The *wateRmelon* Package

Chloe CY Wong, Ruth Pidsley and Leonard C Schalkwyk

April 27, 2020

1   About the package

The *wateRmelon* package is designed to make it convenient to use the data quality metrics and normalization methods from our paper [1] as part of existing pipelines or work flows, and so as much as possible we have implemented S4 methods for *MethyLumiSet* objects (*methylumi* package), *MethylSet* and *RGChannelSet* objects (*minfi* package) and *exprmethy450* objects (*IMA* package).

In addition to our own functions, the package also contains functions by Matthieu Defrance [2] and Nizar Touleimat [3] and Andrew Teschendorff [4] as well as a wrapper for the SWAN method [5].

2   Installation

Because it is designed to work with several Bioconductor packages it unavoidably has many dependencies from CRAN as well as Bioconductor. In principle install.packages() installs dependencies automatically, but if there are problems you can install them by hand using the following commands:

```
> install.packages('ROCR', 'matrixStats')
> if (!requireNamespace("BiocManager", quietly=TRUE))
+     install.packages("BiocManager")
> BiocManager::install( 'limma', 'minfi',
+    'IlluminaHumanMethylation450kmanifest',
+    'methylumi', 'lumi')
```

Installing the latest package from a local copy (assuming it is in the current working directory of your R session):

```
> install.packages('wateRmelon_0.9.9.tar.gz', repos=NULL, type='source')
```

3 Trying it out

The package contains a small subset of 450K array data which can be used to explore functions quickly; the `melon` data set for example is a *MethyLumiSet* with 12 samples but only 3363 features:

```
> library('wateRmelon')
> # load in melon dataset
> data (melon)
> # display dimensions of data matrix
> dim(melon)

Features  Samples
    3363       12

> # quality filter using default thresholds
> melon.pf<-pfilter(melon)

0 samples having 1 % of sites with a detection p-value greater than 0.05 were removed
Samples removed:
72 sites were removed as beadcount <3 in 5 % of samples
40 sites having 1 % of samples with a detection p-value greater than 0.05 were remove

> # preprocess using our best method
> melon.dasen.pf <- dasen(melon.pf)
```

4 Our performance metrics

We have taken advantage of known DNA methylation patterns associated with genomic imprinting and X-chromosome inactivation (XCI), in addition to the performance of SNP genotyping assays present on the array, to derive three independent metrics which we use to test alternative schemes of correction and normalization. These metrics also have potential utility as quality scores for datasets. All of them are expressed in such a way that lower values indicate better performance (i.e. better predicted ability to detect real methylation differences between samples).

4.1 Genomic imprinting

This is based on the hemi-methylation of genomic imprinting differentially methylated regions (DMRs), and is a standard-error-like measure of dispersion(SE).

```
> # calculate iDMR metrics on QC'd betas
> dmrse_row(melon.pf)

[1] 0.005428861

> # calculate iDMR metrics on QC'd and preprocessed betas
> dmrse_row(melon.dasen.pf)

[1] 0.002086381

> # slightly lower (better) standard errors
```

4.2 SNP genotypes

A very simple genotype calling by one-dimensional K-means clustering is performed on each SNP, and for those SNPs where there are three genotypes represented the squared deviations are summed for each genotype (similar to a standard deviation for each of allele A homozygote, AB heterozygote and allele B homozygote). By default these are further divided by the square root of the number of samples to get a standard error-like statistic.

```
> # calculate SNP metrics on QC'd betas
> genki(melon.pf)

NULL
[1] 8.129585e-05 2.020173e-04 7.819409e-05

> # calculate SNP metrics on QC'd and preprocessed betas
> genki(melon.dasen.pf)

NULL
[1] 0.0000507474 0.0001255203 0.0000452997

> # slightly lower (better) standard errors
```

