The FEM R package: Identification of Functional Epigenetic Modules

Yinming Jiao and Andrew E. Teschendorff

October 17, 2016

1 Summary

This vignette provides examples of how to use the package FEM to identify interactome hotspots of differential promoter methylation and differential expression, where an inverse association between promoter methylation and gene expression is assumed [**?**]. By "interactome hotspot" we mean a connected subnetwork of the protein interaction network (PIN) with an exceptionally large average edge-weight density in relation to the rest of the network. The weight edges are constructed from the statistics of association of DNA methylation and gene expression with the phenotype of interest. Thus, the FEM algorithm can be viewed as a functional supervised algorithm, which uses a network of relations between genes (in our case a PPI network), to identify subnetworks where a significant number of genes are associated with a phenotype of interest (POI). We call these "hotspots" also Functional Epigenetic Modules (FEMs). Current functionality of FEM works for Illumina Infinium 450k data, however, the structure is modular allowing easy application or generalization to DNA methylation data generated with other technologies. The FEM algorithm on Illumina 27k data was first presented in [**?**], with its extension to Illumina 450k data described in [**?**]. The module detection algorithm used is the spin-glass algorithm of [**?**]. The PIN used in this vignette includes only protein-protein interactions, derives from Pathway Commons [**?**] and is available from *http://sourceforge.net/projects/funepimod/files* under filename *hprdAsigH*.Rd*, but the user is allowed to specify his own network.
There are three main components to this vignette. These are:

- Application to simulated data.
- Real world example: application to Endometrial Cancer.
- Further details of the algorithm.

2 Application to simulated data

Since FEM is a BioC package, the user first needs to install both R and Bioconductor. R is a free open source software project freely downloadable from the CRAN website *http://cran.r-project.org/*. FEM has several package dependencies, such as igraph, marray, corrplot and graph. So these need to be installed

first, if they have not been installed already. For example. to install igraph and corrplot,etc. Type *install.packages(c("igraph","corrplot"))*. Since marray is a bioconductor package, we can install it with
source("http://bioconductor.org/biocLite.R");
biocLite("marray").
Load them with

```
> library("igraph");
> library("marray");
> library("corrplot");
> library("graph");
```

You can check your version of iGraph by entering *sessionInfo()$otherPkgs$igraph$Version.* It should be at least version 0.6.
To load the FEM package in your R session,

```
> library("FEM");
```

We demonstrate the functionality of FEM using first a simulated toy dataset. Briefly, the toy dataset consists of a small random graph gene network, with a clique embedded in it. To load it into the session, use

```
> data(Toydata)
```

The object *Toydata* is a list and has four elements:

```
> names(Toydata)
```

```
[1] "statM"     "statR"     "adjacency" "tennodes"
```

The first element is *statM*

```
> head(Toydata$statM)
```

```
            t-value      p-value
117144 -0.065111202 4.740427e-01
56898   0.001161095 4.995368e-01
8916    0.195829329 4.223719e-01
54476   3.934019816 4.176845e-05
7368   -0.025401308 4.898674e-01
79058  -0.089637422 4.642877e-01
```

statM is a matrix of differential DNA methylation t-statistics and P-values (one row for each gene promoter) with rownames annotated to arbitrary entrez gene IDs. There are 84 rows because the maximally connected component of the original 100-node random graph was of size 84. Most of the rows have statistics which have been simulated to be close to zero (between -0.5 and 0.5), meaning that for these there is no association between DNA methylation and the phenotype of interest (POI). There are also ten randomly selected probes/genes whose t-statistics are randomly chosen to lie between 2 and 5, meaning that for these promoters high methylation is associated positively with the POI.
Now lets's check which 10 genes have t-statistics larger than 2:

```
> rownames(Toydata$statM)[which(Toydata$statM[,1]>2)]->tennodes;
> tennodes;
```

```
 [1] "54476"  "166929" "4701"   "4299"   "57696"  "9865"   "9632"   "1465"
 [9] "9985"   "64208"
```

This should agree with the tennodes entry of *Toydata*. The second member is *statR*

```
> head(Toydata$statR)
```

```
           t-value     p-value
117144 -0.09591858 0.461792617
56898  -0.03296763 0.486850201
8916    0.17791827 0.429393580
54476  -2.57586127 0.004999538
7368   -0.12861502 0.448831142
79058  -0.03371232 0.486553276
```

statR is a matrix of differential expression t-statistics and P-values (same dimension as *statM.m* and ordered in same way) with rownames annotated with the same Entrez gene ID. The t-statistics of the previously selected 10 genes are set to lie between -2 and -5. Thus, the toydata object models the case of ten promoters which are underexpressed due to hypermethylation. Indeed, the following command identifies the genes that are significantly underexpressed. They agree with those that are hypermethylated:

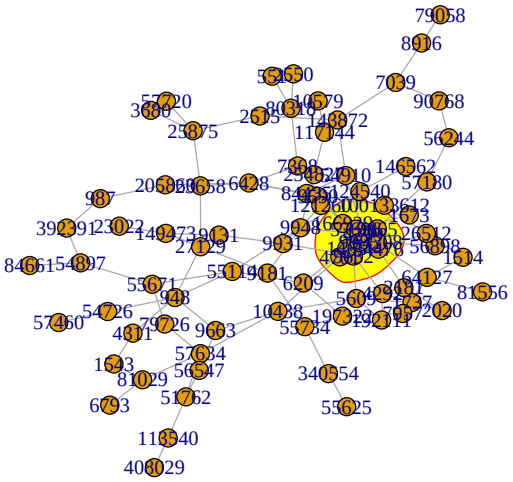
```
> rownames(Toydata$statM)[which(Toydata$statR[,1]< -2)];
```

```
 [1] "54476"  "166929" "4701"   "4299"   "57696"  "9865"   "9632"   "1465"
 [9] "9985"   "64208"
```

The third member is *adjacency*. This is the adjacency matrix, with number of rows and columns equal to the number of rows of statR (or statM), ordered in same way and with same gene identifier. The graph is connected, and was constructed as the maximally connected subgraph of a 100-node random graph (Erdos-Renyi graph). Specifically, the original random graph was generated with *igraph::erdos.renyi.game(100, 2/100)* and then the solitary nodes were removed, resulting in a maximally connected subnetwork of 84 nodes/genes.
We note that the 10 nodes which are differentially methylated and differentially expressed, form a clique, meaning that each of these ten nodes is connected to each other. So they belong to a deliberately created module with high absolute differential methylation and differential expression t-statistics. They are indicated in the following plot.

Plot this network from *adjacency* with the ten nodes marked as a group.

```
> mod.idx <- match(Toydata$tennodes,rownames(Toydata$adj));
> plot.igraph(graph.adjacency(Toydata$adjacency,mod="undirected"),
+ vertex.size=8,mark.groups=mod.idx,mark.col="yellow")
```



As mentioned before, FEM is used to detect interactome hotspots of differential promoter methylation and differential expression, where an inverse association between promoter methylation and gene expression is assumed. So let us test whether FEM can detect the simulated module of ten nodes.

We use DoFEMbi() to find the community structures induced by these phenotype changes in the Toydata. First, however, we need to define the input object, as follows:

```
> intFEM.o <- list(statM=Toydata$statM,statR=Toydata$statR,adj=Toydata$adj);

> DoFEMtoy.o  <- DoFEMbi(intFEM.o,nseeds=1,gamma=0.5,nMC=1000,
+ sizeR.v=c(1,100),minsizeOUT=10,writeOUT=TRUE,nameSTUDY="TOY",ew.v=NULL);
```

Some of the arguments of this function are worth describing here:

nseeds: This is the number of seeds/modules to search for.

gamma: This is the tuning parameter of the spin-glass module detection algorithm. This parameter is very important because it controls the average module size. Default value generally leads to modules in the desired size range of 10 to 100 genes.

nMC: The number of Monte Carlo runs for establishing statistical significance of modularity values under randomisation of the molecular profiles on the network.

You can also use "sizeR.v" to set the desired size range for modules, "minsizeOUT" to set the minimum size of modules to report as interesting and "writeOUT" to indicate whether to write out tables in text format.

There are two output files summarising the results of the *DoFEMbi* function. One output file contains colums which describe the following:

- size: a vector of inferred module sizes for each of the ntop seeds.
- mod: a vector of associated modularities.
- pv: a vector of associated significance P-values (with resolution of nMC runs).
- selmod: index positions of significant modules of size at least minsizeOUT and smaller than the maximum specified in sizeR.v
- fem: a summary matrix of the selected modules.
- topmod: a list of summary matrices for each of the selected module

