MLML2R package User's Guide

Samara F. Kiihl and Maria Tellez-Plaza

2018-02-21

Abstract

We present a guide to the *R* package *MLML2R*. The package provides computational efficient maximum likelihood estimates of DNA methylation and hydroxymethylation proportions when data from the DNA processing methods bisulfite conversion (BS), oxidative bisulfite conversion (ox-BS), and Tet-assisted bisulfite conversion (TAB) are available. Estimates can be obtained when data from all the three methods are available or when any combination of only two of them are available. The package does not depend on other *R* packages, allowing the user to read and preprocess the data with any given software, to import the results into *R* in matrix format, to obtain the maximum likelihood 5-hmC and 5-mC estimates and use them as input for other packages traditionally used in genomic data analysis, such as *minfi*, *sva* and *limma*.

Package version: MLML2R

Contents

1 Introduction

In a given CpG site from a single cell we will either have a C or a T after DNA processing conversion methods, with a different interpretation for each of the available methods. This is a binary outcome and we assume a Binomial model and use the maximum likelihood estimation method to obtain the estimates for hydroxymethylation and methylation proportions.

T reads are referred to as converted cytosine and C reads are referred to as unconverted cytosine. Conventionally, T counts are also referred to as unmethylated counts, and C counts as methylated counts. In case of Infinium Methylation arrays, we have intensities representing the methylated (M) and unmethylated (U) channels that are proportional to the number of unconverted and converted cytosines (C and T, respectively). The most used summary from these experiments is the proportion $\beta = \frac{M}{M+U}$, commonly referred to as *beta-value*, which reflects the methylation level at a CpG site. Naïvely using the difference between betas from BS and oxBS as an estimate of 5-hmC (hydroxymethylated cytosine), and the difference between betas from BS and TAB as an estimate of 5-mC (methylated cytosine) can many times provide negative proportions and instances where the sum of 5-C (unmodified cytosine), 5-mC and 5-hmC proportions is greater than one due to measurement errors.

MLML2R package allows the user to jointly estimate hydroxymethylation and methylation consistently and efficiently.

The function `MLML` takes as input the data from the different methods and returns the estimated proportion of methylation, hydroxymethylation and unmethylation for a given CpG site. Table 1 presents the arguments of the `MLML` and Table 2 lists the results returned by the function.

The function assumes that the order of the rows and columns in the input matrices are consistent. In addition, all the input matrices must have the same dimension. Usually, rows represent CpG loci and columns are the samples.

Table 1: MLML function and random variable notation.

Arguments	Description
G.matrix	Unmethylated channel (Converted cytosines/ T counts) from TAB-conversion (reflecting 5-C + 5-mC).
H.matrix	Methylated channel (Unconverted cytosines/ C counts) from TAB-conversion (reflecting True 5-hmC).
L.matrix	Unmethylated channel (Converted cytosines/ T counts) from oxBS-conversion (reflecting 5-C + 5-hmC).
M.matrix	Methylated channel (Unconverted cytosines/ C counts) from oxBS-conversion (reflecting True 5-mC).
T.matrix	Methylated channel (Unconverted cytosines/ C counts) from standard BS-conversion (reflecting 5-mC+5-hmC).
U.matrix	Unmethylated channel (Converted cytosines/ T counts) from standard BS-conversion (reflecting True 5-C).

Table 2: Results returned from the MLML function

Value	Description
mC	maximum likelihood estimate for the 5-mC proportion
hmC	maximum likelihood estimate for the 5-hmC proportion
C	maximum likelihood estimate for the 5-mC proportion
methods	the conversion methods used to produce the MLE

2 Worked examples

2.1 Publicly available data: GSE63179

We will use the dataset from Field (2015), which consists of eight DNA samples from the same DNA source treated with oxBS-BS and hybridized to the Infinium 450K array.

When data is obtained through Infinium Methylation arrays, we recommend the use of the *minfi* package (Aryee et al. 2014), a well-established tool for reading, preprocessing and analysing DNA methylation data from these platforms. Although our example relies on *minfi* and other *Bioconductor* tools, *MLML2R* does not depend on any packages. Thus, the user is free to read and preprocess the data using any software of preference and then import the intensities (or T and C counts) for the methylated and unmethylated channel (or converted and uncoverted cytosines) into *R* in matrix format.

To start this example we will need the following packages:

```
library(MLML2R)
library(minfi)
library(GEOquery)
```

It is usually best practice to start the analysis from the raw data, which in the case of the 450K array is a .IDAT file.

