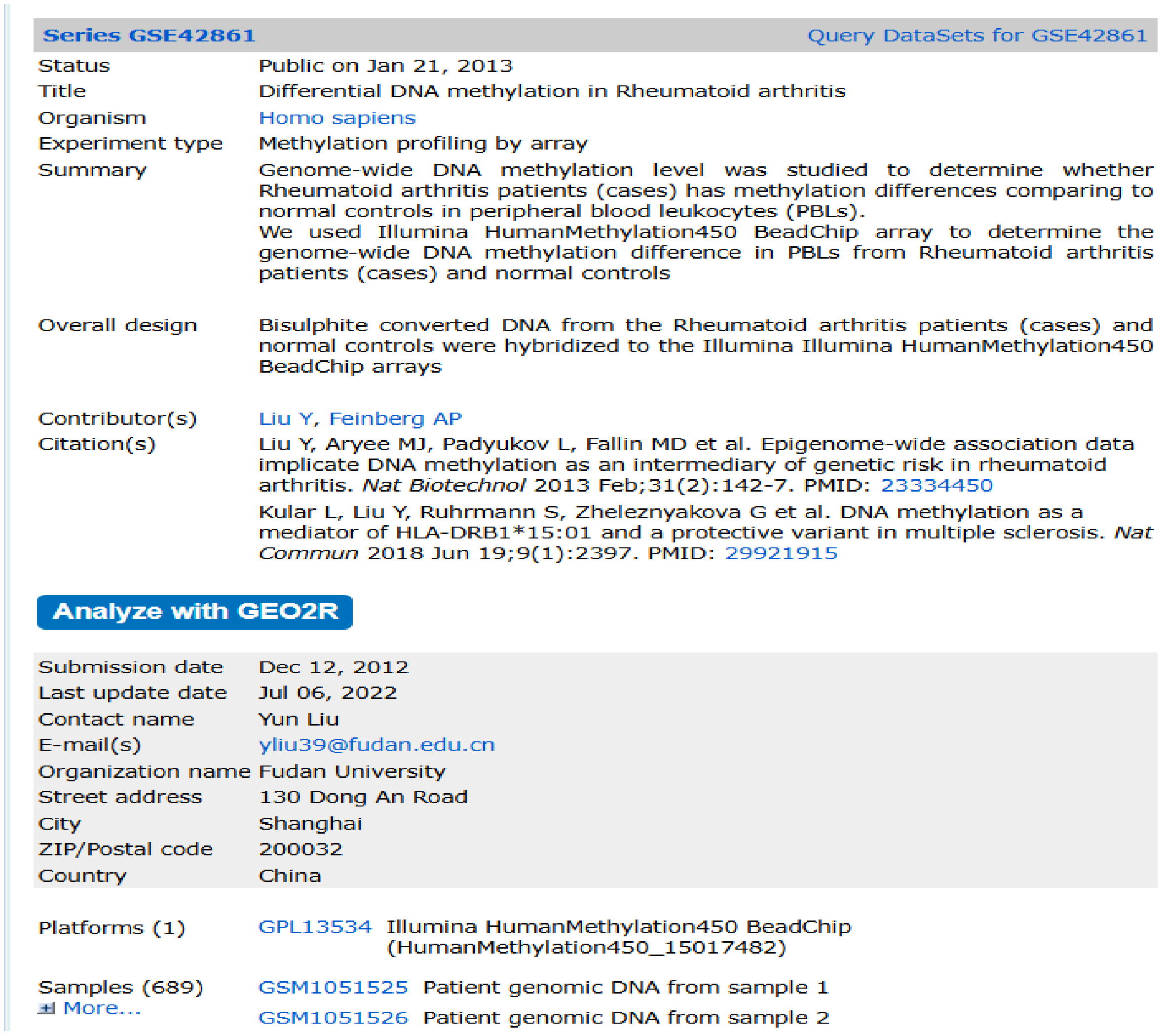
DNA Methylation Analysis (Array data)

Gayatri Samant

2025-11-18

I have selected the GEO dataset **GSE42861 (Illumina 450K array)** which contains **689** human blood samples **(RA patients vs controls).** For initial analysis, I will **download 6 samples** (3 RA patients + 3 controls) to ensure computational feasibility on my laptop. I have created the project folder on the **E: drive (E:\MethylationProject\GSE42861\raw_idats\)** and prepared the **metadata file SampleSheet.csv**. Next I will load these files into R and perform quality control.