4.3 X-chromosome inactivation

This is based on the male-female difference in DNA methylation, almost all of which is due to hypermethylation of the inactive X in females. This difference is thus a good predictor of X-chromosome location for probes, and this can be used for a Receiver Operating Characteristic (ROC) curve analysis. We report 1-(area under curve) (AUC) so that smaller values

3

indicate better performance, just as in our other two metrics. This requires the samples to be of known sex (and not all the same) and chromosome assignments for all probes. It takes more time to calculate than the other two metrics. Note that the roc and the auk are both sea birds.

```
> # calculate X-chromosome metrics on QC'd betas
> seabi(melon.pf, sex=pData(melon.pf)$sex, X=fData(melon.pf)$CHR=='X')

[1] 0.2597014

> # calculate X-chromosome metrics on QC'd and preprocessed betas
> seabi(melon.dasen.pf,
+     sex=pData(melon.dasen.pf)$sex,
+     X=fData(melon.dasen.pf)$CHR=='X'
+ )

[1] 0.1010268
```

5   Suggested analysis workflow

5.1   Load data

You can use a variety of methods to load your data, either from GenomeStudio final report text files or from iDAT files. *methylumi* and *IMA* can read text files, we recommend *methylumi* because the `exprmethy450` object only stores betas and not raw intensities. *methylumi* and *minfi* can both read iDAT files, and produce objects that can be used by our functions. Neither contains the full annotation that comes inside the final report text file. If you use the GenomeStudio file we recommend saving the unnormalized, uncorrected version of the data. We also recommend keeping the barcode names (SentrixID_RnnCnn) as the column headers or in a separate dataframe.

```
> library(methylumi)
> melon <- methyLumiR('finalreport.txt')
```

5.2   Reading IDAT files in *wateRmelon*

Due to the release of the EPIC micro-array and the *methylumi* package not being updated in a long time, we have provided our own IDAT reader that will convert idat files into `MethyLumiSet` objects. The `readEPIC` function is very similar to the *methylumi* idat reader but can handle the EPIC array.

In addition to this, `readEPIC` can recursively search a specified file path for idat files and parse them together.

```
> mlumi <- readEPIC('path/to/directory')
```

5.3 Tidying data

Normalization only works well at cleaning up minor distributional differences between samples. Failed or otherwise atypical samples should be filtered out beforehand. Also if you have different tissues or similar drastic divides in your data it may not be optimal to normalize everything together.