The raw files are deposited in GEO and can be downloaded by using the getGEOSuppFiles. There are two files for each replicate, since the 450k array is a two-color array. The .IDAT files are downloaded in compressed format and need to be uncompressed before they are read by the read.metharray.exp function.

```
getGEOSuppFiles("GSE63179")
untar("GSE63179/GSE63179_RAW.tar", exdir = "GSE63179/idat")
```

```
list.files("GSE63179/idat", pattern = "idat")
files <- list.files("GSE63179/idat", pattern = "idat.gz$", full = TRUE)
sapply(files, gunzip, overwrite = TRUE)
```

The .IDAT files can now be read:

```
rgSet <- read.metharray.exp("GSE63179/idat")
```

To access phenotype data we use the pData function. The phenotype data is not yet available from the rgSet.

```
pData(rgSet)
```

In this example the phenotype is not really relevant, since we have only one sample: male, 25 years old. What we do need is the information about the conversion method used in each replicate: BS or oxBS. We will access this information automatically from GEO:

```
geoMat <- getGEO("GSE63179")
pD.all <- pData(geoMat[[1]])
pD <- pD.all[, c("title", "geo_accession", "characteristics_ch1.1",
                 "characteristics_ch1.2","characteristics_ch1.3")]
pD
```

This phenotype data needs to be merged into the methylation data. The following commands guarantee we have the same replicate identifier in both datasets before merging.

```
sampleNames(rgSet) <- sapply(sampleNames(rgSet),function(x)
  strsplit(x,"_")[[1]][1])
rownames(pD) <- pD$geo_accession
pD <- pD[sampleNames(rgSet),]
pData(rgSet) <- as(pD,"DataFrame")
rgSet
```

The rgSet object is a class called *RGChannelSet* used for two color data (green and a red channel). The input in the MLML funcion is *MethylSet*, which contains the methylated and unmethylated signals. The most basic way to construct a *MethylSet* is using the function preprocessRaw. Here we chose the function preprocessNoob (Triche et al. 2013) for background correction and construction of the *MethylSet*.

```
MSet.noob<- preprocessNoob(rgSet)
```

After the preprocessed steps we can use MLML from the *MLML2R* package.

The BS replicates are in columns 1, 3, 5, and 6 (information from pD$title). The remaining columns are from the oxBS treated replicates.

```
MethylatedBS <- getMeth(MSet.noob)[,c(1,3,5,6)]
UnMethylatedBS <- getUnmeth(MSet.noob)[,c(1,3,5,6)]
MethylatedOxBS <- getMeth(MSet.noob)[,c(7,8,2,4)]
UnMethylatedOxBS <- getUnmeth(MSet.noob)[,c(7,8,2,4)]
```

When only two methods are available, the default option of MLML function returns the exact constrained maximum likelihood estimates using the the pool-adjacent-violators algorithm (PAVA) (Ayer et al. 1955).

```
results_exact <- MLML(T.matrix = MethylatedBS , U.matrix = UnMethylatedBS,
                      L.matrix = UnMethylatedOxBS, M.matrix = MethylatedOxBS)
```

Maximum likelihood estimate via EM-algorithm approach (Qu et al. 2013) is obtained with the option iterative=TRUE. In this case, the default (or user specified) tol is considered in the iterative method.

```
results_em <- MLML(T.matrix = MethylatedBS , U.matrix = UnMethylatedBS,
                   L.matrix = UnMethylatedOxBS, M.matrix = MethylatedOxBS,
```

