epihet:

An R Package for Calculating and Analyzing

the Epigenetic Heterogeneity of Cancer Cells

Xiaowen Chen,Haitham Ashoor,Ryan Musich,Mingsheng Zhang,Jiahui Wang,Sheng Li

Oct 2020

Contents

1 Introduction

This manual introduces the epihet package and shows how the package can be used to calculate epigenetic heterogeneity of cells and visualize the results through various types of graphs. This package was designed to use output from methclone, a C++ library that analyzes the evolution of epialleles using Bisulfite Sequencing methylation data.

2 Installation

1. Download the package from Bioconductor.

```
if (!requireNamespace("BiocManager", quietly = TRUE))
install.packages("BiocManager")
BiocManager::install("epihet")
```

Or install the development version of the package from Github.

```
BiocManager::install("TheJacksonLaboratory/epihet")
```

2. Load the package into R session.

```
library(epihet)
```

3 Background

3.1 DNA Methylation & Epigenetic Heterogeneity

Methylation occurs on the DNA strand where a methyl group attaches to the cytosine of a bonded cytosine and guanine pairing. Normal methylation patterns assure the proper regulation of gene expression and stable gene silencing. Areas of DNA that have a high percentage of methylation are considered to be hypermethylated and have been associated with the silencing of certain tumor-suppressor genes. Areas with lower percentages of methylation are called hypomethylated and are associated with cell transformation.

Recently, cell-to-cell variations in cancer patients have been proposed to contribute to the treatment failure as it may provide an alternative trajectory for the cancer cells to escape therapy. In addition to the genetic allelic heterogeneity, it has been reported that cancer cells may display various epigenome status within the same patients. Specifically, the epigenetic heterogeneity and dynamics measured by the phased DNA methylation patterns have been reported to associate with clinical outcome in acute myeloid leukemia (AML), chronic lymphocytic leukemia (CLL), diffuse large B cell lymphoma, and Ewing sarcoma (Sheffield et al, 2017). As the cancer cell evolves from

diagnosis to relapse, methylation patterns at a given locus can further disrupt promoter regions and gene regulation (Pan et al, 2015). These differing methylation patterns on the same locus/gene are called epialleles and are key to determine epigenetic heterogeneity. Previous studies have shown that the methylation patterns of cancer cells on relapse can vastly differ from the patterns of the same cells on diagnosis (Li et al, 2014). This package uses epialleles consisting of four adjacent cytosine bases to create 16 (either a methylated or unmethylated cytosine across four unique loci) distinct methylation patterns to analyze epigenetic heterogeneity in the samples.

3.2 Important Variables for Analysis

To determine epigenetic heterogeneity, epihet uses the following three variables: proportion of discordant reads (PDR), epipolymorphism, and Shannon entropy values. By comparing the similarities and differences between these values at the same locus across multiple samples, the extent of heterogeneity between the samples can be analyzed by the user.

3.2.1 Proportion of Discordant Reads (PDR)

One important variable to analyze for heterogeneity between cells is the proportion of discordant reads (PDR). PDR at each locus is defined as the proportion of discordant reads compared to the number of total reads from that locus (Landau et al, 2014). A bisulfite sequencing read at a given locus is classified as a concordant read or a discordant read. Here, a concordant read is one that shows fully unmethylated or fully methylated sites at a given locus, such as a four methylated cytosines. A discordant read is one that shows varying states of methylated and unmethylated regions at a given locus, such as a methylated cytosine followed by three unmethylated cytosines. PDR values can then be used for analyzing epigenetic heterogeneity because a greater value for PDR corresponds to a greater amount of discord within the sample. With a greater level of discord, the sample is considered to be more heterogeneous in nature and could lead to adverse clinical outcomes upon treatment.

In this package, epihet calculates PDR value for each locus of one sample from the proportion of each methylation pattern. For a given locus, epihet sums the percentage of reads support all discordant methylation patterns to obtain PDR value. The resulting value is between 0 and 1.

3.2.2 Epipolymorphism

When analyzing epigenetic heterogeneity, epigenetic polymorphism, or epipolymorphism, is an important variable to calculate. Epipolymorphism is defined as "the probability that two epialleles randomly sampled from the locus differ from each other" (Landan et al, 2012). Calculating the epipolymorphism value for a given locus uses the proportion of each methylation pattern to determine how likely the methylation patterns differ across multiple reads. Epipolymorphism value ranges from 0 to 1 with larger values being associated with larger differences in methylation patterns.

In this package, epipolymorphism value for each locus is calculated based on the formula described by Landen

et al. (Landan et al, 2012). For each locus, the proportion of each methylation pattern is squared then summed together and subtracted from 1. The resulting value is the epipolymorphism value for the given locus.

3.2.3 Shannon Entropy

An important aspect of epigenetic heterogeneity is knowing how diverse a given epiallele is compared to other epialleles. This variable is best tracked through Shannon's information entropy. To calculate Shannon entropy, the proportion for each methylation pattern of a given sample must be known. Next, each proportion is multiplied by the logarithmic result of that proportion. These products are summed together and the result is negated to find the value for Shannon entropy. This result is equal to the exponential that corresponds to the "effective number" of epialleles needed to generate an equivalent Shannon entropy value based on the given methylation pattern proportions (Sherwin, 2010). A large value for Shannon entropy corresponds to a greater number of epiallele patterns needed and would be considered more diverse and, therefore, more heterogeneous. A low value for Shannon entropy corresponds to a lesser amount of epiallele patterns needed and be considered less heterogeneous based on its epigenetics. A value of 0 for Shannon entropy shows that the locus only contains a single methylation pattern across all the reads.

4 Building the Comparison Matrix

In order to prepare for analysis, epihet has built-in functions that take in multiple txt files for multiple samples from the program methclone and transforms the data into a large matrix that contains the location information and PDR, epipolymorphism, and Shannon entropy values for each locus that is shared between the inputted samples. The following sections provide examples on how to use the functions in epihet to build the comparison matrix and prepare the data for analysis.

