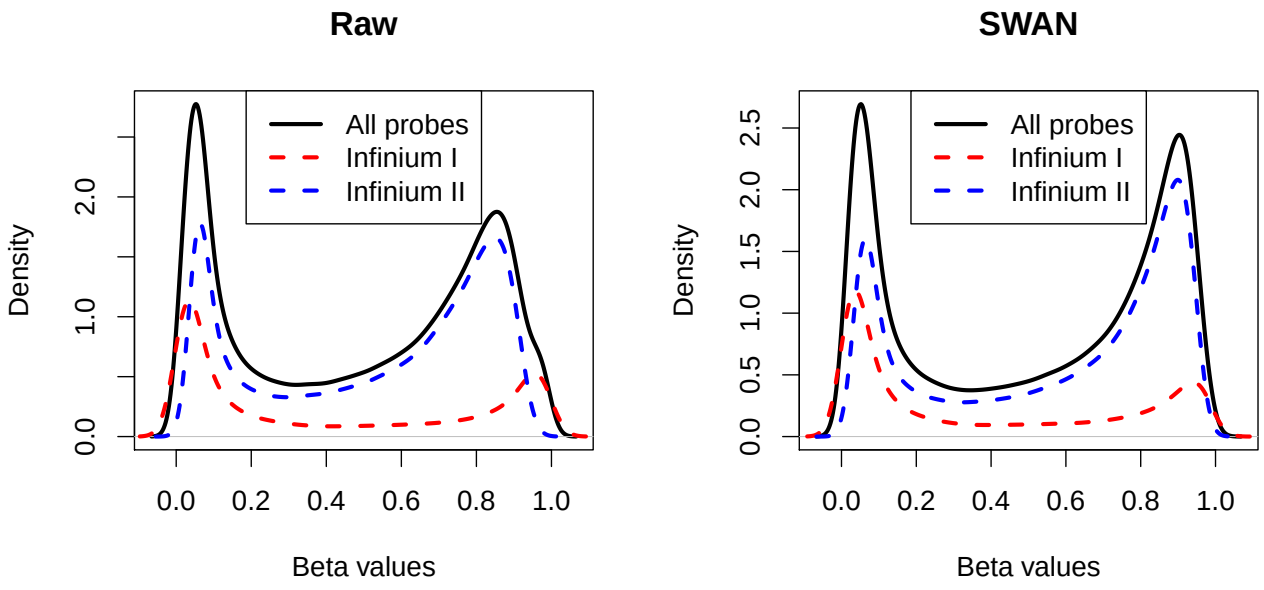

Figure 1: Density distributions of β values before and after using SWAN.

4 Filter out poor quality probes

Poor quality probes can be filtered out based on the detection p-value. For this example, to retain a CpG for further analysis, we require that the detection p-value is less than 0.01 in all samples.

```
> detP <- detectionP(rgSet)
> keep <- rowSums(detP < 0.01) == ncol(rgSet)
> mSet <- mSet[keep,]
```

5 Extracting β and M-values

Now that the data has been SWAN normalised we can extract β and M-values from the *MethylSet* object. We prefer to add an offset to the methylated and unmethylated intensities when calculating M-values, hence we extract the methylated and unmethylated channels separately and perform our own calculation. For all subsequent analysis we use a random selection of 20 000 CpGs to reduce computation time.

```
> mset_reduced <- mSet[sample(1:nrow(mSet), 20000),]
> meth <- getMeth(mset_reduced)
> unmeth <- getUnmeth(mset_reduced)
> Mval <- log2((meth + 100)/(unmeth + 100))
> beta <- getBeta(mset_reduced)
> dim(Mval)
```

```
[1] 20000     6
```

An MDS plot (Figure 2) is a good sanity check to make sure samples cluster together according to the main factor of interest, in this case, cancer and normal.

