LumiWCluster Package Example (Version 1.0)

Pei Fen Kuan

June 18, 2010

1 Introduction

This is an example of using the `LumiWCluster` package in `R`. The `LumiWCluster` package implements the proposed weighted model based clustering for Illumina Methylation BeadArray (Kuan et al., 2010). There are two parts to this package. The first part provides function to normalize the methylation data from the Illumina GoldenGate platform. We will include the normalization for Illumina Infinium platform in future. The second part provides function to identify subgroups with distinct methylation profiles in Illumina BeadArray. It works for both Infinium and GoldenGate platform, as well as any other platforms including gene expression arrays. The core of the algorithm is based on the paper by Wang and Zhu (2008), which automatically selects important CpGs/probes. `LumiWCluster` incorporates the detection p-values (a quality measure of probe performance reported by Illumina BeadStudio or GenomeStudio) systematically in the clustering algorithm and avoids arbitrary cutoff for excluding unreliable probes. Further details and motivation can be found in Kuan et al. (2010). To load the package

```
> library(LumiWCluster)
```

An example of input data file (subset of 6 ovarian cancer samples from Houshdaran et al. (2009).

```
> data(ovarian.dat)
```

```
> str(ovarian.dat)
List of 4
 $ Probe_ID   : Factor w/ 1505 levels "AATK_E63_R","AATK_P519_R",..: 1 2 3 4 5 6 ...
 $ Sample_ID  : chr [1:6] "Cell Line" "Cell Line" "Endometrioid" "Endometrioid" ...
 $ beta       : num [1:1505, 1:6] 9.87e-01 9.70e-01 9.41e-01 1.50e-08 2.10e-08 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:6] "CL_2008.3" "CL_A1847.3" "E_T3.3" "E_T4.3" ...
 $ det.p.value: num [1:1505, 1:6] 3.68e-38 3.68e-38 1.22e-10 1.14e-33 6.34e-18 ...
```

```
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:6] "CL_2008.1" "CL_A1847.1" "E_T3.1" "E_T4.1" ...
```

The input file is a list of 4:

- `Probe_ID`: a vector of CpG/probe names
- `Sample_ID`: a vector of sample names. In this example, we have a total of 6 samples.
- `beta`: a matrix of (average) beta values representing the methylation levels reported by Illumina BeadStudio or GenomeStudio, where $0 \leq \text{beta} \leq 1$. Each column is the observations for a sample.
- `det.p.value`: a matrix of detection p-values reported by Illumina BeadStudio or GenomeStudio. Each column is the observations for a sample.

2 Normalization of Illumina GoldenGate Methylation BeadArray

The current version implements normalization for sequence length and GC content for Illumina GoldenGate platform. The `LumiWCluster` package provides the design file for this platform. We will include other platform (e.g., Infinium and VeraCode) in future. To load the design file

```
> data(GoldenGateDesignFile)
```

```
> str(GoldenGateDesignFile)
'data.frame':   1505 obs. of  3 variables:
 $ Probe_ID: Factor w/ 1505 levels "AATK_E63_R","AATK_P519_R",..: 1 2 3 4 5 6 ...
 $ L       : int  58 46 46 48 57 55 52 56 54 53 ...
 $ GC      : num  0.638 0.674 0.783 0.562 0.491 ...
```

It is important that `Probe_ID` in `GoldenGateDesignFile` matches `Probe_ID` names in the input file above. To normalize the data

```
> norm.dat <- normalize.GoldenGate(Probe_ID = ovarian.dat$Probe_ID,
    beta = ovarian.dat$beta, det.p.value = ovarian.dat$det.p.value,
    GoldenGateDesignFile, plot.option = FALSE)
```

```
> str(norm.dat)
List of 4
 $ norm.beta     : num [1:1505, 1:6] 9.85e-01 9.81e-01 9.74e-01 ...
 $ norm.logitbeta: num [1:1505, 1:6] 4.17 3.92 3.62 -18.06 -18.38 ...
```

```
$ w              : Named num [1:6] 0.135 0.153 0.129 0.188 0.198 ...
 ..- attr(*, "names")= chr [1:6] "CL_2008.1" "CL_A1847.1" "E_T3.1" ...
$ g              : num [1:1505] 0.000845 0.000845 0.000331 0.000715 ...
```