To begin using epihet, the user should already have fed bisulfite sequencing data to methclone, which outputs compressed text file containing epiallele patterns and the percentage of reads supporting each of epiallele patterns at the genomic locus in the sample. Included in epihet are example files used for the sample code in this vignette, which include only the result for epialleles on chromosome 22 for two normal samples (N1, N2) and two AML patients with the CEBPA_sil (isocitrate dehydrogenase 2) mutation (D2238, D2668). The following lines of code can be run to obtain the files and their corresponding ID names:

```
files = c(system.file("extdata","D-2238.chr22.region.methClone_out.gz",package = "epihet"),
          system.file("extdata","D-2668.chr22.region.methClone_out.gz",package = "epihet"),
          system.file("extdata","N-1.chr22.region.methClone_out.gz",package = "epihet"),
          system.file("extdata","N-2.chr22.region.methClone_out.gz",package = "epihet"))
ids = epihet::splitn(basename(files),"[.]",1)
```

```
##
```

4.1 Creating a List of GenomicRanges Objects

To use epihet to calculate epigenetic heterogeneity, users need to assign the compressed text file suffix to methClone_out.gz. Then, makeGR function in epihet reads in the compressed text files and creates a GenomicRanges object for each file. The GenomicRanges objects are returned in a list. The makeGR function is the first step for epihet's pipeline as the result is needed for input in the comparison matrix function, compMatrix. The function takes a vector of file paths, 'files', and a vector of sample names that corresponds to the files, 'ids'. The following code creates a list of GenomicRanges objects using epihet's sample files:

```
GR.List = epihet::makeGR(files = files, ids = ids,
                         cores = 1, sve = FALSE)
```

One row in each GenomicRanges object in the list contains the information of one locus, including chromosome number, range of the strand, and strand type, as well as six additional columns that include the locus ID, number of reads, average methylation percentage of the locus, and PDR, epipolymorphism, and Shannon entropy values of the locus. If makeGR is being used to process many files, the variable 'cores' can be changed to specify the number of cores to use for parallel execution. If the resulting GenomicRanges list must be saved for later use, the variable 'sve' can be set to TRUE and the result will be saved to a .rda file.

4.2 Generating Comparison Matrix

The list of GenomicRanges objects can be used to generate the comparison matrix that is used for the analysis functions included in epihet. The comparison matrix is created using the 'compMatrix' function which locates epialleles that are shared by at least a certain percentage of the samples and organizing the data into sections for read number, average methylation levels, PDR, epipolymorphism, and Shannon entropy values at these matching loci. The following code creates a comparison matrix from the GenomicRanges list created in the previous section and finds epialleles that are present in 100% of the samples (p = 1):

```
comp.Matrix = epihet::compMatrix(epi.gr = GR.List, outprefix = NULL,
                                 readNumber = 60, p = 1,
                                 cores = 1, sve = FALSE)
```

The parameter 'p' takes values from 0 to 1. The comparison matrix comp.Matrix contains epigenetic heterogeneity values of the locus shared by at least 100p percent of samples. For example, if the loci in comp.Matrix are to be shared by at least half of the samples, then 'p' should be set to 0.50. If the resulting comparison matrix needs to be saved, 'sve' can be set to TRUE and 'outprefix' can be specified to add a prefix to the resulting .rda file. If

large files are being used, the matrix can be created using multiple cores to speed up the execution as specified by the 'cores' variable.

Additionally, the users can add other epigenetic heterogeneity metrics they develop or interest in this step. If there are multiple customized metrics, all the analysis will be performed one measurement one time. Firstly, the customized metrics on all the loci for one sample are saved as an element of a list. The example data 'myValues' are provided by epihet. Here we denoted the customized metrics as myValues. Then, we added the myValues into the metadata of the GRanges object from GR.List sample by sample. Finally, the users can get the comparison matrix including the customized metrics through setting the argument 'metrics' to a vector including value 'myValues'. When the comparison matrix is generated, the users can also perform other downstream analysis using the customized metrics through setting the argument 'value' to 'myValues'.

```
data(myValues,package = "epihet")
myGR.List<-list()
for (n in names(GR.List)) {
  tmp<-GR.List[[n]]
  tmp$values.myValues<-myValues[[n]]
  myGR.List[[n]]<-tmp
}
mycomp.Matrix = epihet::compMatrix(epi.gr = myGR.List, outprefix = NULL,
                                   readNumber = 60, p = 1,
                                   metrics = c("read1","meth1", "pdr", "epipoly",
                                               "shannon","myValues"),
                                   cores = 1, sve = FALSE)
```

4.3 Creating Single GenomicRanges Object

Instead of creating a list of GenomicRanges objects, a single GenomicRanges object can be created using the 'readGR' function. The function takes a vector of files, 'files', and a vector of IDs, 'ids', that corresponds to the files. The index of the file, 'n', to be used for creating the GenomicRanges object is also needed. The following code generates the GenomicRanges object for the third file in the vector:

```
GR.Object = epihet::readGR(files = files, ids = ids, n = 3)
```

4.4 Simple Summary for Two Samples

If only a quick comparison is needed between two samples, epihet's summarize function can be used to provide a simple overview of how the values of one sample correlate to the values of another sample. The summarize

function works by taking in two GenomicRanges objects and the values of each object that will be compared. The possible values for the summarize function are 'pdr', 'epipoly', and 'shannon' that correspond to the data values of the same name. Two different cutoffs specify read coverage of a locus which is included in the summary. The output of summarize is a dataframe that contains the mean of the first and second value of common loci between the two samples with a number of reads greater than both cutoffs. The correlation between value one and value two at these loci are also calculated as well as the number of common loci at both read cutoffs. The following code generates a summary of PDR and epipolymorphism values for the first and second sample (D2238, D2668) in the GR.List created earlier with read coverage cutoffs of 10 and 60:

```
summary = epihet::summarize(gr1 = GR.List[[1]], gr2 = GR.List[[2]],
                            value1 = 'pdr', value2 = 'epipoly',
                            cutoff1 = 10, cutoff2 = 60)
```

The result of this code shows that the PDR and epipolymorphism values have a high positive correlation at both cutoffs.

5 Analyzing the Data

Once the comparison matrix has been generated for the sample data, the epigenetic heterogeneity can be analyzed by using epihet's built-in analysis functions. With epihet, the data in the comparison matrix can be used to create boxplots, heat maps, and PCA, t-SNE, and MA plots. For most of the functions used for analysis in epihet, annotation information can be added as a parameter that will annotate or group the data based on the cancer type or subtype information provided by the user. The annotation information must be a dataframe with row names as the samples, data entries as the corresponding group annotations. For our example, the subtype groupings for the samples will be used as annotations and can be created by the following code:

```
subtype = data.frame(Type= c(rep('CEBPA_sil', 2), rep('Normal', 2)),
                     row.names = names(GR.List),stringsAsFactors = FALSE)
```

5.1 Creating Boxplots

A simple way to analyze how data is distributed across samples are comparative boxplot. By creating boxplot, one can easily analyze the spread, median, range, and find any potential outliers in the data. The boxplot function in epihet, called epiBox, is used to create a boxplot of a specific value, such as 'pdr', 'epipoly', or 'shannon', for each grouping of samples as inputted by the user. The following call to epiBox compares epipolymorphism values across the samples and creates the figure seen below:

```
epihet::epiBox(compare.matrix = comp.Matrix, value = 'epipoly',
               type = subtype, box.colors = NULL, add.points = FALSE,
               points.colors = NULL, pdf.height = 4, pdf.width = 4,
               sve = TRUE)
```

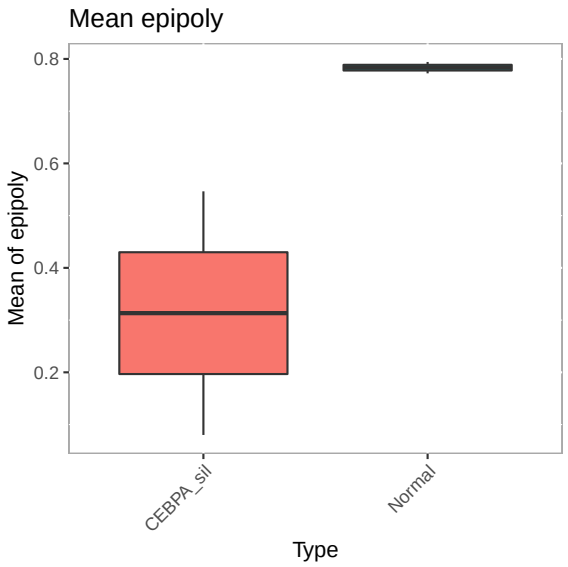

Figure 1: Boxplot of CEBPA_sil and Normal samples by epipolymorphism value.