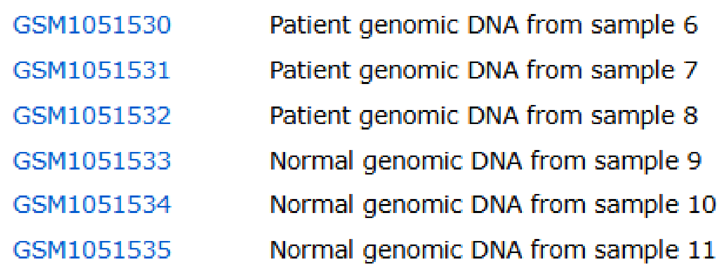
1) Downlaoding the Dataset **GSE42861** from GEO



Series GSE42861	Query DataSets for GSE42861
Status	Public on Jan 21, 2013
Title	Differential DNA methylation in Rheumatoid arthritis
Organism	Homo sapiens
Experiment type	Methylation profiling by array
Summary	Genome-wide DNA methylation level was studied to determine whether Rheumatoid arthritis patients (cases) has methylation differences comparing to normal controls in peripheral blood leukocytes (PBLs). We used Illumina HumanMethylation450 BeadChip array to determine the genome-wide DNA methylation difference in PBLs from Rheumatoid arthritis patients (cases) and normal controls
Overall design	Bisulphite converted DNA from the Rheumatoid arthritis patients (cases) and normal controls were hybridized to the Illumina Illumina HumanMethylation450 BeadChip arrays
Contributor(s)	Liu Y, Feinberg AP
Citation(s)	Liu Y, Aryee MJ, Padyukov L, Fallin MD et al. Epigenome-wide association data implicate DNA methylation as an intermediary of genetic risk in rheumatoid arthritis. *Nat Biotechnol* 2013 Feb;31(2):142-7. PMID: 23334450 Kular L, Liu Y, Ruhrmann S, Zheleznyakova G et al. DNA methylation as a mediator of HLA-DRB1*15:01 and a protective variant in multiple sclerosis. *Nat Commun* 2018 Jun 19;9(1):2397. PMID: 29921915

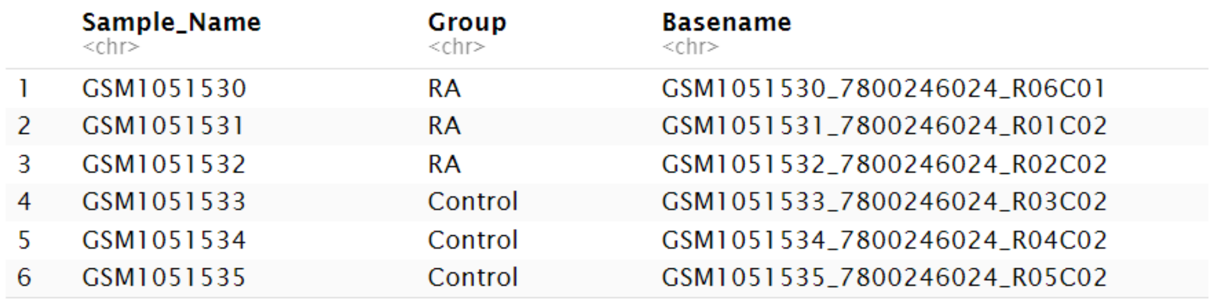
Submission date	Dec 12, 2012
Last update date	Jul 06, 2022
Contact name	Yun Liu
E-mail(s)	yliu39@fudan.edu.cn
Organization name	Fudan University
Street address	130 Dong An Road
City	Shanghai
ZIP/Postal code	200032
Country	China
Platforms (1)	GPL13534 Illumina HumanMethylation450 BeadChip (HumanMethylation450_15017482)
Samples (689) More...	GSM1051525 Patient genomic DNA from sample 1 GSM1051526 Patient genomic DNA from sample 2

Selected samples are :