Let's check the details of the output:

```
> DoFEMtoy.o$fem
```

```
     EntrezID(Seed) Symbol(Seed) Size Mod                P
[1,] "166929"       "SGMS2"      "36" "4.17008628882728" "0"
     Genes
[1,] "SGMS2 RNF216 UGT8 NDUFA7 AFF1 DDX55 TRIL AKAP12 SEC24C CSRP1 REC8 POPDC3 BDH2 ACSL3
```

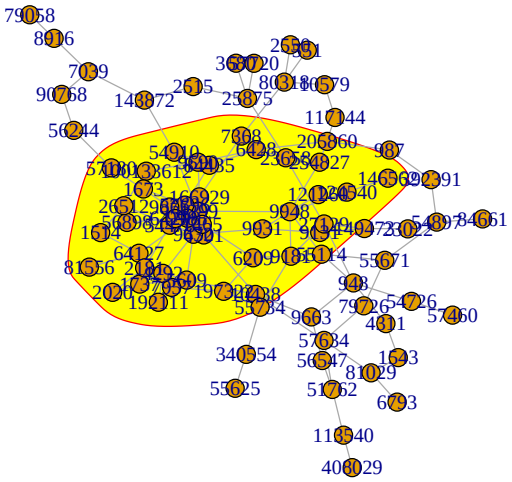
We can see that we get one module with seed gene "SGMS2" (entrez gene ID = 166929), hence module name "SGMS2". This module has 36 members. With the following command, we can have a look at the details of the first five genes of the "SGMS2" module:

```
> head(DoFEMtoy.o$topmod$SGMS2,n=5L)
```

```
       EntrezID Symbol   stat(DNAm)            P(DNAm)
166929 "166929" "SGMS2"  "3.98617629287764"    "3.35733147288641e-05"
54476  "54476"  "RNF216" "3.93401981610805"    "4.17684458448805e-05"
7368   "7368"   "UGT8"   "-0.0254013077355921" "0.489867434011738"
4701   "4701"   "NDUFA7" "3.49410701170564"    "0.00023782515959272"
4299   "4299"   "AFF1"   "2.18818748369813"    "0.0143279742210821"
       stat(mRNA)           P(mRNA)                 stat(Int)
166929 "-3.05292705958709"  "0.00113310507560836"  "7.29480934183558"
54476  "-2.57586126867682"  "0.00499953781149238"  "6.72562916567027"
7368   "-0.128615015652031" "0.44883114211167"     "0"
4701   "-3.28854257101193"  "0.00050353776488235"  "7.05809017310707"
4299   "-3.31781403906643"  "0.000453624323520969" "5.78389382246553"
```

To show the SGMS2 module in the whole network, use:

```
> mod.idx<- match(DoFEMtoy.o$topmod$SGMS2[,1],rownames(Toydata$adj));
> plot.igraph(graph.adjacency(Toydata$adjacency,mod="undirected"),
+ vertex.size=8,mark.groups=mod.idx,mark.col="yellow")
```



In order to check the effectiveness of FEM in detecting the true module, we compute the sensitivity, defined as the fraction of the truly associated genes that are captured by the inferred module:

```
> sensitivity=length(intersect(tennodes,rownames(Toydata$adj)[mod.idx]))/length(tennodes);
> sensitivity

[1] 1
```

Thus, we find that the sensitivity is 100%, meaning that FEM inferred a module that contained all 10 genes that were truly differentially methylated and expressed. Although, specificity is not 100%, this is to be expected since a larger module can still exhibit a higher than random average weight density.

3 A real world example: application to Endometrial Cancer

To validate the FEM algorithm on real Illumina 450k data we collected and analyzed 118 endometrial cancers and 17 normal endometrial samples, all with matched RNA-Seq data from the TCGA study [**?**]. To assign DNA methylation values to a given gene, in the case of Illumina 27k data, we assigned the probe value closest to the transcription start site (TSS). In the case of Illumina 450k data, we assigned to a gene, the average value of probes mapping to within 200bp of the TSS. If no probes map to within 200bp of the TSS, we use the average of probes mapping to the 1st exon of the gene. If such probes are also not present, we use the average of probes mapping to within 1500bp of the TSS. Justification

for this procedure is provided in our Bioinformatics paper [**?**]. For each gene g in the maximally connected subnetwork, we then derive a statistic of association between its DNA methylation profile and the POI (here normal/cancer status), denoted by $t_g^{(D)}$ as well as between its mRNA expression profile and the same POI, which we denote by $t_g^{(R)}$. These statistics have already been computed beforehand using the limma package. We now load them in:

```
> data(Realdata);
> attributes(Realdata);

$names
[1] "statM"     "statR"     "adjacency" "fembi.o"
```