As the figure 1 shows, the CEBPA_sil mutation samples have a range in epipolymorphism values ranging from 0.08 to 0.55, while the normal samples have a very small range with all average epipolymorphism values being around 0.78. The figure also shows a relatively large difference between the median epipolymorphism values between the two subtypes with CEBPA_sil mutation at 0.3133 and normal at 0.7833.

Some other important features of the epiBox function are adding the individual data points for each sample to the boxplots through the 'add.points' parameter, customizing colors of both the boxes and points by adding a vector of colors to both 'box.colors' and 'points.colors', and saving the resulting figure as a .pdf file using 'sve', 'pdf.height', and 'pdf.width'.

5.2 Generating a Heat Map

We can cluster the samples based on the epigenetic heterogeneity using the most variable genetic loci. The function epiMap uses pheatmap function (default value) to create a heatmap plot based the top user-inputted percent of loci with the highest standard deviation across all samples. In the example, the function uses epipolymorphism values to cluster the sample based on all the loci. The user can create the appropriate input for the parameter annotate.colors to color the samples by subtype information:

```
pmap = epihet::epiMap(compare.matrix = comp.Matrix,
                      value = 'epipoly',annotate = subtype,
                      clustering_distance_rows = "euclidean",
                      clustering_distance_cols = "euclidean",
                      clustering_method = "complete",annotate.colors = NA,
                      color= colorRampPalette(c("blue","white","red"))(1000),
                      loci.percent = 1, show.rows = FALSE,
                      show.columns = TRUE, font.size = 15,
                      pdf.height = 10, pdf.width = 10, sve = TRUE)
```

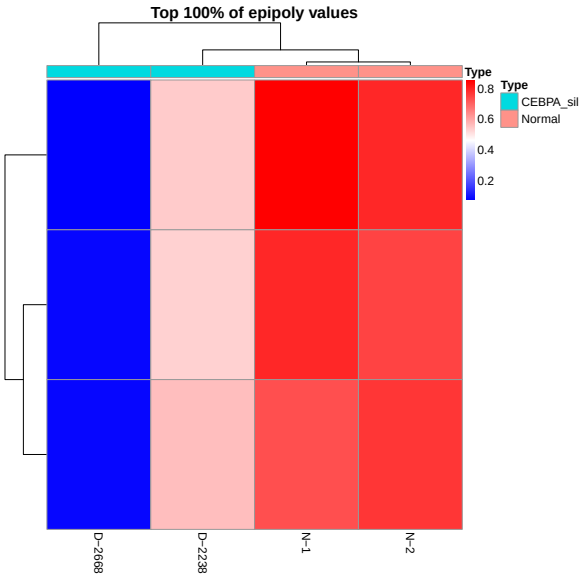

Figure 2: Heatmap of CEBPA_sil and Normal samples by epipolymorphism value.

Other features of epiMap include customizing the size of the font in the image through 'font.size', showing row or column names using 'show.rows' and 'show.columns', and the ability to save the image as a .pdf file through 'sve', 'pdf.height', and 'pdf.width'. The colors of the annotations can also be customized by creating a vector of colors and storing them in a list. The following code creates the appropriate input for the 'annotate.colors' parameter to color CEBPA_sil mutation samples in orange and normal samples in forestgreen:

```
box.colors=c("orange","forestgreen")
names(box.colors)=c("CEBPA_sil","Normal")
annotate.colors = list(Type=box.colors)
```

5.3 Graphing a PCA Plot

An important analysis for epigenetic heterogeneity is examining how the investigated samples are grouped based on PDR, epipolymorphism, and Shannon entropy values. This can be accomplished through a principle component analysis (PCA) plot for the comparison matrix. A PCA plot uses an orthogonal transformation to change the data values in the matrix to coordinates based on variance. The epiPCA function creates a PCA plot for either PDR, epipolymorphism, or Shannon entropy values and colors the points by an inputted annotation. The following code creates the PCA plot seen below for epipolymorphism values and colors the points on the plot based on subtype groupings:

```
suppressPackageStartupMessages(library(ggfortify))
epihet::epiPCA(compare.matrix = comp.Matrix, value = 'epipoly',
               type = subtype, points.colors = NULL,
               frames = FALSE, frames.colors = NULL,
               probability = FALSE, pdf.height = 4,
               pdf.width = 5, sve = TRUE)
```

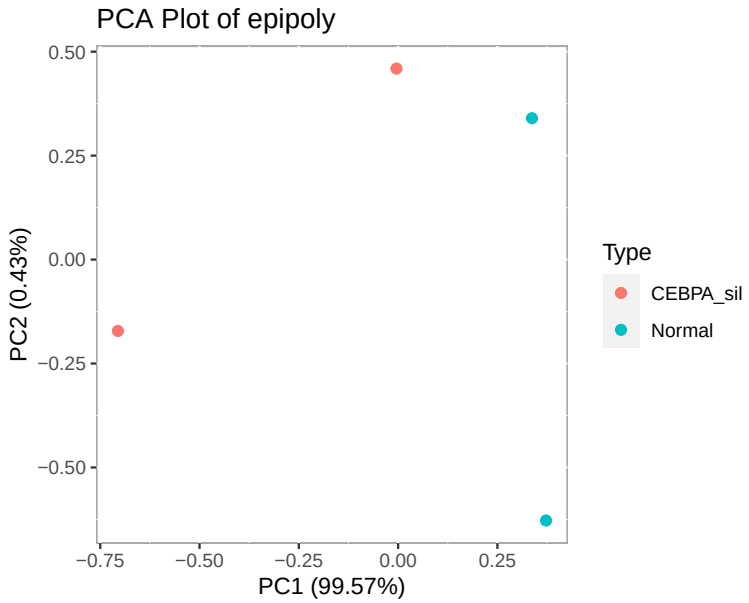

Figure 3: PCA Plot of CEBPA_sil and Normal samples by epipolymorphism value.

The PCA plot shows CEBPA_sil and normal samples can separate each other very well.

Other features of epiPCA include adding frames or probability ellipses to better define the groupings of annotations with 'frames' or 'probability', customizing the colors of the points and frames by adding a vector of colors to 'points.colors' and 'frames.colors', and saving the plot as a .pdf file using 'sve', 'pdf.height', and 'pdf.width'.