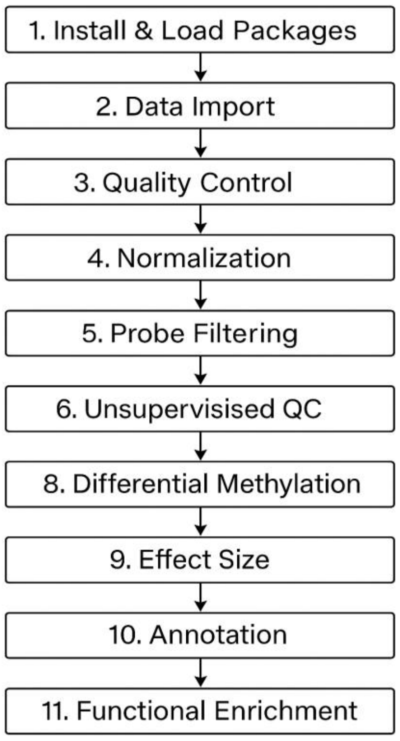
GSM1051530	Patient genomic DNA from sample 6
GSM1051531	Patient genomic DNA from sample 7
GSM1051532	Patient genomic DNA from sample 8
GSM1051533	Normal genomic DNA from sample 9
GSM1051534	Normal genomic DNA from sample 10
GSM1051535	Normal genomic DNA from sample 11

2) Created a metadata file of the selected samples and named it as **SampleSheet.csv**



	Sample_Name <chr>	Group <chr>	Basename <chr>
1	GSM1051530	RA	GSM1051530_7800246024_R06C01
2	GSM1051531	RA	GSM1051531_7800246024_R01C02
3	GSM1051532	RA	GSM1051532_7800246024_R02C02
4	GSM1051533	Control	GSM1051533_7800246024_R03C02
5	GSM1051534	Control	GSM1051534_7800246024_R04C02
6	GSM1051535	Control	GSM1051535_7800246024_R05C02

3) Started working in an R markdown file using the following pipeline.



This document outlines a full Illumina 450k methylation array workflow in **R**, including QC, normalization, differential methylation analysis, annotation, and enrichment

Installing all required packages (R 4.3.3 + Bioc 3.17)

Purpose: Install Bioconductor and CRAN packages for methylation analysis.

```
## 1. Install BiocManager if missing
# install.packages("BiocManager")
# BiocManager::install(version = "3.17", ask = FALSE)

R.version.string

## [1] "R version 4.3.3 (2024-02-29 ucrt)"

# Make sure BiocManager is set to 3.17
BiocManager::version()

## [1] '3.17'
```

Install Bioconductor packages

```
# BiocManager::install(c(
#   "minfi",
#   "limma",
#   "GenomicRanges",
#   "GenomicFeatures",
#   "rGREAT"
# ))
```

Install CRAN packages

```
# install.packages(c(
#   "tidyverse",
#   "R.utils"
# ))
```

Manual install for 450k annotation packages

1. IlluminaHumanMethylation450kmanifest

2. IlluminaHumanMethylation450kanno.ilmn12.hg19

These two Bioconductor packages are older and not yet updated . Because of that, BiocManager::install() failed to fetch them normally. So, we installed them manually (from Bioconductor source URLs / using remotes) to ensure, compatibility and to provide the 450k array probe manifest and annotation data needed for methylation analysis.

```
# #---450k array probe manifest---
#
# if (!requireNamespace("remotes", quietly = TRUE))
# install.packages("remotes")
#
# # Install from Bioconductor release tarball
#
# remotes::install_url("https://bioconductor.org/packages/release/data/annota
tion/src/contrib/IlluminaHumanMethylation450kmanifest_0.4.0.tar.gz")

# # --- Install 450k Annotation Package ---
# if (!requireNamespace("remotes", quietly = TRUE))
#    install.packages("remotes")
#
# remotes::install_url(
#    "https://bioconductor.org/packages/release/data/annotation/src/contrib/I
lluminaHumanMethylation450kanno.ilmn12.hg19_0.6.0.tar.gz"
#  )
#
# options(download.file.method = "libcurl")
# BiocManager::install("IlluminaHumanMethylation450kanno.ilmn12.hg19")

#BiocManager::install("DMRcate")

# # Dependency package for DMRcate, if this not installed the package will no
t load


# BiocManager::install('IlluminaHumanMethylationEPICanno.ilm10b4.hg19')

# # Dependency packages for rGREAT, if this not installed the package will no
t load
# BiocManager::install('TxDb.Hsapiens.UCSC.hg19.knownGene')
# BiocManager::install('TxDb.Hsapiens.UCSC.hg38.knownGene')
```