6 Testing for differential methylation

To test for differential methylation we use the *limma* package [9], which employs an empirical Bayes framework based on Guassian model theory. First we need to set up the design matrix. There are a number of ways to do this, the most straightforward is directly from the targets file. There are a number of variables, with the status column indicating cancer/normal samples. From the person column of the targets file, we see that the cancer/normal samples are matched, with 3 individuals each contributing both a cancer and normal sample. Since the limma model framework can handle any experimental design which can be summarised by a design matrix, we can take into account the paired nature of the data in the analysis. For more complicated experimental designs, please refer to the *limma* Userś Guide.

```
> library(limma)
> group <- factor(targets$status,levels=c("normal","cancer"))
> id <- factor(targets$person)
> design <- model.matrix(~id + group)
> design
```

```
  (Intercept) idid2 idid3 groupcancer
1           1     0     1           0
2           1     1     0           0
3           1     0     1           1
4           1     0     0           1
5           1     0     0           0
6           1     1     0           1
attr(,"assign")
[1] 0 1 1 2
```

```
> par(mfrow=c(1,1))
> plotMDS(Mval, labels=targets$Sample_Name, col=as.integer(factor(targets$status)))
> legend("topleft",legend=c("Cancer","Normal"),pch=16,cex=1.2,col=1:2)
```

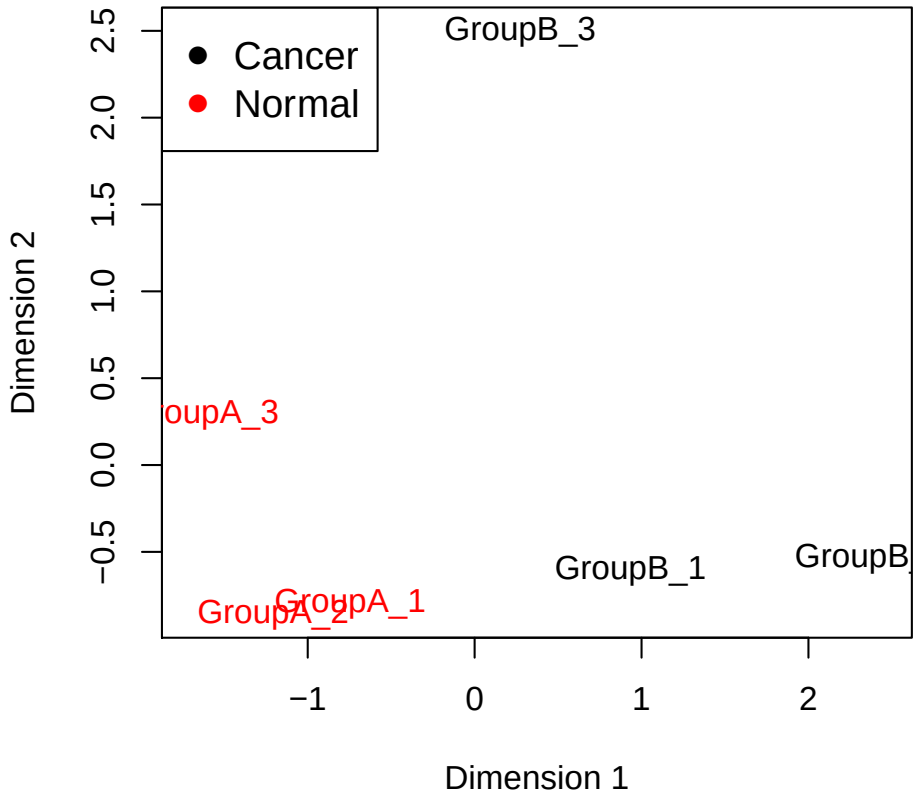

Figure 2: A multi-dimensional scaling (MDS) plot of cancer and normal samples.

```
attr(,"contrasts")
attr(,"contrasts")$id
[1] "contr.treatment"

attr(,"contrasts")$group
[1] "contr.treatment"
```

Now we can test for differential methylation using the `lmFit` and `eBayes` functions from *limma*. As input data we use the matrix of M-values.

```
> fit <- lmFit(Mval,design)
> fit <- eBayes(fit)
```