5.4 Graphing a t-SNE Plot

Another plot used to analyze how the samples are group based on a given value is a t-distributed stochastic neighbor embedding (t-SNE) plot. Similar to a PCA plot, a t-SNE plot is used for analyzing patterns in data by grouping points together, however, a t-SNE plot uses multiple dimensions for these groupings. The results are placed on a human-readable plot in 2-dimensions. The function in epihet used for t-SNE plot creation is epiTSNE. By using either PDR, epipolymorphism, or Shannon entropy values, epiTSNE can create a t-SNE plot and color them by an inputted annotation. The following code creates the t-SNE plot below using epipolymorphism values and colors the points based on their subtype information:

```
set.seed(42)
epihet::epiTSNE(compare.matrix = comp.Matrix, value = 'epipoly',
                type = subtype, points.colors = NULL, theta = 0.5,
                perplexity = 1, max_iter = 1000, pdf.height = 4,
                pdf.width = 5, sve = TRUE)
```

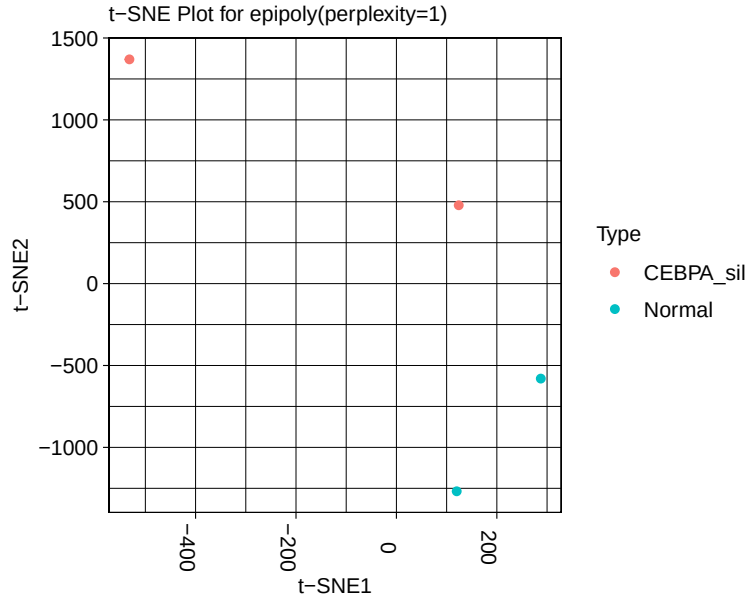

Figure 4: t-SNE Plot of CEBPA_sil and Normal samples.

The t-SNE plot shows the CEBPA_sil and normal samples cluster two different groups. As these two groups are at opposite ends of the t-SNE plot, one can assume that the epipolymorphism values for the CEBPA_sil mutation samples differ widely in comparison to the normal samples. This corresponds to a large difference in the number of methylation patterns found in the two subtypes.

Other features of epiTSNE include customizing the colors of the points by adding a vector of colors to 'points.colors', customizing the parameters of the Rtsne function used to generate the data for the t-SNE plot through 'theta', 'perplexity', and 'max_iter', and the option to save the resulting plot as a .pdf file using 'sve', 'pdf.height', and

'pdf.width'.

5.5 Identifying Differential Epigenetic Heterogeneity locus

The diffHet functions is the main function to identify differential epigenetic heterogeneity (DEH) locus. Depending on the measures of epigenetic heterogeneity user investigates, it will either use t-test or permutation test to calculate p-values. p-values will be adjusted using multiple test correction. When you identify the loci with differential PDR or epipolymorphism comparing test versus control samples, the function will use t-test. Otherwise, the function will employ EntropyExplorer R package to perform the permutation test. And the function also returns the mean epigenetic heterogeneity for each group and the mean epigenetic heterogeneity difference between test and control samples. The users can use the mean epigenetic heterogeneity difference and adjusted p-values to identify DEH loci. The following code creates the dataframe for all the loci containing differential epipolymorphism between normal and CEBPA_sil mutation samples for epipolymorphism values with a heterogeneity difference cutoff of 0.20:

```
samples=data.frame(Sample=colnames(comp.Matrix)[1:(length(comp.Matrix)-2)],
                   Genotype=c(rep ("CEBPA_sil", 2), rep ("Normal", 2)),
                   stringsAsFactors = FALSE)
rownames(samples)=samples$Sample
seed = sample(1:1e+06, 1)
set.seed(seed)
diff.het.matrix = epihet::diffHet(compare.matrix = comp.Matrix,
                                  value = 'epipoly', group1 = 'CEBPA_sil',
                                  group2 = 'Normal', subtype = samples,
                                  het.dif.cutoff = 0.20,
                                  permutations = 1000,
                                  p.adjust.method = 'fdr', cores = 1)

## [1] "using t-test to identify DEH loci"
## [1] "Finish p value calculation"
```

After calculating the heterogeneity difference and adjusted p-value for each locus, an MA plot can be created. For each locus, the average of the means heterogeneity for two groups versus the heterogeneity difference was plotted. DEH loci (adjusted p-values lower than the inputted cutoff) are highlighted in red. The following code creates the MA plot below using the heterogeneity difference cutoff of 0.2 and an adjusted p-value cutoff of 0.05:

```
data(diffhetmatrix,package="epihet")
epihet::epiMA(pval.matrix = diffhetmatrix, padjust.cutoff = 0.05,
              pch = 19, sve = TRUE,pointsize=1.5)
```

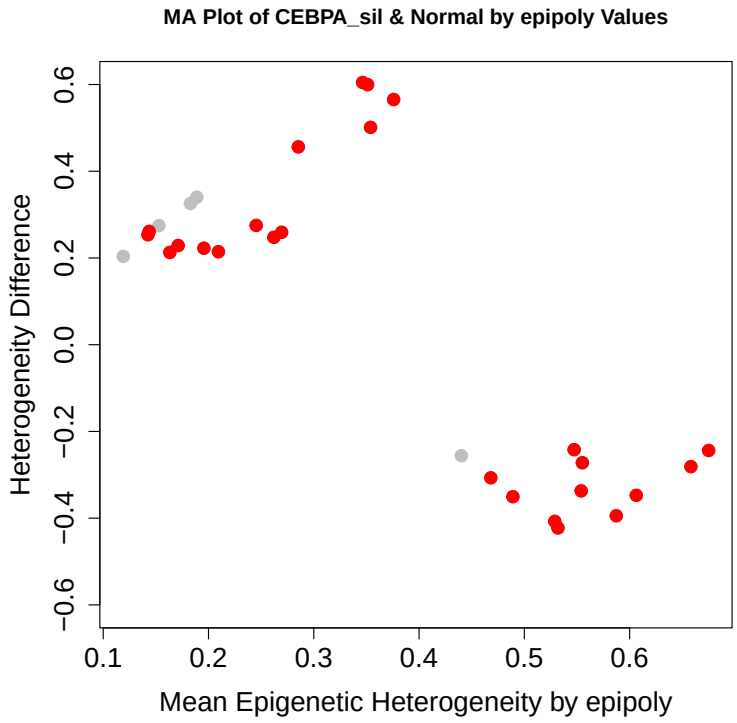

Figure 5: MA Plot of CEBPA_sil and Normal samples. Red dots represent DEH loci. Gray dots indicate non-DEH loci.