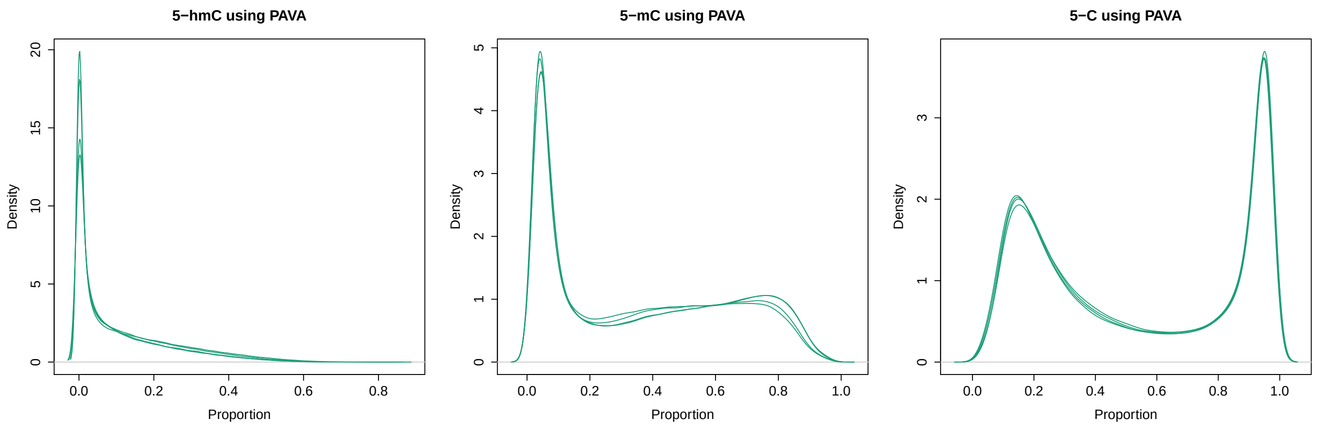

Figure 1: Estimated proportions of hydroxymethylation, methylation and unmethylation for the CpGs in the dataset using the MLML function with default options.

```
                            iterative = TRUE)
```

The estimates are very similar for both methods:

```
all.equal(results_exact$hmC,results_em$hmC,scale=1)
```

2.2 Simulated data

To illustrate the package when all the three methods are available or when any combination of only two of them are available, we will simulate a dataset.

We will use a sample of the estimates of 5-mC, 5-hmC and 5-C of the previous example as the true proportions, as shown in Figure 2.

Two replicate samples with 1000 CpGs will be simulated. For CpG i in sample j:

$$T_{i,j} \sim Binomial(n = c_{i,j}, p = p_m + p_h)$$
$$M_{i,j} \sim Binomial(n = c_{i,j}, p = p_m)$$
$$H_{i,j} \sim Binomial(n = c_{i,j}, p = p_h)$$
$$U_{i,j} = c_{i,j} - T_{i,j}$$
$$L_{i,j} = c_{i,j} - M_{i,j}$$
$$G_{i,j} = c_{i,j} - H_{i,j}$$

where the random variables are defined in Table 1, and $c_{i,j}$ represents the coverage for CpG i in sample j.

The following code produce the simulated data:

```
set.seed(112017)

index <- sample(1:dim(results_exact$mC)[1],1000,replace=FALSE) # 1000 CpGs

Coverage <- round(MethylatedBS+UnMethylatedBS)[index,1:2] # considering 2 samples

temp1 <- data.frame(n=as.vector(Coverage),
```