Loading the required packages

```
# These packages support data import, normalization, QC, modeling, DMR callin
g, and enrichment analysis.

library(minfi)              # Core Illumina methylation processing
library(limma)              # Differential methylation modeling
```

```r
library(GenomicRanges)      # For genomic region manipulation
library(GenomicFeatures)    # Annotation utilities
library(IlluminaHumanMethylation450kanno.ilmn12.hg19)  # 450k annotation
library(DMRcate)            # Region-level differential methylation
library(rGREAT)             # Functional enrichment for genomic regions
library(tidyverse)          # Data wrangling + visualization
library(R.utils)            # For unzipping compressed IDAT files
```

1. Project Paths

These paths point to raw IDAT files and sample metadata. We have to change these to match our local directory structure.

```r
base_dir <- "E:/MethylationProject/GSE42861"
idat_dir <- file.path(base_dir, "raw_idats")
sample_sheet_path <- file.path(idat_dir, "SampleSheet.csv")
```

```r
list.files(idat_dir)[1:15]
```

```
##  [1] "DNA_Methyl_Array.html"
##  [2] "DNA_Methyl_Array.Rmd"
##  [3] "GSM1051530_7800246024_R06C01_Grn.idat"
##  [4] "GSM1051530_7800246024_R06C01_Red.idat"
##  [5] "GSM1051531_7800246024_R01C02_Grn.idat"
##  [6] "GSM1051531_7800246024_R01C02_Red.idat"
##  [7] "GSM1051532_7800246024_R02C02_Grn.idat"
##  [8] "GSM1051532_7800246024_R02C02_Red.idat"
##  [9] "GSM1051533_7800246024_R03C02_Grn.idat"
## [10] "GSM1051533_7800246024_R03C02_Red.idat"
## [11] "GSM1051534_7800246024_R04C02_Grn.idat"
## [12] "GSM1051534_7800246024_R04C02_Red.idat"
## [13] "GSM1051535_7800246024_R05C02_Grn.idat"
## [14] "GSM1051535_7800246024_R05C02_Red.idat"
## [15] "SampleSheet.csv"
```

2. Load Raw IDAT Files

Illumina IDAT files store raw green/red channel intensities. minfi uses both intensities + control probes to compute detection p-values, perform background correction, and extract Beta/M-values.

```
# Unzip .idat.gz files if needed
gz_files <- list.files(
  idat_dir,
  pattern = "\\.idat\\.gz$",
  full.names = TRUE
)

if (length(gz_files) > 0) {
  message("Unzipping compressed IDAT files...")

  sapply(gz_files, function(f) {

    # output filename: remove .gz
    out_f <- sub("\\.gz$", "", f)

    # unzip only if the decompressed file does NOT already exist
    if (!file.exists(out_f)) {
      R.utils::gunzip(f, remove = FALSE, overwrite = FALSE)
    }

  })
}

# Read sample sheet
sample_sheet <- read.csv(sample_sheet_path, stringsAsFactors = FALSE)
head(sample_sheet)

##   Sample_Name   Group                     Basename
## 1  GSM1051530      RA GSM1051530_7800246024_R06C01
## 2  GSM1051531      RA GSM1051531_7800246024_R01C02
## 3  GSM1051532      RA GSM1051532_7800246024_R02C02
## 4  GSM1051533 Control GSM1051533_7800246024_R03C02
## 5  GSM1051534 Control GSM1051534_7800246024_R04C02
## 6  GSM1051535 Control GSM1051535_7800246024_R05C02

# Make Basename point to FULL PATH of IDAT files
sample_sheet$Basename <- file.path(idat_dir, sample_sheet$Basename)

# Verify Basename is correct
head(sample_sheet$Basename)

## [1] "E:/MethylationProject/GSE42861/raw_idats/GSM1051530_7800246024_R06C01
"
## [2] "E:/MethylationProject/GSE42861/raw_idats/GSM1051531_7800246024_R01C02
"
## [3] "E:/MethylationProject/GSE42861/raw_idats/GSM1051532_7800246024_R02C02
"
## [4] "E:/MethylationProject/GSE42861/raw_idats/GSM1051533_7800246024_R03C02
"
## [5] "E:/MethylationProject/GSE42861/raw_idats/GSM1051534_7800246024_R04C02
```

```
"
## [6] "E:/MethylationProject/GSE42861/raw_idats/GSM1051535_7800246024_R05C02
"

# Check IDAT files present in the directory
list.files(idat_dir, pattern = "idat$", full.names = FALSE)

##  [1] "GSM1051530_7800246024_R06C01_Grn.idat"
##  [2] "GSM1051530_7800246024_R06C01_Red.idat"
##  [3] "GSM1051531_7800246024_R01C02_Grn.idat"
##  [4] "GSM1051531_7800246024_R01C02_Red.idat"
##  [5] "GSM1051532_7800246024_R02C02_Grn.idat"
##  [6] "GSM1051532_7800246024_R02C02_Red.idat"
##  [7] "GSM1051533_7800246024_R03C02_Grn.idat"
##  [8] "GSM1051533_7800246024_R03C02_Red.idat"
##  [9] "GSM1051534_7800246024_R04C02_Grn.idat"
## [10] "GSM1051534_7800246024_R04C02_Red.idat"
## [11] "GSM1051535_7800246024_R05C02_Grn.idat"
## [12] "GSM1051535_7800246024_R05C02_Red.idat"




# Load RGChannelSet (raw intensity object)
rgSet <- read.metharray.exp(targets = sample_sheet, extended = TRUE)
rgSet

## class: RGChannelSetExtended
## dim: 622399 6
## metadata(0):
## assays(5): Green Red GreenSD RedSD NBeads
## rownames(622399): 10600313 10600322 ... 74810490 74810492
## rowData names(0):
## colnames(6): GSM1051530_7800246024_R06C01 GSM1051531_7800246024_R01C02
##   ... GSM1051534_7800246024_R04C02 GSM1051535_7800246024_R05C02
## colData names(4): Sample_Name Group Basename filenames
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19
```