`w` and `g` are functions of detection p-values representing the weight for each sample and CpG, respectively. `norm.logitbeta` is the logit transformed normalized beta values, which will be used in the clustering algorithm. If `plot.option = TRUE` in the function `normalize.GoldenGate`, boxplots comparing the beta values against sequence length and GC content before and after normalization will be generated. Let us illustrate this on the positive control sample from Houshdaran et al. (2009)

```
> data(posctrl.dat)
> posctrl.norm <- normalize.GoldenGate(Probe_ID =posctrl.dat$Probe_ID,
    beta = posctrl.dat$beta, det.p.value = posctrl.dat$det.p.value,
    GoldenGateDesignFile, plot.option = TRUE)
```

which will generate Figure 1.

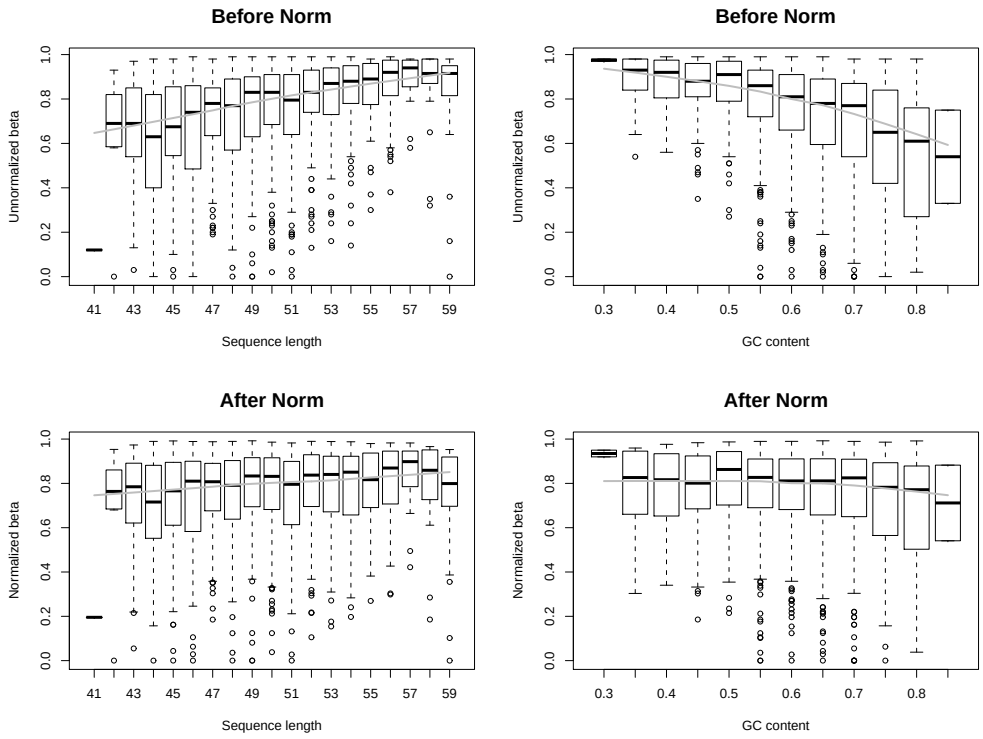

Figure 1: Effects of sequence length and GC content before and after normalization.

3 Weighted model based clustering

The main function that implements the weighted model based clustering in Kuan et al. (2010) is `LumiWCluster`. Argument 1 is the data set to be clustered. It can be from any platform, normalized or unnormalized, as long as the observations can be assumed to be normally distributed. For Illumina Methylation BeadArray, we will use a logit transformed beta values. Arguments 2 and 3 are the weights for each sample and probe, respectively. If missing, the function will assign all samples and probes to have equal weight. Arguments 4 and 5 are the number of candidate number of clusters `K` and range of tuning parameters `lambda` to consider. Arguments 6-8 are logical values (either `TRUE` or `FALSE`):

- If `adaptive = TRUE`, adaptive version of Wang and Zhu (2008) will be implemented. We recommend trying both versions.
- If `center = TRUE`, the data will be centered. For most cases, we should set `center = TRUE`.
- If `normalize = TRUE`, the data will be scaled to have uni-variance.