The numbers of hyper-methylated (1) and hypo-methylated (-1) can be displayed using the `decideTests` function in *limma* and the top 10 differentially methylated CpGs for cancer versus normal outputted using `topTable`.

```
> summary(decideTests(fit))
```

```
   (Intercept) idid2 idid3 groupcancer
-1        7232     0    97         473
0         3570 20000 19891       19076
1         9198     0    12         451
```

```
> top<-topTable(fit,coef=4)
> top
```

```
              logFC    AveExpr        t      P.Value  adj.P.Val        B
cg20078466 5.585935 -1.1518941 18.37313 5.319893e-06 0.02569476 4.470937
cg16423505 5.551619 -1.0717620 17.84744 6.190945e-06 0.02569476 4.372146
cg24588375 4.557029 -1.6639952 16.79764 8.493796e-06 0.02569476 4.158012
cg22473620 4.265443 -0.3420890 15.93455 1.117879e-05 0.02569476 3.963277
cg02012576 4.541847 -1.4459294 15.90667 1.128111e-05 0.02569476 3.956680
cg24115221 3.825275 -0.6682089 15.22962 1.414324e-05 0.02569476 3.790195
cg15732851 4.320273 -1.0953041 14.92630 1.570041e-05 0.02569476 3.711502
cg17802600 3.630589 -0.5195973 14.00491 2.184423e-05 0.02569476 3.455451
cg13056495 4.042332 -1.4759564 13.58631 2.555652e-05 0.02569476 3.329997
cg02584459 3.881264 -1.0860886 13.53103 2.610069e-05 0.02569476 3.312976
```

Note that since we performed our analysis on M-values, the logFC and AveExpr columns are computed on the M-value scale. For interpretability and visualisation we can look at the β values. The beta values for the top 4 differentially methylated CpGs shown in Figure 3.

7 Testing for differential variability (DiffVar)

7.1 Methylation data

Rather than testing for differences in mean methylation, we may be interested in testing for differences between group variances. For example, it has been hypothesised that highly variable CpGs in cancer are important for tumour progression [4]. Hence we may be interested in CpG sites that are consistently methylated in the normal samples, but variably methylated in the cancer samples. In general we recommend at least 10 samples in each group for accurate variance estimation, however for the purpose of this vignette we perform the analysis on 3 vs 3. In this example, we are interested in testing for differential variability in the cancer versus normal group. When we specify the *coef* parameter, which corresponds to the columns of the design matrix to be used for testing differential variability, we specify both the intercept and the fourth column. The ID variable is a nuisance parameter and not used when obtained the absolute deviations, however it can be included in the linear modelling step. For methylation data, the `varFit` function will take either a matrix of M-values, β values or a *MethylSet* object as input. If β values are supplied, a logit transformation is performed. Note that as a default, `varFit` uses the robust setting in the *limma* framework, which requires the use of the *statmod* package.

```
> fitvar <- varFit(Mval, design = design, coef = c(1,4))
```

The numbers of hyper-variable (1) and hypo-variable (-1) genes in cancer vs normal can be obtained using `decideTests`.

```
> summary(decideTests(fitvar))
```

```
   (Intercept) idid2 idid3 groupcancer
-1           0     0     4           4
0        19679 20000 19987       19979
1          321     0     9          17
```

```
> topDV <- topVar(fitvar, coef=4)
> topDV
```

```
           SampleVar LogVarRatio DiffLevene         t      P.Value Adj.P.Value
cg15898840  6.673549    3.167320   2.858034  5.406597 6.460049e-08 0.001292010
```

```
> cpgs <- rownames(top)
> par(mfrow=c(2,2))
> for(i in 1:4){
+ stripchart(beta[rownames(beta)==cpgs[i],]~design[,4],method="jitter",
+ group.names=c("Normal","Cancer"),pch=16,cex=1.5,col=c(4,2),ylab="Beta values",
+ vertical=TRUE,cex.axis=1.5,cex.lab=1.5)
+ title(cpgs[i],cex.main=1.5)
+ }
```