Through analysis of the MA plot, we can clearly see the distribution of the DEH loci. The red points with the negative heterogeneity difference values are decrease DEH loci. The red points with the positive heterogeneity difference values are increase DEH loci.

Other features of the diffHet function include specifying a cutoff for the heterogeneity difference through 'het.dif.cutoff', changing the method used to calculate adjusted p-values using 'p.adjust.method', and the ability to use multiple cores for parallel execution using 'cores'. If Shannon entropy values are to be examined, diffHet uses the Entropy-Explorer function to calculate the appropriate p-value for each locus. The variable for permutations in Entropy-Explorer can be modified using 'permutations'.

Other features for epiMA include specifying the adjusted p-value cutoff to find significant values using 'padjust.cutoff', changing the individual point designs for the plot using 'pch', and the ability to save the plot as a .pdf file using 'sve'.

6 Constructing co-epigenetic heterogeneity network

6.1 Network construction and module identification

We can construct co-epigenetic heterogeneity network based on the DEH loci using WGCNA R package. For algorithm details, please refer to the tutorial of WGCNA. Here, there are two methods to construct network, which were decided by the parameter node.type. One method calculates the co-epigenetic heterogeneity between any two DEH loci. Another one is to identify genes with genome region annotated by DEH loci and calculates the co-epigenetic heterogeneity between any two genes. Genome region can be promoter, CpG islands, CpG shores and so on. The epigenetic heterogeneity of one gene was measured with the average epigenetic heterogeneity of loci associated with the gene. At this case, annotation files in BED format are need for annotating your DEH loci. You can download annotation from UCSC table browser for your genome of interest. Then you should save the BED file as the GRanges object in R, and input it into the parameter annotation.obj. We also provide gene promoter files for Refseq genes. Additionally, users can also supply the clinical traits of patients in dataframe to the parameter datTraits, such as age, gender, survival time, to identify clinically significant modules. Then network can be obtained as follows:

```
suppressPackageStartupMessages(library(GenomicRanges))
suppressPackageStartupMessages(library(doParallel))
registerDoParallel(cores=1)
data(sharedmatrix,package="epihet")
data(DEH,package = "epihet")
data(datTraits,package = "epihet")
data(promoter,package = "epihet")
classes=data.frame(Sample=
                   c(colnames(sharedmatrix)[1:(length(sharedmatrix)-2)],
                   paste("N",1:14,sep = "-")),group=c(rep("CEBPA_sil",6),
                   rep("Normal",14)),stringsAsFactors = FALSE)
rownames(classes)=classes$Sample
epi.network=epihet::epiNetwork(node.type = "gene",DEH,sharedmatrix,
                               value = "epipoly",group="CEBPA_sil",
                               subtype=classes,datTraits = datTraits,
                               promoter,networktype = "signed",
                               method = "pearson",prefix="epipoly",
                               mergeCutHeight = 0.25,minModuleSize = 30)
```

This function is used to generate the co-epigenetic heterogeneity network and modules. It will return a list contain-

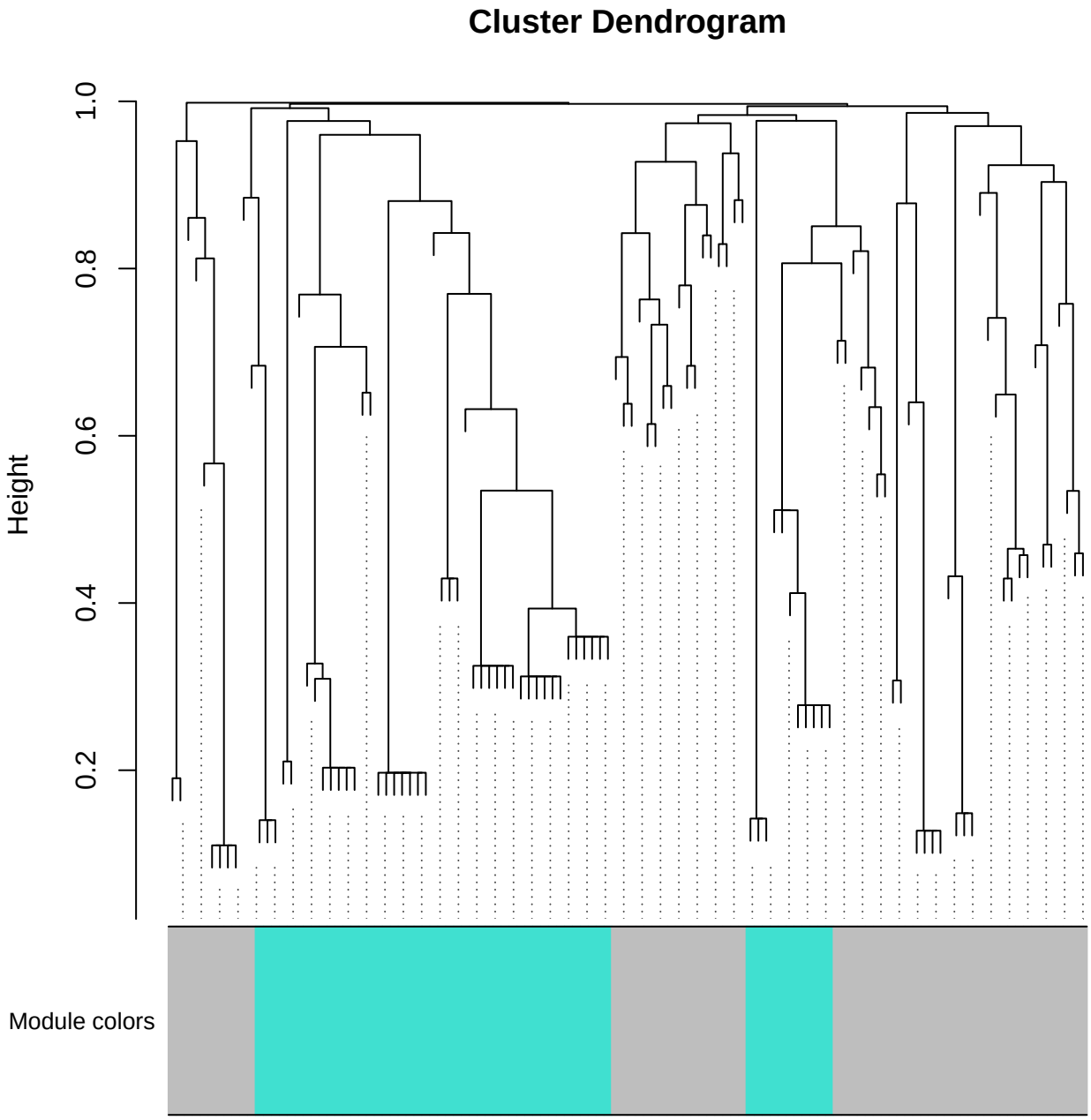

Figure 6: Clustering dendrogram of genes.

ing epigenetic heterogeneity matrix of patients, module information and genes of each module. At the same time, the function saves Topological Overlap Matrices(TOM) as RData format. And, it creates a clustering dendrogram of loci/genes showing module information through assigning different colors.