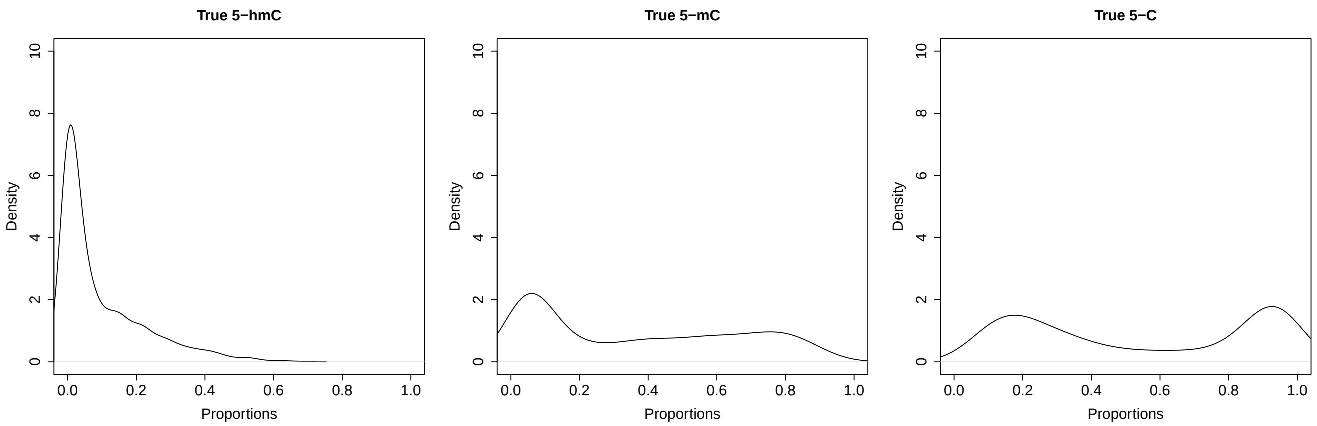

Figure 2: True proportions of hydroxymethylation, methylation and unmethylation for the CpGs used to generate the datasets.

```
                    p_m=c(results_exact$mC[index,1],results_exact$mC[index,1]),
                    p_h=c(results_exact$hmC[index,1],results_exact$hmC[index,1]))

MethylatedBS_temp <- c()
for (i in 1:dim(temp1)[1])
{
  MethylatedBS_temp[i] <- rbinom(n=1, size=temp1$n[i], prob=(temp1$p_m[i]+temp1$p_h[i]))
}

UnMethylatedBS_sim2 <- matrix(Coverage - MethylatedBS_temp,ncol=2)
MethylatedBS_sim2 <- matrix(MethylatedBS_temp,ncol=2)


MethylatedOxBS_temp <- c()
for (i in 1:dim(temp1)[1])
{
  MethylatedOxBS_temp[i] <- rbinom(n=1, size=temp1$n[i], prob=temp1$p_m[i])
}

UnMethylatedOxBS_sim2 <- matrix(Coverage - MethylatedOxBS_temp,ncol=2)
MethylatedOxBS_sim2 <- matrix(MethylatedOxBS_temp,ncol=2)


MethylatedTAB_temp <- c()
for (i in 1:dim(temp1)[1])
{
  MethylatedTAB_temp[i] <- rbinom(n=1, size=temp1$n[i], prob=temp1$p_h[i])
}


UnMethylatedTAB_sim2 <- matrix(Coverage - MethylatedTAB_temp,ncol=2)
MethylatedTAB_sim2 <- matrix(MethylatedTAB_temp,ncol=2)

true_parameters_sim2 <- data.frame(p_m=results_exact$mC[index,1],p_h=results_exact$hmC[index,1])
true_parameters_sim2$p_u <- 1-true_parameters_sim2$p_m-true_parameters_sim2$p_h
```

2.2.1 BS and oxBS methods

When only two methods are available, the default option returns the exact constrained maximum likelihood estimates using the the pool-adjacent-violators algorithm (PAVA) (Ayer et al. 1955).

```
library(MLML2R)
 results_exactBO1 <- MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2)
```

Maximum likelihood estimate via EM-algorithm approach (Qu et al. 2013) is obtained with the option `iterative=TRUE`. In this case, the default (or user specified) `tol` is considered in the iterative method.

```
 results_emBO1 <- MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,iterative=TRUE)
```

When only two methods are available, we highly recommend the default option `iterative=FALSE` since the difference in the estimates obtained via EM and exact constrained is very small, but the former requires more computational effort:

```
 all.equal(results_emBO1$hmC,results_exactBO1$hmC,scale=1)
## [1] "Mean absolute difference: 9.581949e-05"
```

```
 library(microbenchmark)
 mbmBO1 = microbenchmark(
    EXACT = MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
                 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2),
    EM =    MLML(T.matrix = MethylatedBS_sim2, U.matrix = UnMethylatedBS_sim2,
                 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
                 iterative=TRUE),
    times=10)
 mbmBO1
## Unit: microseconds
##   expr       min        lq       mean     median       uq       max neval
##  EXACT   364.081   369.246   685.0832   413.0855   419.94  1963.115    10
##     EM 13386.102 15000.356 19854.0737 16206.4055 17478.70 54161.686    10
```

Comparison between approximate exact constrained and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_exactBO1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.01165593"
```

Comparison between EM-algorithm and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_emBO1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.01011952"
```

2.2.2 BS and TAB methods

Using PAVA:

```
results_exactBT1 <- MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2)
```

Using EM-algorithm:

```
 results_emBT1 <- MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
 G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2,iterative=TRUE)
```

Comparison between PAVA and EM:

```
all.equal(results_emBT1$hmC,results_exactBT1$hmC,scale=1)
## [1] "Mean absolute difference: 7.675267e-07"
```

```
mbmBT1 = microbenchmark(
   EXACT = MLML(T.matrix = MethylatedBS_sim2, U.matrix = UnMethylatedBS_sim2,
                G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2),
   EM =    MLML(T.matrix = MethylatedBS_sim2, U.matrix = UnMethylatedBS_sim2,
                G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2,
                iterative=TRUE),
   times=10)
mbmBT1
## Unit: microseconds
##   expr       min        lq       mean     median        uq       max neval
##  EXACT   335.845   373.036   491.6498   386.8215   415.581  1188.552    10
##     EM 14376.247 14581.158 15420.6476 15216.6280 16009.418 17211.354    10
```

Comparison between approximate exact constrained and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_exactBT1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.00644861"
```

Comparison between EM-algorithm and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_emBT1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.004719911"
```

2.2.3 oxBS and TAB methods

Using PAVA:

```
results_exactOT1 <- MLML(L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2)
```

Using EM-algorithm:

```
results_emOT1 <- MLML(L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2,iterative=TRUE)
```