Here we have Checked the dimensions (e.g., 622399 probes × 6 samples) and annotation (IlluminaHumanMethylation450k). Confirming data integrity.

3. Quality Control: Detection P-values

Detection p-value = how confidently a probe is detected above background.

We remove:

- Samples with >5% failed probes

- Probes failing in >5% samples

This step removes poor-quality signals and increases reliability.

```
detP <- detectionP(rgSet)

## Loading required package: IlluminaHumanMethylation450kmanifest

summary(as.vector(detP))

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000000 0.0002224 0.0000000 0.9999997



# Filter samples
failed_samples <- colMeans(detP > 0.01) > 0.05
sample_sheet <- sample_sheet[!failed_samples, ]
rgSet <- rgSet[, !failed_samples]

# Filter probes
failed_probes <- rowMeans(detP > 0.01) > 0.05
rgSet <- rgSet[!failed_probes, ]

head(colMeans(detP > 0.01))

## GSM1051530_7800246024_R06C01 GSM1051531_7800246024_R01C02
##                 0.0014273592                 0.0006796948
## GSM1051532_7800246024_R02C02 GSM1051533_7800246024_R03C02
##                 0.0008156338                 0.0001812520
## GSM1051534_7800246024_R04C02 GSM1051535_7800246024_R05C02
##                 0.0001441777                 0.0002121472
```

NOTE:

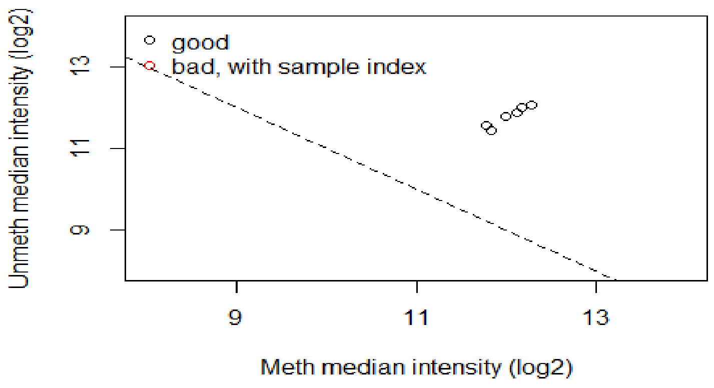
The detection p-values showed excellent probe-level quality, with a median p-value of 0 and fewer than 0.2% failed probes in every sample. Since none of the samples exceeded the standard 5% failure threshold, all six samples passed QC and were retained for downstream methylation analysis.