When users provided the external clinical traits, the clinically significant modules were identified. The result was visualized by the heatmap plot in the Figure 7. Each row in the table corresponds to a module, and each column to a trait. Numbers in the table report the correlations of the corresponding module eigengenes and traits, with the p values printed below the correlations in parentheses. The color legend denotes correlation.

The function also creates a bar plot showing the number of genes associated with DEH loci in each module in Figure 8.

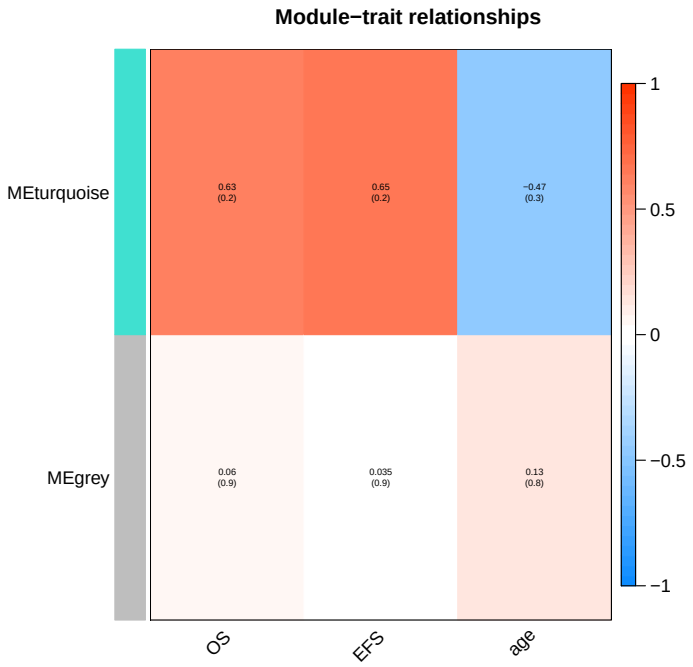

Figure 7: Heatmap showing the relationship between module eigengenes and clinical traits.

6.2 Module Visualization

We can also visualize the modules and perform the network topology analysis using the following function.

```
load("epipoly-block.1.RData")
module.topology=epihet::moduleVisual(TOM,
                                     value.matrix=epi.network$epimatrix,
                                     moduleColors=epi.network$module$color,
                                     mymodule="turquoise",cutoff=0.02,
                                     prefix='CEBPA_sil_epipoly',sve = TRUE)
```

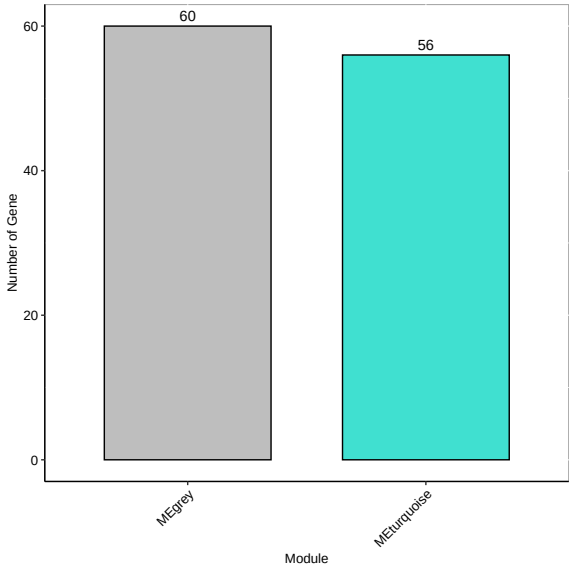

Figure 8: The bar plot showing the number of genes in each module.

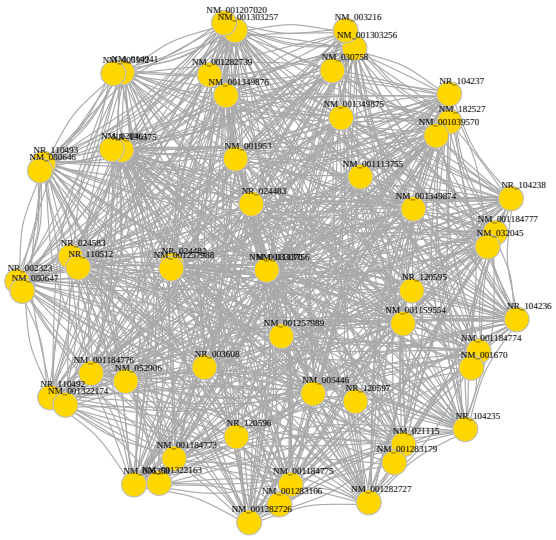

Figure 9: The lightgreen module.

Here, TOM file is loaded into R workspace, which was created by the above function epiNetwork. This function can also generate an edge file and a node file as the txt format. Users can specify network layout and style using Cytoscape.

6.3 Module Annotation

The epihet package contains a function to perform pathway enrichment analysis using a simple, single step. To run the function, ReactomePA needs to be installed before running the code. Here, ReactomePA needs entrez ID, so we firstly transform refseq ID to entrez ID using function bitr() in R package clusterProfiler.

```
suppressPackageStartupMessages(library(clusterProfiler))
gene=unique(epi.network$module$gene)
entrez=bitr(gene,fromType = "REFSEQ",toType = "ENTREZID",
            OrgDb = "org.Hs.eg.db")
genelist=epi.network$module
head(genelist)
genelist=merge(genelist,entrez,by.x="gene",by.y="REFSEQ")
genelist=unique(genelist[,c(4,2,3)])
head(genelist)
pathway = epihet::epiPathway(genelist,cutoff = 0.05,showCategory=8,
                             prefix="CEBPA_sil",pdf.height = 10,
                             pdf.width = 10)
```