Comparison between PAVA and EM:

```
all.equal(results_emOT1$hmC,results_exactOT1$hmC,scale=1)
## [1] "Mean absolute difference: 2.019638e-07"
```

```
mbmOT1 = microbenchmark(
   EXACT = MLML(L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
                G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2),
   EM =    MLML(L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
                G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2,
                iterative=TRUE),
   times=10)
mbmOT1
## Unit: microseconds
##   expr      min       lq      mean    median       uq      max neval
##  EXACT  281.385  287.219  388.7614  297.3005  313.585 1212.235    10
##     EM 5407.036 6013.146 6005.1827 6059.8590 6076.385 6209.908    10
```

Comparison between approximate exact constrained and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_exactOT1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.006451817"
```

Comparison between EM-algorithm and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_emOT1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.00645154"
```

2.2.4 BS, oxBS and TAB methods

When data from the three methods are available, the default otion in the MLML function returns the constrained maximum likelihood estimates using an approximated solution for Lagrange multipliers method.

```
results_exactBOT1 <- MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2)
```

Maximum likelihood estimate via EM-algorithm approach (Qu et al. 2013) is obtained with the option iterative=TRUE. In this case, the default (or user specified) tol is considered in the iterative method.

```
 results_emBOT1 <- MLML(T.matrix = MethylatedBS_sim2 , U.matrix = UnMethylatedBS_sim2,
 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
 G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2,iterative=TRUE)
```

We recommend the default option iterative=FALSE since the difference in the estimates obtained via EM and the approximate exact constrained is very small, but the former requires more computational effort:

```
 all.equal(results_emBOT1$hmC,results_exactBOT1$hmC,scale=1)
## [1] "Mean absolute difference: 1.627884e-06"
```

```
 mbmBOT1 = microbenchmark(
    EXACT = MLML(T.matrix = MethylatedBS_sim2, U.matrix = UnMethylatedBS_sim2,
                 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
                 G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2),
    EM =    MLML(T.matrix = MethylatedBS_sim2, U.matrix = UnMethylatedBS_sim2,
                 L.matrix = UnMethylatedOxBS_sim2, M.matrix = MethylatedOxBS_sim2,
                 G.matrix = UnMethylatedTAB_sim2, H.matrix = MethylatedTAB_sim2,
                 iterative=TRUE),
    times=10)
 mbmBOT1
## Unit: microseconds
##   expr      min       lq     mean   median       uq      max neval
##  EXACT  861.023  877.852 1110.329  916.691 1045.546  1952.98    10
##     EM 1965.518 2730.089 7056.278 2924.055 3657.908 44019.26    10
```

Comparison between approximate exact constrained and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_exactBOT1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.005664222"
```

Comparison between EM-algorithm and true hydroxymethylation proportion used in simulation:

```
all.equal(true_parameters_sim2$p_h,results_emBOT1$hmC[,1],scale=1)
## [1] "Mean absolute difference: 0.004146021"
```

References

Aryee, Martin J., Andrew E. Jaffe, Hector Corrada-Bravo, Christine Ladd-Acosta, Andrew P. Feinberg, Kasper D. Hansen, and Rafael A. Irizarry. 2014. "Minfi: A flexible and comprehensive Bioconductor package for the analysis of Infinium DNA Methylation microarrays." *Bioinformatics* 30 (10):1363–9. https://doi.org/10.1093/bioinformatics/btu049.

Ayer, Miriam, H. D. Brunk, G. M. Ewing, W. T. Reid, and Edward Silverman. 1955. "An Empirical Distribution Function for Sampling with Incomplete Information." *Ann. Math. Statist.* 26 (4). The Institute of Mathematical Statistics:641–47. https://doi.org/10.1214/aoms/1177728423.

Field, Dario AND Bachman, Sarah F. AND Beraldi. 2015. "Accurate Measurement of 5-Methylcytosine and 5-Hydroxymethylcytosine in Human Cerebellum Dna by Oxidative Bisulfite on an Array (Oxbs-Array)." *PLOS ONE* 10 (2). Public Library of Science:1–12. https://doi.org/10.1371/journal.pone.0118202.

Qu, Jianghan, Meng Zhou, Qiang Song, Elizabeth E. Hong, and Andrew D. Smith. 2013. "MLML: Consistent Simultaneous Estimates of Dna Methylation and Hydroxymethylation." *Bioinformatics* 29 (20):2645–6. https://doi.org/10.1093/bioinformatics/btt459.

Triche, Timothy J., Daniel J. Weisenberger, David Van Den Berg, Peter W. Laird, and Kimberly D. Siegmund. 2013. "Low-Level Processing of Illumina Infinium DNA Methylation BeadArrays." *Nucleic Acids Research* 41 (7):e90. https://doi.org/10.1093/nar/gkt090.