Arguments 9-11 are the control for the EM algorithm in `LumiWCluster`. We should set `epsilon` to be a very small number (e.g. 1e-6). If the algorithm does not converge, increase `max.iter`. If `trace = TRUE`, each EM iteration will be printed. An example to run `LumiWCluster`

```
> fit.LumiWCluster <- LumiWCluster(norm.dat$norm.logitbeta, norm.dat$w,
    norm.dat$g, K = c(2:3), lambda = c(0.1,1), adaptive = TRUE, center = TRUE,
    normalize = FALSE, epsilon = 1e-4, max.iter = 2000, trace = FALSE)

Searching for K= 2 3 ...
Running for K = 2 and lambda = 0.1
The algorithm converges
Running for K = 2 and lambda = 1
The algorithm converges
Running for K = 3 and lambda = 0.1
The algorithm converges
Running for K = 3 and lambda = 1
The algorithm converges
DONE! Optimal K = 3
```

Here we consider `K`=2 and 3 clusters, `lambda`=0.1 and 1 in the penalty function. Large `lambda` will shrink more CpGs to zero. The optimal `K` and `lambda` are determined from the BIC scores, and are given by `fit.LumiWCluster$opt.K` and `fit.LumiWCluster$opt.lambda`. In this example, optimal `K` and `lambda` are 3 and 1, respectively.

```
> str(fit.LumiWCluster)
List of 14
 $ opt.K       : int 3
 $ opt.lambda  : num 1
 $ clusterID   : int [1:6] 1 1 2 2 3 3
 $ BIC         : num 30714
 $ BIC_iter    : num [1:2, 1:2] 37000 31354 36529 30714
 $ lambda      : num [1:2] 0.1 1
 $ K           : int [1:2] 2 3
 $ converge    : num 1
 $ pi.vec      : num [1:3] 0.288 0.317 0.395
 $ mu          : num [1:3, 1:1505] 3.96 2.55 1.44 2.75 2.75 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : NULL
 $ mu.centering: num [1:3, 1:1505] 1.41 0 -1.11 0 0 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : NULL
 $ init.mu     : num [1:3, 1:1505] 1.708 -0.319 -1.389 1.145 -0.482 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : NULL
 $ sigma2.vec  : num [1:1505] 1.7102 1.5996 0.9897 0.0152 0.0496 ...
 $ iter        : num 366
```

The cluster membership is given by `fit.LumiWCluster$clusterID`. We also provide a function which summarizes the sample names within each cluster. The summary of cluster membership can be obtained using the function `ClusterMember`

```
> ClusterMember(fit.LumiWCluster, ovarian.dat$Sample_ID)

Optimal number of clusters: 3
Sample IDs in Cluster 1
Cell Line Cell Line
Sample IDs in Cluster 2
Endometrioid Endometrioid
Sample IDs in Cluster 3
Serous Serous
```

In this example, the 3 distinct samples (Serous, Endometrioid, Cell Line) are clustered correctly.

Finally, the list of probes/CpGs which are informative for clustering the samples (i.e., CpGs which are not shrunk to zero) can be retrieved using the function `InformativeCpG`.

```
> Sig_Probes <- InformativeCpG(fit.LumiWCluster, ovarian.dat$Probe_ID)

> length(Sig_Probes)
[1] 996

> Sig_Probes[1:5]
[1] "AATK_E63_R"   "AATK_P709_R"  "ABCA1_E120_R" "ABCA1_P45_F"  "ABCB4_P892_F"
```

In this example, 996 CpGs remain after running `LumiWCluster`.

References

Houshdaran, S., Hawley, S., Palmer, C., Campan, M., Olsen, M., Ventura, A., Knudsen, B., Drescher, C., Urban, N., Brown, P., and Laird, P. (2009), "DNA Methylation Profiles of Ovarian Epithelial Carcinoma Tumors and Cell Lines," *PLoS ONE*, 5, e9359.

Kuan, P., Wang, S., and Chu, H. (2010), "A statistical framework for Illumina DNA methylation," *Technical Report, Department of Biostatistics, UNC-Chapel Hill*.

Wang, S. and Zhu, J. (2008), "Variable selection for model-based high dimensional clustering and its application to microarray data," *Biometrics*, 64, 440–448.