Visualization of raw intensities is a good way of identifying grossly atypical (failed) samples.

```
> boxplot(log(methylated(melon)), las=2, cex.axis=0.8 )
```
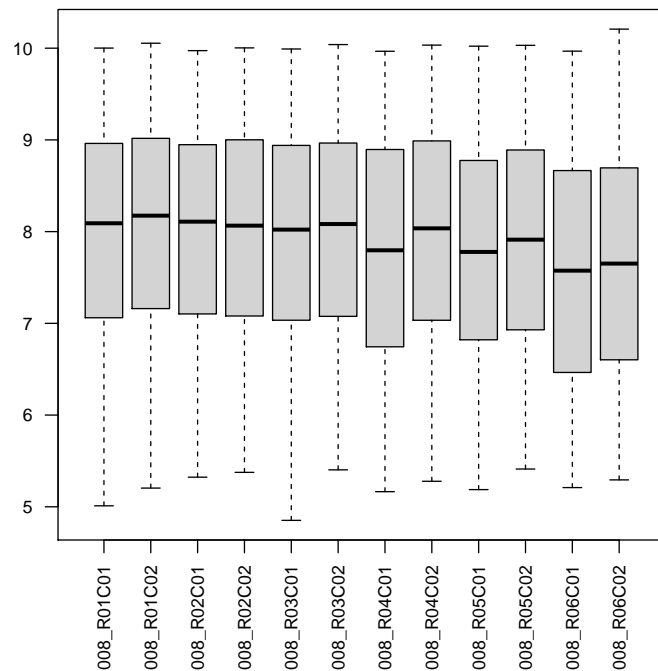


```
> boxplot(log(unmethylated(melon)), las=2, cex.axis=0.8 )
```
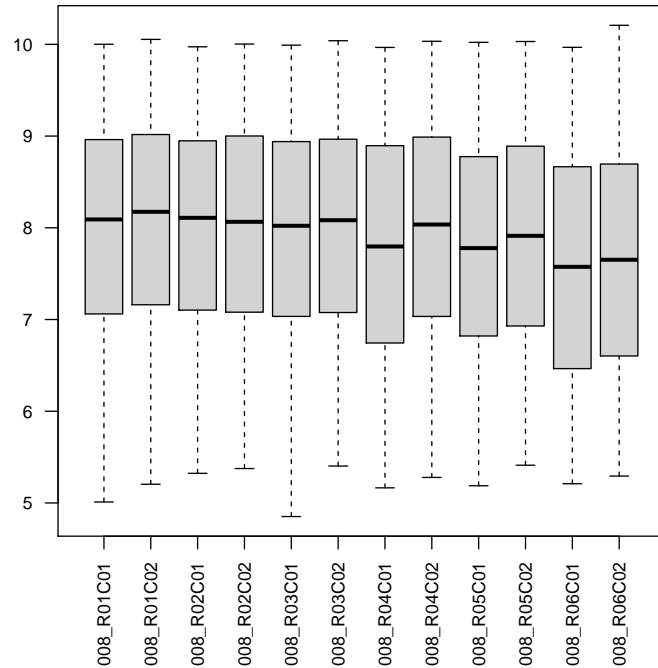
Additionally one can use the `outlyx` and `bscon` functions to easily check data quality. The outlyx function takes any beta matrix (preferably raw) and will identify any samples that are inconsistent with the rest of the data, from the plot we can observe that any data points that fall into the red squares are indeed outlying and should be removed from analysis. If performing this on large data-sets it is possible to take a random subset of probes and run outlyx on that to assess a general idea.

```
> outlyx(melon) # Can take some time on large data-sets

                    iqr     mv outliers
6057825008_R01C01 FALSE FALSE    FALSE
6057825008_R01C02 FALSE FALSE    FALSE
6057825008_R02C01 FALSE  TRUE    FALSE
6057825008_R02C02 FALSE FALSE    FALSE
6057825008_R03C01 FALSE FALSE    FALSE
6057825008_R03C02 FALSE FALSE    FALSE
6057825008_R04C01 FALSE FALSE    FALSE
6057825008_R04C02 FALSE  TRUE    FALSE
```

```
6057825008_R05C01 FALSE FALSE      FALSE
6057825008_R05C02 FALSE FALSE      FALSE
6057825008_R06C01 FALSE FALSE      FALSE
6057825008_R06C02 FALSE FALSE      FALSE
```
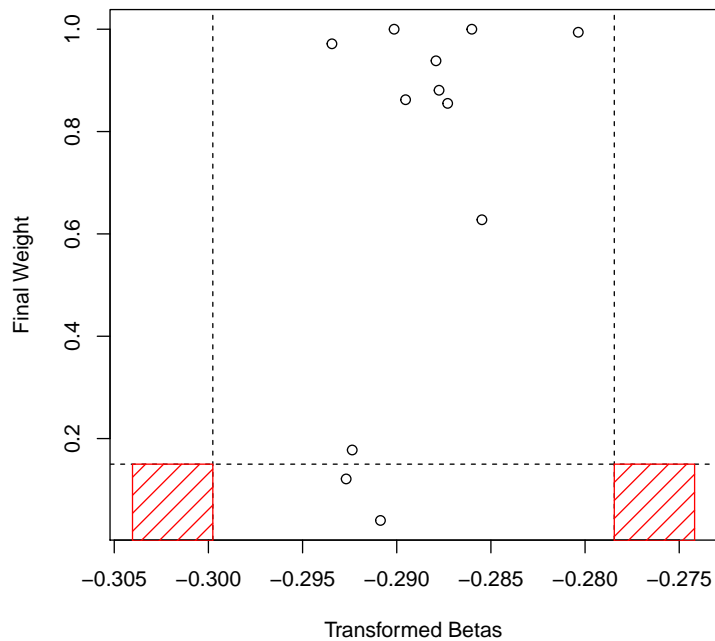