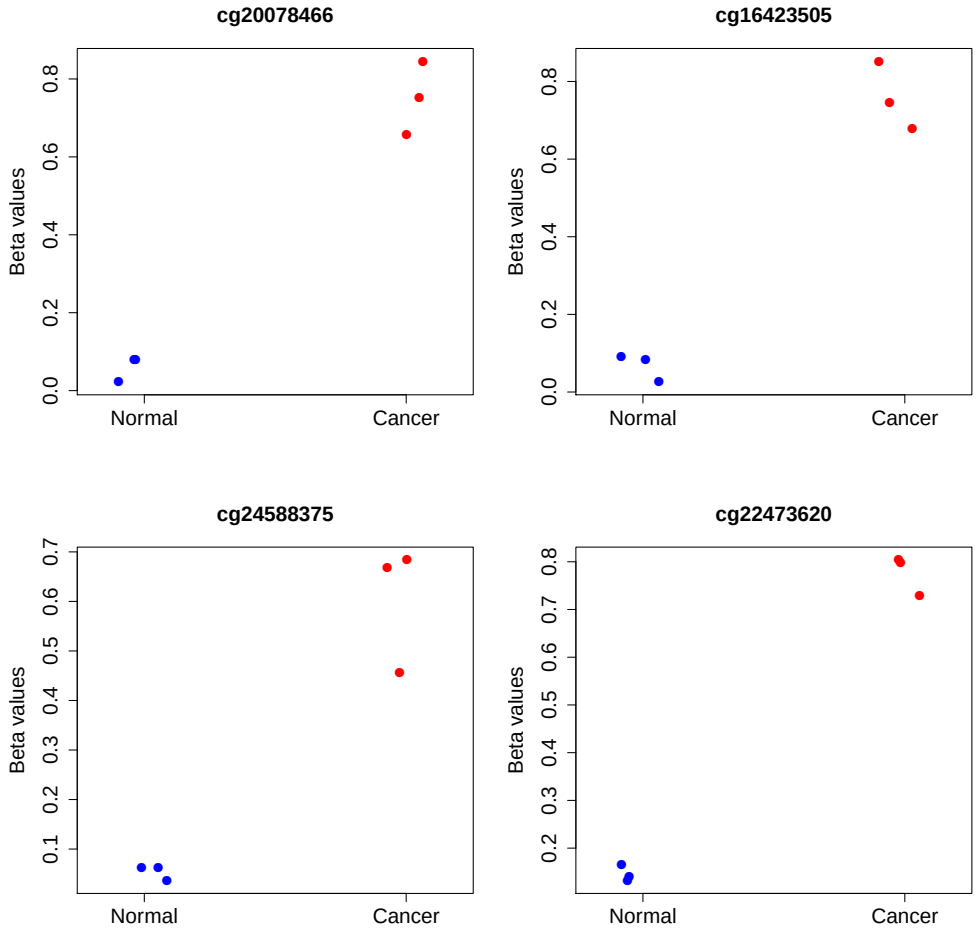

Figure 3: The β values for the top 4 differentially methylated CpGs.

```
cg09342610  7.199632   -3.415189  -2.896866 -5.158359 2.503100e-07 0.002503100
cg15627078  6.032070    5.038609   3.021186  4.867266 1.135830e-06 0.007572199
cg21396456  5.100657    6.650817   3.059418  4.775508 1.798878e-06 0.008994390
cg09260640  6.808946    3.084894   2.600895  4.582460 4.609241e-06 0.018436962
cg01644850  4.541455    7.410185   2.897981  4.411506 1.029219e-05 0.027647769
cg00420348  4.506495    3.843441   2.687509  4.374037 1.222777e-05 0.027647769
cg23543123  5.081260    6.078402   2.842241  4.360031 1.303685e-05 0.027647769
cg12521353  4.899705    2.962561   2.510225  4.333981 1.467936e-05 0.027647769
cg03884792  4.979631    6.890770   2.845655  4.326470 1.518851e-05 0.027647769
```

An alternate parameterisation of the design matrix that does not include an intercept term can also be used, and specific contrasts tested with `contrasts.varFit`. Here we specify the design matrix such that the first two columns correspond

to the normal and cancer groups, respectively.

```
> design2 <- model.matrix(~0+group+id)
> fitvar.contr <- varFit(Mval, design=design2, coef=c(1,2))
> contr <- makeContrasts(groupcancer-groupnormal,levels=colnames(design2))
> fitvar.contr <- contrasts.varFit(fitvar.contr,contrasts=contr)
```