4. Normalization (NOOB)

preprocessNoob performs: Background correction + dye-bias normalization.

Recommended for 450k datasets with <50 samples.

```r
mSet.noob <- preprocessNoob(rgSet)

# Basic visual QC (optional)
qc <- getQC(mSet.noob)
plotQC(qc)
```

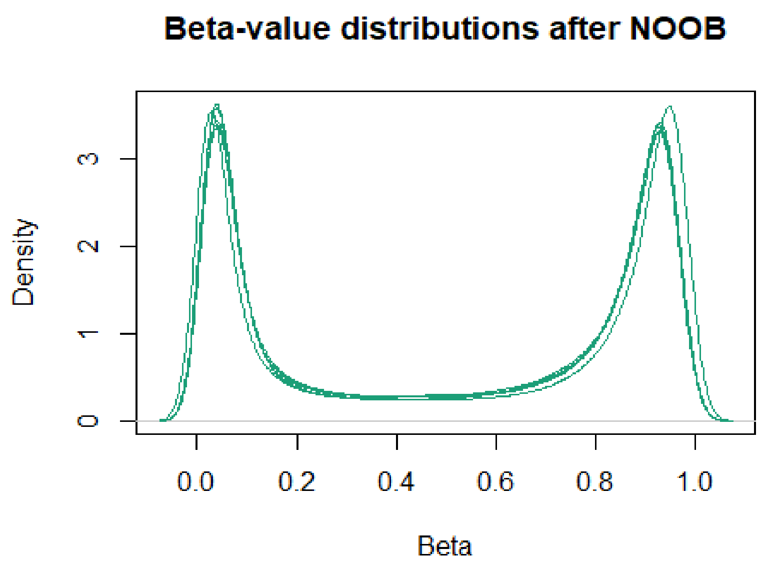


plotQC(qc) shows sample-wise median intensities; outliers indicate technical issues.

Here, None of them were tagged as poor, and considered for further analysis.

```r
# Extract methylation metrics

beta <- getBeta(mSet.noob)    # 0-1 methylation proportion
mval <- getM(mSet.noob)       # logit(beta), used for statistics

dim(beta)

## [1] 484004      6
```

```r
densityPlot(beta, main = "Beta-value distributions after NOOB", legend = FALS
E)
```



Density Plot: densityPlot(beta) should show overlapping curves for all samples .

So here we can see the result of good normalization.

5. Additional Probe Filtering

remove: - Probes overlapping SNPs (dropLociWithSnps)

(Optional) Sex-chromosome probes unless sex is a variable.

These reduce technical variability & false positives.

This step is used to clean our data — removing probes that could give **misleading signals** due to genetics (SNPs) or biological sex differences. This helps you focus on **real methylation differences** that matter for our study.