As before, we prepare the input object:

```
> intFEM.o <- list(statM=Realdata$statM,statR=Realdata$statR,adj=Realdata$adj)
```

Since running DoFEMbi on large data sets can be lengthy (it takes 23 minutes on a 4 3GHz CPU-core Dell Workstation with 16GB memory), we comment the following line out:

```
> #Realdata$fembi.o <- DoFEMbi(intFEM.o,
> #                            nseeds=100,gamma=0.5,nMC=1000,sizeR.v=c(1,100),
> #                            minsizeOUT=10,writeOUT=TRUE,nameSTUDY="TCGA-EC",ew.v=NULL);
```

The results are found in the *fembi.o* entry of *Realdata*. The algorithm predicts a number of FEM modules. Use the following command to display their size and elements.

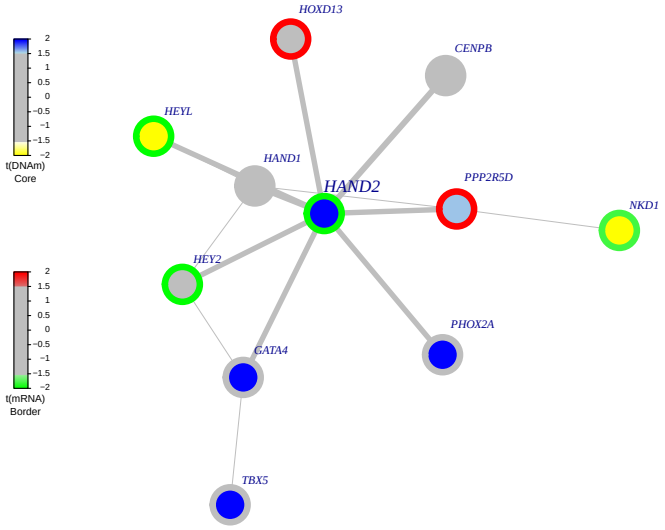
```
> Realdata$fembi.o$fem
```

The details of the modules can also be seen using:

```
> Realdata$fembi.o$topmod
```

In order to illustrate the modules graphically, the user can invoke the function *FemModShow*, which will generate a pdf figure of the module in your working directory and also return an *graphNEL* object which includes methylation and expression color schemes. The *graphNEL* class is defined in the bioconductor graph package, user can use *igraph.from.graphNEL* to convert it to the igraph objects. For instance, the algorithm inferred a module centred around the gene *HAND2*, which has been demonstrated to be causally implicated in the development of endometrial cancer [**?**]. Thus, given its importance, we generate a detailed network plot of this module:

```
> library("marray");
> library("corrplot");
> HAND2.mod<-Realdata$fembi.o$topmod$HAND2;
> HAND2.graphNEL.o=FemModShow(Realdata$fembi.o$topmod$HAND2,name="HAND2",Realdata$fembi.o)
```



Depicted is the functional epigenetic module centred around seed gene HAND2. Edge widths are proportional to the average statistic of the genes making up the edge. Node colours denote the differential DNA methylation statistics as indicated. Border colors denote the differential expression statistics. Observe that despite many nodes exhibiting differential methylation and differential expression, only HAND2, exhibits the expected anticorrelation with hypermethylation (blue) leading to underexpression (green). See Jones et al [**?**] for the functional, biological and clinical significance of *HAND2* in endometrial cancer.

The user can generate all the modules' graphs by

```
> #for(m in 1:length(names(Realdata$fembi.o$topmod))){
> #FemModShow(Realdata$fembi.o$topmod[[m]],
> #name=names(Realdata$fembi.o$topmod)[m],Realdata$fembi.o)}
```

In the above examples we provided the statistics and adjacency matrix input objects. However, in most circumstances, we would need to generate the statistics of differential DNA methylation and gene expression and subsequently integrate them with the network. The integration with the network requires that statistics be generated at the gene-level. This is what the functions *GenStatM* and *GenStatR* do. In the case of *GenStatM*, the input arguments are:

- dnaM.m: a normalised Illumina 450k DNA methylation data matrix, with rownames annotated to 450k probe IDs.
- pheno.v: phenotype vector corresponding to the samples/columns of dnaM.m.

- chiptype: A parameter specifying the the input DNA methylation data matrix, it should be either "450k" for Illumina 450k matrix or "EPIC" for Illumina EPIC matrix.