The results are identical to before.

```
> summary(decideTests(fitvar.contr))
```

```
   groupcancer - groupnormal
-1                         4
0                      19979
1                         17
```

```
> topVar(fitvar.contr,coef=1)
```

```
           SampleVar LogVarRatio DiffLevene         t      P.Value Adj.P.Value
cg15898840  6.673549    3.167320   2.858034  5.406597 6.460049e-08 0.001292010
cg09342610  7.199632   -3.415189  -2.896866 -5.158359 2.503100e-07 0.002503100
cg15627078  6.032070    5.038609   3.021186  4.867266 1.135830e-06 0.007572199
cg21396456  5.100657    6.650817   3.059418  4.775508 1.798878e-06 0.008994390
cg09260640  6.808946    3.084894   2.600895  4.582460 4.609241e-06 0.018436962
cg01644850  4.541455    7.410185   2.897981  4.411506 1.029219e-05 0.027647769
cg00420348  4.506495    3.843441   2.687509  4.374037 1.222777e-05 0.027647769
cg23543123  5.081260    6.078402   2.842241  4.360031 1.303685e-05 0.027647769
cg12521353  4.899705    2.962561   2.510225  4.333981 1.467936e-05 0.027647769
cg03884792  4.979631    6.890770   2.845655  4.326470 1.518851e-05 0.027647769
```

The β values for the top 4 differentially variable CpGs can be seen in Figure 4.

7.2 RNA-Seq expression data

Testing for differential variability in expression data is straightforward if the technology is gene expression microarrays. The matrix of expression values can be supplied directly to the `varFit` function. For RNA-Seq data, the mean-variance relationship that occurs in count data needs to be taken into account. In order to deal with this issue, we apply a `voom` transformation [5] to obtain observation weights, which are then used in the linear modelling step. For RNA-Seq data, the `varFit` function will take a *DGEList* object as input.

To demonstrate this, we use data from the *tweeDEseqCountData* package. This data is part of the International HapMap project, consisting of RNA-Seq profiles from 69 unrelated Nigerian individuals [8]. The only covariate is gender, so we can look at differentially variable expression between males and females. We follow the code from the *limma* vignette to read in and process the data before testing for differential variability.

First we load up the data and extract the relevant information.

```
> library(tweeDEseqCountData)
> data(pickrell1)
> counts<-exprs(pickrell1.eset)
> dim(counts)
```

```
[1] 38415    69
```

```
> gender <- pickrell1.eset$gender
> table(gender)
```

```
gender
female   male
    40     29
```

```
> cpgsDV <- rownames(topDV)
> par(mfrow=c(2,2))
> for(i in 1:4){
+ stripchart(beta[rownames(beta)==cpgsDV[i],]~design[,4],method="jitter",
+ group.names=c("Normal","Cancer"),pch=16,cex=1.5,col=c(4,2),ylab="Beta values",
+ vertical=TRUE,cex.axis=1.5,cex.lab=1.5)
+ title(cpgsDV[i],cex.main=1.5)
+ }
```