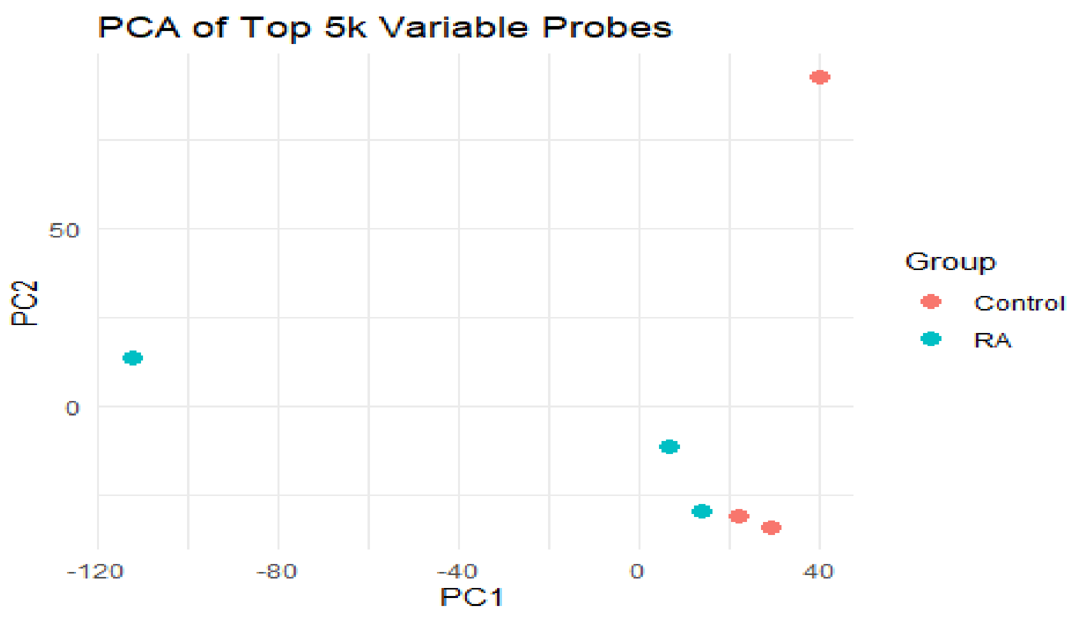
```r
# --- 1. NOOB normalized MethylSet ---
mSet.noob <- preprocessNoob(rgSet)

# --- 2. Convert to GenomicRatioSet (adds chr + position) ---
# ratioConvert() may return RatioSet → not acceptable for SNP filtering
# mapToGenome() ALWAYS returns a GenomicRatioSet → required for dropLociWithS
```

```
nps()
grSet <- mapToGenome(mSet.noob)

# --- 3. Remove CpGs affected by SNPs ---
mSet.clean <- dropLociWithSnps(grSet)

# --- 4. Extract final Beta and M-values ---
beta <- getBeta(mSet.clean)
mval <- getM(mSet.clean)

# --- 5. Remove chrX / chrY probes (optional if not modeling sex) ---
anno <- getAnnotation(IlluminaHumanMethylation450kanno.ilmn12.hg19)

# Match annotation to current probes
anno <- anno[rownames(beta), ]

sex_probes <- anno$chr %in% c("chrX", "chrY")

beta <- beta[!sex_probes, ]
mval <- mval[!sex_probes, ]
```

6. Unsupervised QC: PCA

Here we are doing PCA on top 5,000 variable CpGs.

```
topVar <- order(rowVars(mval), decreasing = TRUE)[1:5000]
pca <- prcomp(t(mval[topVar, ]))

pca_df <- data.frame(
PC1 = pca$x[,1],
PC2 = pca$x[,2],
Group = sample_sheet$Group
)

ggplot(pca_df, aes(PC1, PC2, color = Group)) +
geom_point(size = 3) + theme_minimal() +
labs(title = "PCA of Top 5k Variable Probes")
```



PCA of the top 5,000 most variable CpG sites showed modest separation between RA and Control groups, which is expected given the small sample size (n=3 per group). Two samples appeared as mild outliers along PC1, likely reflecting biological variability or differences in underlying blood cell composition rather than technical artifacts. Overall, the data show no evidence of batch effects and are suitable for downstream differential methylation analysis.

7. Design Matrix (Model Specification)

Creates model for RA vs Control.

Control = reference; RA = comparison. Ensures correct contrast for limma.

```
sample_sheet$Group <- factor(sample_sheet$Group, levels = c("Control","RA"))
design <- model.matrix(~ Group, data = sample_sheet)
design

##   (Intercept) GroupRA
## 1           1       1
## 2           1       1
## 3           1       1
## 4           1       0
## 5           1       0
## 6           1       0
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$Group
## [1] "contr.treatment"
```

The design matrix coded Control as the reference level and RA as the comparison group, resulting in a binary indicator column (GroupRA) that correctly represents the contrast RA vs Control. This ensures limma estimates methylation differences as RA - Control.

8. Differential Methylation Using limma

limma fits a linear model at each CpG site. **M-values are used for better statistical properties.**

```r
fit <- lmFit(mval, design)
fit <- eBayes(fit)
results <- topTable(fit, coef = "GroupRA", number = Inf)
results
```

Fits linear model on M-values; computes p-values and FDR.

`topTable()` gives CpGs ranked by significance.

9. Extract Probe-Level Results + ΔBeta

ΔBeta gives biological effect size (difference in methylation %).

Computes biological effect size (RA – Control).

Significant CpGs = FDR < 0.05 & |ΔBeta| ≥ 0.1.