Another way to check data quality is to assess how well the bisulfite conversion had gone when preparing the DNA. There are numerous control probes that assess this and the bscon function converts the intensities from these probes into an easy to understand percentage. For general purposes removing samples with less than 80% bisulfite conversion is usually appropriate for most analysis, but you can be more stringent if you like!

```
> bsc <- bscon(melon)
> hist(bsc)
```

Histogram of bsc
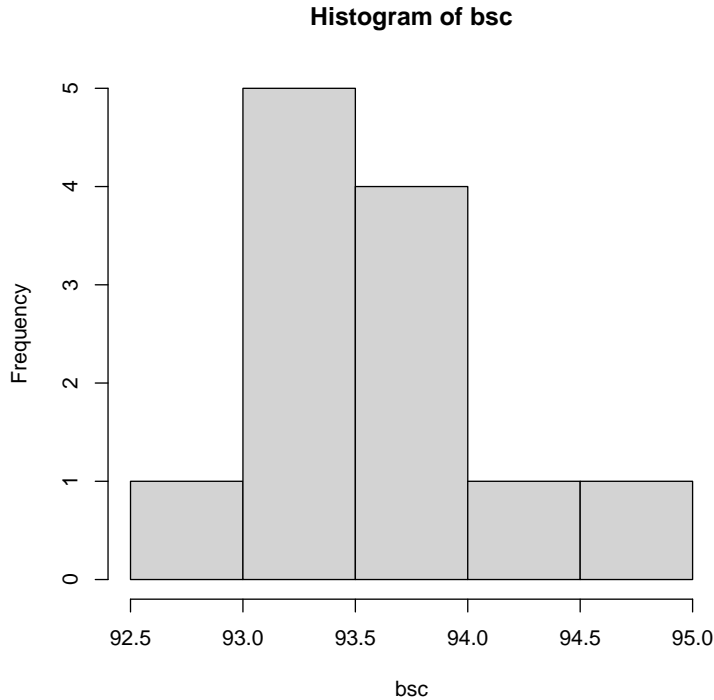
Filtering by detection p-value provides a straightforward approach for removing both failed samples and probes. The `pfilter` function conveniently discards samples with more than (by default) 1% of probes above the .05 detection p-value threshold, and probes with any samples with beadcount under 3 or more than 1% above the p-value threshold.

```
> melon.pf <- pfilter(melon)

0 samples having 1 % of sites with a detection p-value greater than 0.05 were removed
Samples removed:
72 sites were removed as beadcount <3 in 5 % of samples
40 sites having 1 % of samples with a detection p-value greater than 0.05 were remove
```

It has come to our attention that data read in using the various packages and input methods will give subtly variable data output as they calculate detection p-value and beta values differently, and do/don?t give information about beadcount. The pfilter function does not correct for this, but simply uses the detection p-value and bead count provided by each package.

8

5.4 Normalize and calculate betas

In our analysis[1] we tested 15 preprocessing and normalization methods. This involved processing 10 data sets 15 different ways and calculating three metrics of performance from each one. You can do the same thing with your data if you like, but our recommended method `dasen` will work well for most data sets. If you suspect varying dye bias (if you have arrays scanned on different instruments, for example), you might want to try `nanes`.

```
> melon.dasen.pf <- dasen(melon.pf)
```

6 Additional Functionality

We have recently updated the *wateRmelon* package to facilitate a variety of common analyses that are used in EWAS. These include age prediction, cell-type composition estimation and a suite of quality-control tools that we feel are useful for analysis.