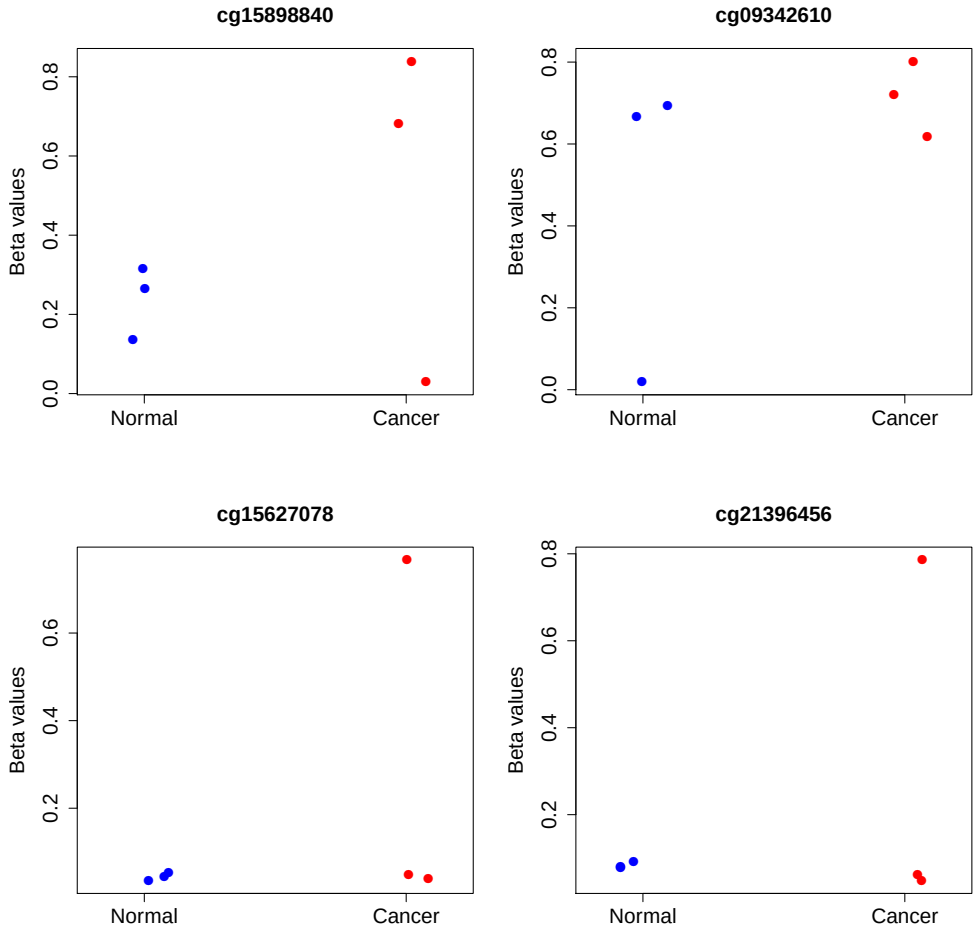

Figure 4: The β values for the top 4 differentially variable CpGs.

```
> rm(pickrell1.eset)
> data(genderGenes)
> data(annotEnsembl63)
> annot <- annotEnsembl63[,c("Symbol","Chr")]
> rm(annotEnsembl63)
```

We now have the counts, gender of each sample and annotation (gene symbol and chromosome) for each Ensemble gene. We can form a *DGEList* object using the *edgeR* package.

```
> library(edgeR)
> y <- DGEList(counts=counts, genes=annot[rownames(counts),])
```

We filter out lowly expressed genes by keeping genes with at least 1 count per million reads in at least 20 samples, as

well as genes that have defined annotation. Finally we perform scaling normalisation.

```
> isexpr <- rowSums(cpm(y)>1) >= 20
> hasannot <- rowSums(is.na(y$genes))==0
> y <- y[isexpr & hasannot,,keep.lib.sizes=FALSE]
> dim(y)
```

```
[1] 17310    69
```

```
> y <- calcNormFactors(y)
```

We set up the design matrix and test for differential variability. In this case there are no nuisance parameters, so *coef* does not need to be explicitly specified.

```
> design.hapmap <- model.matrix(~gender)
> fitvar.hapmap <- varFit(y, design = design.hapmap)
> fitvar.hapmap$genes <- y$genes
```

We can display the results of the test:

```
> summary(decideTests(fitvar.hapmap))
```

```
   (Intercept) gendermale
-1           0          2
0            0      17308
1        17310          0
```

```
> topDV.hapmap <- topVar(fitvar.hapmap,coef=ncol(design.hapmap))
> topDV.hapmap
```

```
                     Symbol Chr  SampleVar LogVarRatio DiffLevene         t      P.Value
ENSG00000213318 RP11-331F4.1  16 5.69839463   -2.562939 -0.9859943 -8.031075 7.238901e-12
ENSG00000129824       RPS4Y1   Y 2.32497726   -2.087025 -0.4585620 -4.957078 3.959119e-06
ENSG00000233864       TTTY15   Y 6.79004140   -2.245369 -0.6085233 -4.612613 1.497880e-05
ENSG00000176171        BNIP3  10 0.41317384    1.199292  0.3632133  4.219459 6.439806e-05
ENSG00000197358      BNIP3P1  14 0.39969125    1.149754  0.3353288  4.058182 1.147653e-04
ENSG00000025039        RRAGD   6 0.91837213    1.091229  0.4926839  3.977116 1.527080e-04
ENSG00000103671        TRIP4  15 0.07456448   -1.457139 -0.1520583 -3.911199 1.921645e-04
ENSG00000171100         MTM1   X 0.44049558   -1.133295 -0.3334619 -3.896210 2.024119e-04
ENSG00000149476          DAK  11 0.13289523   -1.470460 -0.1919880 -3.785886 2.956246e-04
ENSG00000064886       CHI3L2   1 2.70234584    1.468059  0.8449434  3.782149 2.994084e-04
                  Adj.P.Value
ENSG00000213318 1.253054e-07
ENSG00000129824 3.426618e-02
ENSG00000233864 8.642769e-02
ENSG00000176171 2.786826e-01
ENSG00000197358 3.973173e-01
ENSG00000025039 4.379688e-01
ENSG00000103671 4.379688e-01
ENSG00000171100 4.379688e-01
ENSG00000149476 4.428665e-01
ENSG00000064886 4.428665e-01
```