The function then generates for all pairwise contrasts/levels of the phenotype vector,tables of top-ranked genes. The function also returns the average beta-matrix with rows labeling genes. Similarly, the input arguments of *GenStatR* are:

- exp.m: normalized gene expression data matrix with rownames annotated to NCBI Entrez gene IDs. If the mapped Entrez gene IDs are not unique, the function will average values of the same Entrez gene ID.
- pheno.v: phenotype vector corresponding to the columns of exp.m.

As before, the function then generates for all pairwise contrasts/levels of the phenotype vector, ranked tables of top-ranked genes. The function also returns the average expression matrix with rows labeling unique genes.
Suppose the output of *GenStatM* and *GenStatR* are stored in objects *statM.o* and *statR.o*, respectively. Next step is then to integrate the statistics with the network of gene relations, specified by an adjacency matrix *adj.m*. This is accomplished with the *DoIntFEM450k* function:

```
> #DoIntFEM450k(statM.o,statR.o,adj.m,cM,cR)
```

Where we have also specified two integers, *cM* and *cR*, which select the appropriate contrasts in the *statM.o* and *statR.o* objects. For instance, if our phenotype vector consists of 3 categories (say, normal, cancer and metastasis), then there are 0.5*3*2=3 pairwise comparisons. We then need to check which contrast/column in *statM.o$cont* and *statR.o$cont* corresponds to the one of interest, and assign this integer to *cM* and *cR*, respectively.
We should also point out that the adjacency matrix represents a network of gene relationships (e.g. a PPI network) with rownames/colnames annotated to NCBI Entrez gene IDs. For instance, the PPI network can be derived from the Pathway Commons resource[**?**] and its construction could follow the procedure described in [**?**]. The PPI network used in previous papers is available at *http://sourceforge.net/projects/funepimod/files/*. The file name is *hprdAsigH*Rd*. This particular PPI network consists of 8434 genes annotated to NCBI Entrez identifiers, and is sparse containing 303600 documented interactions (edges). Users can also use a total different network of gene relation.
The output of *DoIntFEM450k* is a list with following entries:

- statM: matrix of DNA methylation moderated t-statistics and P-values for the genes in the integrated network
- statR: matrix of gene expression moderated t-statistics and P-values for the genes in the integrated network
- adj: adjacency matrix of the maximally connected integrated network (at present only maximally connected subnetwork is used).
- avexp: average expression data matrix mapped to unique Entrez IDs
- avbeta: average DNA methylation data matrix mapped to unique Entrez IDs

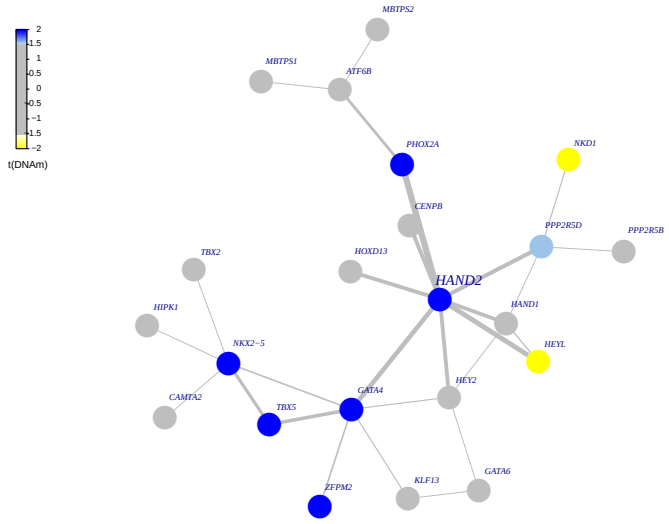
4 EpiMod and ExpMod

It may be that we only wish to infer either differential methylation or differential expression interactome hotspots. To this end, we provide specific functions, i.e. *DoIntEpi450K* to do the integration at the DNA methylation level only. Indeed, *DoIntEpi450K* is the same as *DoIntFEM450k*, except that we do not need the *statR.o* argument. In this case, the edge weights in the interactome network reflect the combined differential methylation statistics (absolute values) of the genes making up the edge. The output of *DoIntFEM450k* would then be used as input to the function *DoEpiMod*. Once we have run *DoEpiMod*, we can use *FemModShow* to show the top modules. The usage and arguments of *FemModShow* for EpiMod is same as previous decribed. You just need to add an argument "mode" and set "mode" = "Epi" (as *FemModShow* has three modes, the default one being "integration", which is the one to use with *DoFEMbi*). The "Epi" mode means *FemModShow* will render the Epi-modules generated by *DoEpiMod*.