6.1 Age Prediction

To perform age prediction within wateRmelon we provide a simple function that will perform the linear function using Horvath's coefficients. These are supplied by default and can be supplied with a different set by manually specifying the `coef` argument.

```
> data(coef)
> agep(melon.dasen.pf, coeff= coef, method='horvath')

                     [,1]
6057825008_R01C01 36.66818
6057825008_R01C02 36.70367
6057825008_R02C01 36.71534
6057825008_R02C02 36.70679
6057825008_R03C01 36.40548
6057825008_R03C02 36.71394
6057825008_R04C01 36.57938
6057825008_R04C02 36.61197
6057825008_R05C01 36.67698
6057825008_R05C02 36.52833
6057825008_R06C01 36.57696
6057825008_R06C02 36.58575
```

6.2 Cell Type Proportion Estimation

Another useful tool for EWAS is estimating cell type proportions, wateR-melon provides a `MethyLumiSet` method for those who prefer using these objects. The two functions are mostly identical, however there is a small distinction between to two.

In wateRmelon we do not normalise the biological data set and the reference data-set together and instead impose the quanitles of the data-set onto the reference data-set then perform the deconvolution step. This approach allows us to one - provide prenormalised data and secondly exploit the nature of normalised data (where each sample has identical quantiles) and impose the normalised quantiles onto the reference dataset. This is beneficial for particularly large data-sets where the sample numbers are in the thousands and the normalising the refernece and biological datasets does not have that much effect on the resultant quantiles.

Currently no EPIC reference datasets exist so any data from Human-MethylationEPIC microarrays will be converted to match the 450K reference dataset.

```
> # Code will not work with melon as melon only has a subset of probes
> estimateCellCounts.wmln(melon.dasen.pf)
```

6.3 Assess Normalisation Violence - `qual`

The `qual` function seeks to explore a facet of analysis that has not been explored in EWAS extensively and that is the potentially source of con-founding introduced by the normalisation method used. The `qual` function seeks to quantify how much a sample has changed during normalisation and will relay it back. The idea behind this is that a sample that has changed drastically from normalisation may have had some nascent problems with it. As a result if you suspect a sample has changed too much during normal-isation the `qual` function will identify this. If you do remove samples using `qual`, make sure to renormalise the data afterwards!

```
> normv <- qual(betas(melon.dasen.pf), betas(melon.pf))
> plot(normv[,1:2])
```
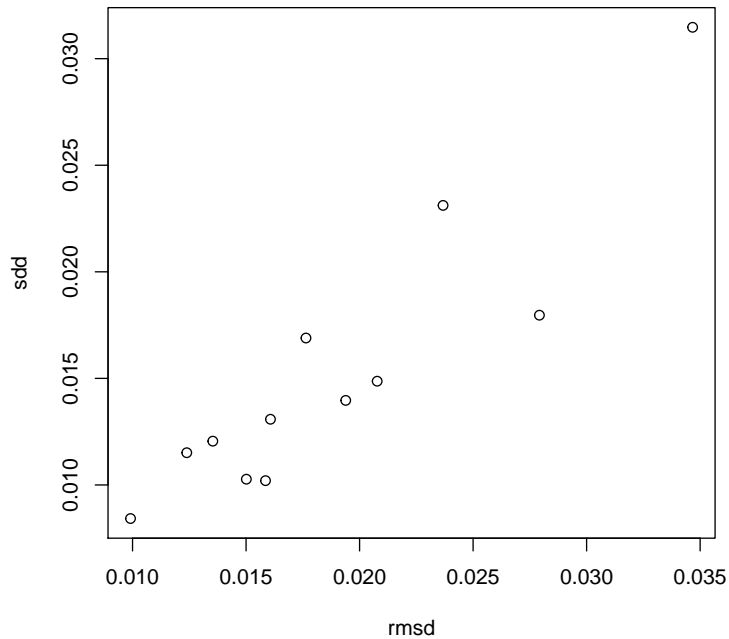
6.4   Remove MAF/SNP heterozygotes before analysis