The log counts per million for the top 4 differentially variable genes can be seen in Figure 5.

```
> genesDV <- rownames(topDV.hapmap)
> par(mfrow=c(2,2))
> for(i in 1:4){
+ stripchart(cpm(y,log=TRUE)[rownames(y)==genesDV[i],]~design.hapmap[,ncol(design.hapmap)],method="jitter",
+ group.names=c("Female","Male"),pch=16,cex=1.5,col=c(4,2),ylab="Log counts per million",
+ vertical=TRUE,cex.axis=1.5,cex.lab=1.5)
+ title(genesDV[i],cex.main=1.5)
+ }
```

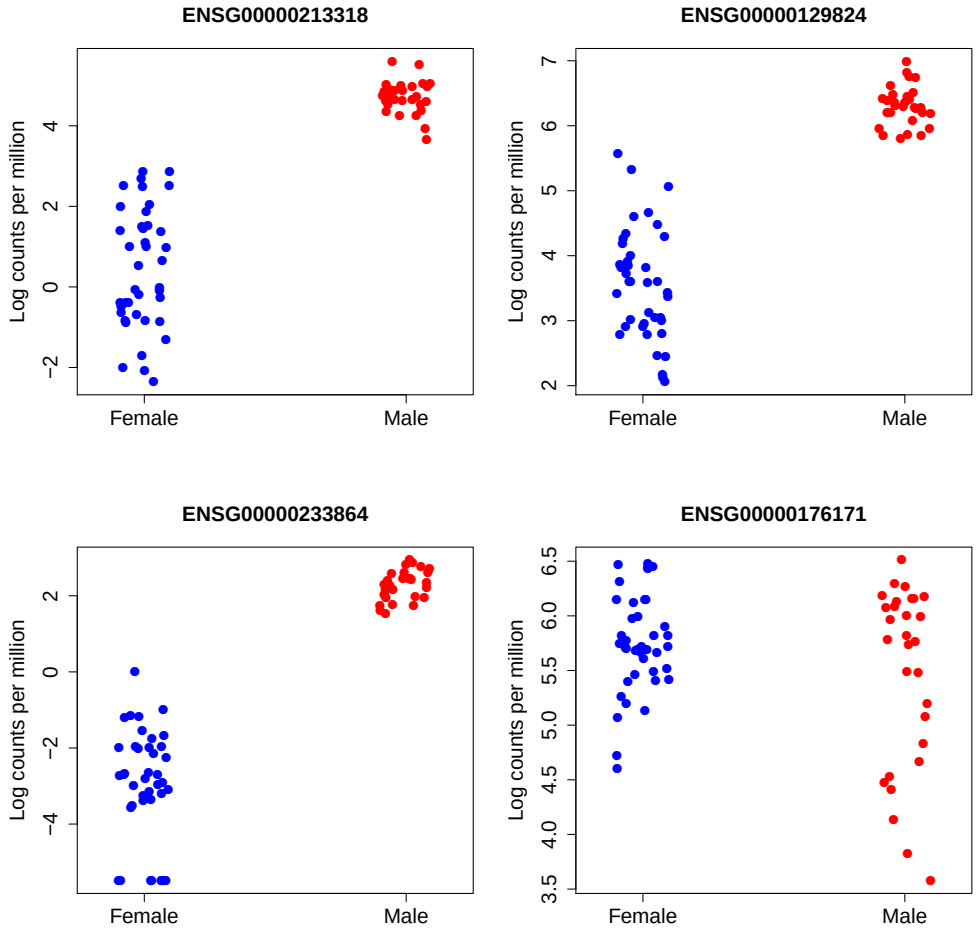

Figure 5: The log counts per million for the top 4 differentially variably expressed genes.