The workflow applying GenStatM, DoIntEpi450k and DoEpiMod would be something like:

```
> #statM.o <- GenStatM(dnaM.m,phenoM.v,"450k");
> #intEpi.o=DoIntEpi450k(statM.o,adj.m,c=1)
> #EpiMod.o=DoEpiMod(intEpi.o,
> #                          nseeds=100,gamma=0.5,nMC=1000,sizeR.
> #                          v=c(1,100), minsizeOUT=10,writeOUT=TRUE,
> #                          nameSTUDY="TCGA-EC",ew.v=NULL);
```

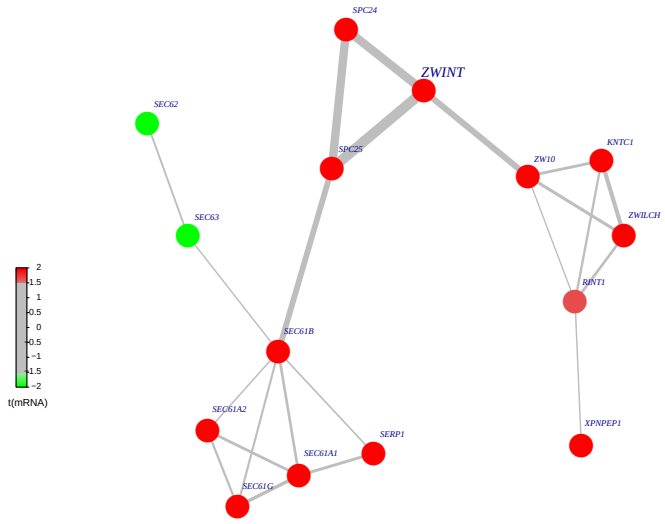
The DoIntEpi450k example data will be available later in an experimental package. In this case, application to the 450K methylation data of 118 endometrial cancers and 17 normal endometrials, results in a module, also centred around *HAND2*:



Depicted is the HAND2 Epi-module which contains many interacting members, most of which are hypermethylated in cancer compared to normal tissue:

If we were interested in inferring differential mRNA expression hotspots, you would run *DoExpMod*. As above, first you should run *GenStatR* and *DoIntExp* functions to generate statistics and integrate these with the network adjacency matrix. For instance, the workflow could look like:

```
> #statR.o <- GenStatR(exp.m,pheno.v);
> #intExp.o=DoIntExp(statR.o,adj.m)
> #ExpMod.o=DoExpMod(intExp.o,
> #                   nseeds=100,gamma=0.5,nMC=1000,
> #                   sizeR.v=c(1,100),minsizeOUT=10,
> #                   writeOUT=TRUE,nameSTUDY="TCGA-EC",ew.v=NULL)
> #
```



There is one ExpMod example, a ZWINT-Centered Interactome Hotspot. In this case, application to the expression data of 118 endometrial cancers and 17 normal endometrials from TCGA, results in a module, centered around ZWINT, which is a known component of the kinetochore complex required for the mitotic spindle checkpoint and thus regulates cell proliferation.

5 Integration with "minfi" package

The function *GenStatM* uses the normalised DNA methylation 450k data matrix with rownames annotated to 450k probe IDs. This kind of normalized 450k data matrix can be generated from raw data by many different tools. For instance, one could use *minfi*, an existing Bioconductor package [**?**]. The simplified workflow would be something like:

```
> #library(minfi);
> #require(IlluminaHumanMethylation450kmanifest);
>
> #baseDIR <- getwd();# the base dir of the Rawdata
> #setwd(baseDIR);
> #targets <- read.450k.sheet(baseDIR);#read the csv file.
> #RGset <- read.450k.exp(baseDIR); #Reads an entire 450k experiment
> #                                  using a sample sheet
> #MSet.raw <- preprocessRaw(RGset);#Converts the Red/Green channel for an Illumina
> #                                  methylation array into methylation signal,
> #                                  without using any normalization
```

```
> #beta.m <- getBeta(MSet.raw,type = "Illumina");# get normalized beta
> #pval.m <- detectionP(RGset,type ="m+u")
```

Before passing on the beta.m object to GenStatM, we recommend adjusting the data for the type-2 probe bias, using for instance BMIQ [**?**]. BMIQ is freely available from either http://sourceforge.net/p/bmiq/, or the ChAMP Bioconductor package [**?**].