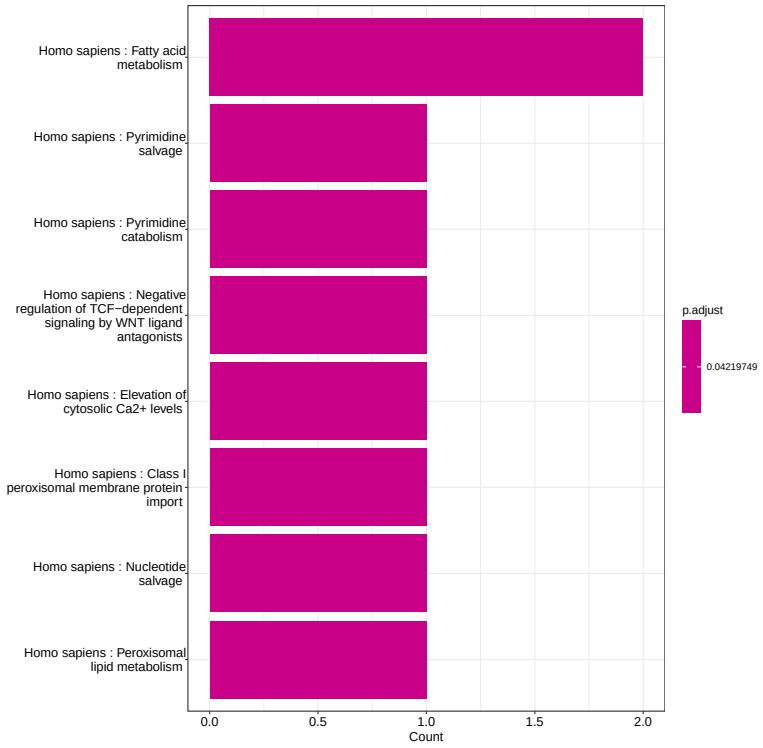

Figure 10: Bar plot of pathway enrichment results.

The function returns a list, one element is a dataframe containing pathways annotated by genes of one module.

Bar plot describes adjusted p-values and gene count as bar color and height respectively.

Furthermore, in this analysis we would like to investigate modules that are associated with transcription variance in cancer compared to normal samples. So we identify the modules significantly enriched by differentially expressed genes (DEGs)

```
data(DEG,package = "epihet")
data(background,package = "epihet")
module.annotation=epihet::moduleAnno(DEG$refseq,background$gene,
                                     module.gene=epi.network$module,
                                     cutoff=0.1,adjust.method = "fdr",
                                     prefix='epipoly',pdf.height = 5,
                                     pdf.width = 5, sve = TRUE)
```

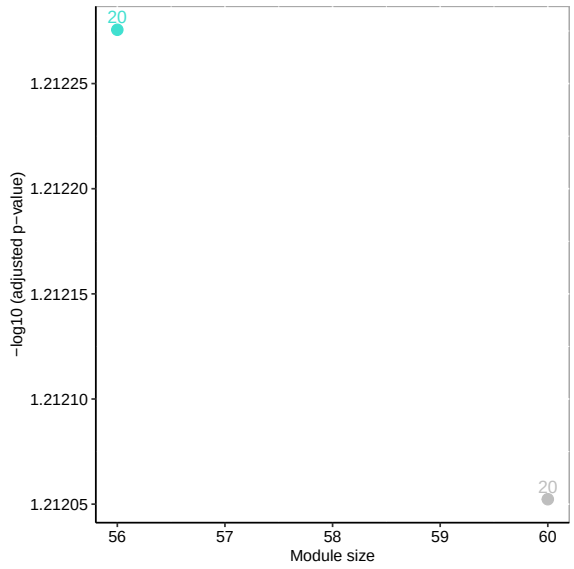

Figure 11: Scatter plot showing the modules enriched by DEGs. x-axis is the number of genes in the module. y-axis is the -log10(adjusted p-value). p-value is obtained from hypergeometric test. The label is the number of DEGs in the module.

6.4 Module comparison

Finally, we compare the similarity between modules using the Jaccard similarity score for different cancers or different subtypes of one cancer.

```
data(modulesil,package = "epihet")
data(moduledm,package = "epihet")
sim.score=epihet::moduleSim(module.subtype1=modulesil,
                            module.subtype2=moduledm,
```

```
                    pdf.height = 3,pdf.width = 3,
                    sve = TRUE)
```

Here, you just need to input TOM files of the two cancer types or subtypes you interested. This function returns a matrix showing the Jaccard similarity score between any two modules from different cancers or different subtypes of one cancer and a heatmap plot to visualization.

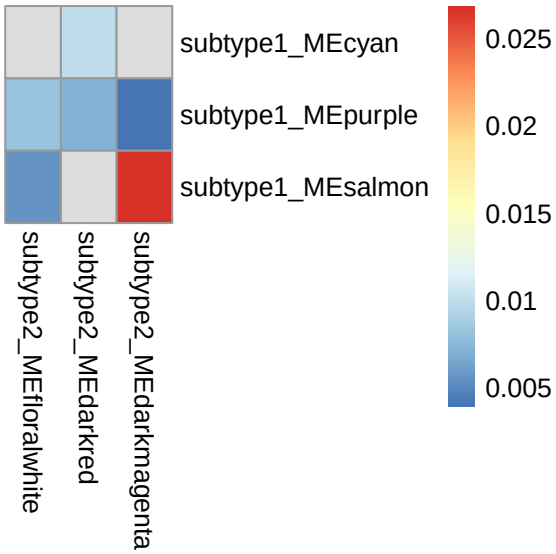

Figure 12: Heatmap plot showing the Jaccard score between any two modules from cancer types/subtypes you interested.