8 Session information

```
> toLatex(sessionInfo())
```

- R version 3.1.1 Patched (2014-09-25 r66681), x86_64-apple-darwin10.8.0
- Locale: en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
- Base packages: base, datasets, graphics, grDevices, methods, parallel, splines, stats, stats4, utils
- Other packages: Biobase 2.26.0, BiocGenerics 0.12.0, Biostrings 2.34.0, bumphunter 1.6.0, edgeR 3.8.0, foreach 1.4.2, GenomeInfoDb 1.2.0, GenomicRanges 1.18.0, IlluminaHumanMethylation450kanno.ilmn12.hg19 0.2.1, IlluminaHumanMethylation450kmanifest 0.4.0, IRanges 2.0.0, iterators 1.0.7, lattice 0.20-29, limma 3.22.0, locfit 1.5-9.1, minfi 1.12.0, minfiData 0.7.0,

missMethyl 1.0.0, S4Vectors 0.4.0, statmod 1.4.20, tweeDEseqCountData 1.3.0, XVector 0.6.0

- Loaded via a namespace (and not attached): annotate 1.44.0, AnnotationDbi 1.28.0, base64 1.1, beanplot 1.2, BiocStyle 1.4.0, codetools 0.2-9, colorspace 1.2-4, DBI 0.3.1, digest 0.6.4, doRNG 1.6, genefilter 1.48.0, grid 3.1.1, illuminaio 0.8.0, MASS 7.3-35, matrixStats 0.10.0, mclust 4.4, methylumi 2.12.0, multtest 2.22.0, nlme 3.1-118, nor1mix 1.2-0, pkgmaker 0.22, plyr 1.8.1, preprocessCore 1.28.0, quadprog 1.5-5, R.methodsS3 1.6.1, RColorBrewer 1.0-5, Rcpp 0.11.3, registry 0.2, reshape 0.8.5, rngtools 1.2.4, RSQLite 0.11.4, siggenes 1.40.0, stringr 0.6.2, survival 2.37-7, tools 3.1.1, XML 3.98-1.1, xtable 1.7-4, zlibbioc 1.12.0

References

[1] Aryee, M. J., Jaffe, A. E., Corrada-Bravo, H., Ladd-Acosta, C., Feinberg, A. P., Hansen, K. D., and Irizarry, R. A. (2014). Minfi: a flexible and comprehensive Bioconductor package for the analysis of Infinium DNA methylation microarrays. *Bioinformatics*, **30**(10):1363-1369.

[2] Bibikova, M., Barnes, B., Tsan, C., Ho, V., Klotzle, B, Le, J.M., Delano, D., Zhang, L., Schroth, G.P., Gunderson, K.L., Fan, J., and Shen, R. (2011). High density DNA methylation array with single CpG site resolution. *Genomics*, **98**(4):288¿295.

[3] Dedeurwaerder, S., Defrance, M., Calonne, E., Denis, H., Sotiriou, C., and Fuks, F. (2011). Evaluation of the Infinium Methylation 450K technology. *Epigenomics*, **3**(6):771-784.

[4] Hansen, K.D., Timp, W., Bravo, H.C., Sabunciyan, S., Langmead, B., McDonald, O.G., Wen, B., Wu, H., Liu, Y., Diep, D., Briem, E., Zhang, K., Irizarry, R., and Feinberg, A.P. (2011). Increased methylation variation in epigenetic domains across cancer types. *Nature Genetics*, **43**:768-75.

[5] Law, C.W., Chen, Y., Shi, W., Smyth, G.K. (2014). Voom: precision weights unlock linear model analysis tools for RNA-seq read counts. *Genome Biology*, 15, R29.

[6] Maksimovic, J., Gordon, L., and Oshlack, A. (2012). SWAN: Subset-quantile within array normalization for illumina infinium HumanMethylation450 BeadChips. *Genome Biology*, **13**(6):R44.

[7] Phipson, B., and Oshlack, A. DiffVar: A new method for detecting differential variability with application to methylation in cancer and ageing. *Genome Biology, Accepted 10 September 2014*.

[8] Pickrell, J.K., Marioni, J.C., Pai, A.A., Degner, J.F., Engelhardt, B.E., Nkadori, E., Veyrieras, J.B., Stephens, M., Gilad, Y., and Pritchard, J.K. (2010). Understanding mechanisms underlying human gene expression variation with RNA sequencing. *Nature*, 464, 768–72.

[9] Smyth, G.K. (2005). Limma: linear models for microarray data. In: 'Bioinformatics and Computational Biology Solutions using R and Bioconductor'. R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, W. Huber (eds), Springer, New York, pages 397-420.