The last function we will discuss is pwod (probe wise outlier detection). This tool rather simply scans a betas matrix and removes any highly outlying signals (4 IQRs away). These signals are likely caused my MAFs and or SNP heterozygotes within your sample population and may be in probes that are not in the SNP lists/cross reactive probe lists that you may want to remove from the data prior to statistical testing.

```
> melon.pwod.dasen.pf <- pwod(betas(melon.dasen.pf))
```

```
72 probes detected.
```

6.5   Example workflow: method for MethyLumiSet

```
> data(melon)
> # load in melon dataset
> # Optional read-in data using:
```

```
> # melon <- readEPIC('path/to/idats')
>
> outliers <- outlyx(melon, plot=F)
> sum(outliers$out)

[1] 0

> sum(bscon(melon)<85)

[1] 0

> # Check for outliers
>
> melon.pf<-pfilter(melon)

0 samples having 1 % of sites with a detection p-value greater than 0.05 were removed
Samples removed:
72 sites were removed as beadcount <3 in 5 % of samples
40 sites having 1 % of samples with a detection p-value greater than 0.05 were remove

> # perform QC on raw data matrix using default thresholds
> melon.dasen.pf<-dasen(melon.pf)
> # preprocess using our best method
>
> qual <- qual(betas(melon.dasen.pf), betas(melon.pf))
> # Check for bad samples (again)
>
> sex  <- pData(melon.dasen.pf)$sex
> # extract phenotypic information for test
> bet<-betas(melon.dasen.pf)
> # extract processed beta values
> melon.sextest<-sextest(bet,sex)
> # run t-test to idenitify sex difference
>
> agep(melon.dasen.pf)

                      [,1]
6057825008_R01C01 36.66818
6057825008_R01C02 36.70367
6057825008_R02C01 36.71534
6057825008_R02C02 36.70679
```

```
6057825008_R03C01 36.40548
6057825008_R03C02 36.71394
6057825008_R04C01 36.57938
6057825008_R04C02 36.61197
6057825008_R05C01 36.67698
6057825008_R05C02 36.52833
6057825008_R06C01 36.57696
6057825008_R06C02 36.58575


> # Check ages (see if they match up e.t.c)
>
> melon.pwod <- pwod(bet)

72 probes detected.

> # Clean up data for statistical testing
```

6.6 Further analysis

We can't offer much help with the actual analysis, which will be different for every experiment. In general though, you need to get your experimental variables into the same order as your arrays and apply some kind of statistical test to each row of the table of betas. In the workflow shown here, the array barcodes are preserved and can be retrieved with `sampleNames` or `colnames`. It's convenient to read in a samplesheet, for example in csv format, with the barcodes as row names.

7 References

[1] Pidsley R, Wong CCY, Volta M, Lunnon K, Mill J, Schalkwyk LC: A data-driven approach to preprocessing Illumina 450K methylation array data (submitted)

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

[3] Touleimat N, Tost J: Complete pipeline for Infinium R Human Methylation 450K BeadChip data processing using subset quantile normalization for accurate DNA methylation estimation. Epigenomics 2012, 4:325-341)

[4]Teschendorff AE, Marabita F, Lechner M, Bartlett T, Tegner J, Gomez-Cabrero D, Beck S. A Beta-Mixture Quantile Normalisation method for cor-

recting probe design bias in Illumina Infinium 450k DNA methylation data. Bioinformatics. 2012 Nov 21.

[5] Maksimovic J, Gordon L, Oshlack A: SWAN: Subset quantile Within-Array Normalization for Illumina Infinium HumanMethylation450 Bead-Chips. Genome biology 2012, 13(6):R44