7 SessionInfo

```
  sessionInfo()
```

```
## R version 4.2.0 RC (2022-04-19 r82224)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dylib
##
## locale:
## [1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] org.Hs.eg.db_3.15.0   AnnotationDbi_1.58.0  Biobase_2.56.0
##  [4] clusterProfiler_4.4.0 doParallel_1.0.17     iterators_1.0.14
##  [7] foreach_1.5.2         GenomicRanges_1.48.0  GenomeInfoDb_1.32.0
## [10] IRanges_2.30.0        S4Vectors_0.34.0      BiocGenerics_0.42.0
## [13] ggfortify_0.4.14      ggplot2_3.3.5         knitr_1.38
##
## loaded via a namespace (and not attached):
##   [1] shadowtext_0.1.2       backports_1.4.1        Hmisc_4.7-0
##   [4] fastmatch_1.1-3        systemfonts_1.0.4      epihet_1.12.0
##   [7] plyr_1.8.7             igraph_1.3.1           lazyeval_0.2.2
##  [10] splines_4.2.0          BiocParallel_1.30.0    digest_0.6.29
##  [13] yulab.utils_0.0.4      htmltools_0.5.2        GOSemSim_2.22.0
##  [16] viridis_0.6.2          GO.db_3.15.0           fansi_1.0.3
##  [19] magrittr_2.0.3         checkmate_2.1.0        memoise_2.0.1
##  [22] cluster_2.1.3          fastcluster_1.2.3      Biostrings_2.64.0
##  [25] graphlayouts_0.8.0     matrixStats_0.62.0     enrichplot_1.16.0
```

```
##  [28] jpeg_0.1-9              colorspace_2.0-3        blob_1.2.3
##  [31] rappdirs_0.3.3          ggrepel_0.9.1           textshaping_0.3.6
##  [34] xfun_0.30               dplyr_1.0.8             crayon_1.5.1
##  [37] RCurl_1.98-1.6          jsonlite_1.8.0          graph_1.74.0
##  [40] scatterpie_0.1.7        impute_1.70.0           survival_3.3-1
##  [43] ape_5.6-2               glue_1.6.2              polyclip_1.10-0
##  [46] gtable_0.3.0            zlibbioc_1.42.0         XVector_0.36.0
##  [49] graphite_1.42.0         scales_1.2.0            DOSE_3.22.0
##  [52] pheatmap_1.0.12         DBI_1.1.2               Rcpp_1.0.8.3
##  [55] viridisLite_0.4.0       htmlTable_2.4.0         gridGraphics_0.5-1
##  [58] tidytree_0.3.9          foreign_0.8-82          bit_4.0.4
##  [61] reactome.db_1.79.0      preprocessCore_1.58.0   Formula_1.2-4
##  [64] htmlwidgets_1.5.4       httr_1.4.2              fgsea_1.22.0
##  [67] RColorBrewer_1.1-3      ellipsis_0.3.2          pkgconfig_2.0.3
##  [70] farver_2.1.0            nnet_7.3-17             utf8_1.2.2
##  [73] dynamicTreeCut_1.63-1   labeling_0.4.2          ggplotify_0.1.0
##  [76] tidyselect_1.1.2        rlang_1.0.2             reshape2_1.4.4
##  [79] munsell_0.5.0           tools_4.2.0             cachem_1.0.6
##  [82] downloader_0.4          cli_3.3.0               generics_0.1.2
##  [85] RSQLite_2.2.12          evaluate_0.15           stringr_1.4.0
##  [88] fastmap_1.1.0           ragg_1.2.2              ggtree_3.4.0
##  [91] bit64_4.0.5             tidygraph_1.2.1         purrr_0.3.4
##  [94] KEGGREST_1.36.0         ggraph_2.0.5            ReactomePA_1.40.0
##  [97] nlme_3.1-157            aplot_0.1.3             DO.db_2.9
## [100] rstudioapi_0.13         compiler_4.2.0          png_0.1-7
## [103] treeio_1.20.0           tibble_3.1.6            tweenr_1.0.2
## [106] stringi_1.7.6           highr_0.9               lattice_0.20-45
## [109] Matrix_1.4-1            vctrs_0.4.1             pillar_1.7.0
## [112] lifecycle_1.0.1         data.table_1.14.2       bitops_1.0-7
## [115] patchwork_1.1.1         qvalue_2.28.0           R6_2.5.1
## [118] latticeExtra_0.6-29     gridExtra_2.3           codetools_0.2-18
## [121] MASS_7.3-57             assertthat_0.2.1        withr_2.5.0
## [124] EntropyExplorer_1.1     GenomeInfoDbData_1.2.8  grid_4.2.0
## [127] rpart_4.1.16            ggfun_0.0.6             tidyr_1.2.0
## [130] Rtsne_0.16              ggforce_0.3.3           WGCNA_1.71
```

```
## [133] base64enc_0.1-3
```

8 References

1. H. Pagès, M. Lawrence and P. Aboyoun (2017). S4Vectors: S4 implementation of vectors and lists. R package version 0.12.2.

2. H. Wickham. ggplot2: Elegant Graphics for Data Analysis. SpringerVerlag New York, 2009.

3. Jesse H. Krijthe (2015). Rtsne: TDistributed Stochastic Neighbor Embedding using a BarnesHut Implementation, URL: https://github.com/jkrijthe/Rtsne

4. JJ Allaire, Joe Cheng, Yihui Xie, Jonathan McPherson, Winston Chang, Jeff Allen, Hadley Wickham, Aron Atkins, Rob Hyndman and Ruben Arslan (2017). rmarkdown: Dynamic Documents for R. R package version 1.4. https://CRAN.Rproject.org/package=rmarkdown

5. Kai Wang, Charles A. Phillips, Arnold M. Saxton and Michael A. Langston (2015). EntropyExplorer: Tools for Exploring Differential Shannon Entropy, Differential Coefficient of Variation and Differential Expression. R package version 1.1.
https://CRAN.Rproject.org/package=EntropyExplorer

6. Landan G, Cohen NM, et al. "Epigenetic Polymorphism and the Stochastic Formation of Differentially Methylated Regions in Normal and Cancerous Tissues." *Nature Genetics* **44** (2012): 12071214. 10.1038/ng.2442

7. Landau, Dan et al. "Locally Disordered Methylation forms the Basis of Intra tumor Methylome Variation in Chronic Lymphocytic Leukemia." *Cancer Cell* **26(6)** (2014): 813825. 10.1016/j.ccell.2014.10.012

8. Lawrence M, Huber W, Pagès H, Aboyoun P, Carlson M, et al. (2013) Software for Computing and Annotating Genomic Ranges. PLoS Comput Biol 9(8): e1003118.
doi:10.1371/journal.pcbi.1003118

9. Li, Sheng et al. "Distinct Evolution and Dynamics of Epigenetic and Genetic Heterogeneity in Acute Myeloid Leukemia." *Nature Medicine* **22.7** (2016): 792799. 10.1038/nm.4125

10. Li, Sheng et al. "Dynamic Evolution of Clonal Epialleles Revealed by Methclone." *Genome Biology* **15** (2014).

11. Masaaki Horikoshi and Yuan Tang (2016). ggfortify: Data Visualization Tools for Statistical Analysis Results. https://CRAN.Rproject.org/package=ggfortify

12. Matt Dowle and Arun Srinivasan (2017). data.table: Extension of 'data.frame'. R package version 1.10.4. https://CRAN.Rproject.org/package=data.table

13. Pan, Heng et al. “Epigenomic Evolution in Diffuse Large Bcell Lymphomas.” *Nature Communications* **6** (2015). 10.1038/ncomms7921

14. R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.Rproject.org/.

15. Raivo Kolde (2015). pheatmap: Pretty Heatmaps. R package version 1.0.8. https://CRAN.Rproject.org/package=pheatmap

16. Revolution Analytics and Steve Weston (2015). doMC: Foreach Parallel Adaptor for 'parallel'. R package version 1.3.4. https://CRAN.Rproject.org/package=doMC

17. Revolution Analytics and Steve Weston (2015). foreach: Provides Foreach Looping Construct for R. R package version 1.4.3. https://CRAN.Rproject.org/package=foreach

18. Sheffield et al. “DNA Methylation Heterogeneity Defines a Disease Spectrum in Ewing Sarcoma.” *Nature Medicine* **23** (2017): 386395. 10.1038/nm.4273

19. Sherwin, William. “Entropy and Information Approaches to Genetic Diversity and its Expression: Genomic Geography.” *Entropy* **12** (2010): 17651798. 10.3390/e12071765

20. Yihui Xie (2017). knitr: A GeneralPurpose Package for Dynamic Report Generation in R. R package version 1.16.

21. Yuan Tang, Masaaki Horikoshi, and Wenxuan Li. ggfortify: Unified Interface to Visualize Statistical Result of Popular R Packages. The R Journal, 2016.