Result: ~6,489 CpGs (probe-level) significant.

```r
# Get all CpGs
topCpGs <- topTable(fit, coef = 1, number = Inf)

# Calculate deltaBeta
mean_RA <- rowMeans(beta[, sample_sheet$Group == "RA"])
mean_Control <- rowMeans(beta[, sample_sheet$Group == "Control"])
topCpGs$deltaBeta <- mean_RA - mean_Control

# Standard cutoff,for larger sample size with FDR
# Here Significant probes or CpGs are filtered based on FDR of 5% and Biologi
cal effect
```

```r
sig_probes <- subset(topCpGs, adj.P.Val < 0.05 & abs(deltaBeta) >= 0.1)
sig_probes
```

10. Annotate Significant Probes

Adds gene symbols, genomic coordinates, and CpG context.

Enables biological interpretation (e.g., RA-associated genes).

```r
library(IlluminaHumanMethylation450kanno.ilmn12.hg19)

anno <- getAnnotation(IlluminaHumanMethylation450kanno.ilmn12.hg19)
sig_probes$CpG <- rownames(sig_probes)

annot_sig <- merge(sig_probes, anno, by.x = "CpG", by.y = "Name", all.x = TRU
E)

head(annot_sig)

## DataFrame with 6 rows and 40 columns
##             CpG     logFC   AveExpr         t     P.Value   adj.P.Val
B
##     <character> <numeric> <numeric> <numeric>   <numeric>   <numeric> <numer
ic>
## 1   cg00007426   4.05244   4.21035   20.8245 2.62576e-08 6.33858e-08   9.99
403
## 2   cg00015373   3.83035   3.75085   31.4033 9.87464e-10 9.38409e-09  13.16
656
## 3   cg00017059   3.93033   3.82075   31.7810 8.97190e-10 9.11508e-09  13.25
345
## 4   cg00017203   3.21838   3.31425   15.1371 3.25578e-07 5.30075e-07   7.37
610
## 5   cg00018866   3.63048   3.55748   24.1313 8.12171e-09 2.67660e-08  11.16
779
## 6   cg00020446   2.91599   2.94845   21.7139 1.88308e-08 4.90191e-08  10.33
022
##   deltaBeta         chr       pos      strand    AddressA    AddressB
##   <numeric> <character> <integer> <character> <character> <character>
## 1  0.100338        chr2 240404782           -    11713347
## 2  0.109087        chr9 134247288           +    47621496
## 3  0.129595        chr2  45797255           -    52659380
## 4  0.161070        chr8 145000033           +    70745331
## 5  0.110412       chr10  50343286           +    34633343
## 6  0.147683       chr12 124585444           +    42621387
##                 ProbeSeqA   ProbeSeqB        Type    NextBase       Color
##               <character> <character> <character> <character> <character>
## 1 AAAAAAAAACRCTAACCAAA..                      II
## 2 TAAATAACTTAAACAACATT..                      II
## 3 TCATTCATTATACATTATCT..                      II
## 4 ACTTCTCTTCTTCTCTCTAC..                      II
## 5 TAAAAATACCAAACAAACTC..                      II
## 6 AACTCTATTTTCACTCTATT..                      II
##      Probe_rs Probe_maf      CpG_rs   CpG_maf      SBE_rs   SBE_maf
##   <character> <numeric> <character> <numeric> <character> <numeric>
## 1  rs73003371  0.218721          NA        NA          NA        NA
## 2          NA        NA          NA        NA          NA        NA
## 3          NA        NA          NA        NA          NA        NA
## 4   rs7002152  0.237347          NA        NA          NA        NA
## 5  rs73304705  0.063071          NA        NA          NA        NA
## 6          NA        NA          NA        NA          NA        NA
##             Islands_Name Relation_to_Island       Forward_Sequence
##              <character>        <character>            <character>
## 1                                   OpenSea GGAAGGAGGAAGGGAAGGAG..
## 2 chr9:134248688-13424..            N_Shore ATTCGCCCGCCTTGGCCTCC..
## 3                                   OpenSea CACAACCATGCTCATTCATT..
## 4 chr8:144990270-14500..             Island AGCTGCCGCTGCTTCTCCAG..
## 5 chr10:50339890-50340..            S_Shelf GCTCTTCCTTGTGCCTGTTC..
## 6                                   OpenSea ACGGCGGCAATCAGCAGTGG..
##                SourceSeq Random_Loci Methyl27_Loci      UCSC_RefGene_Name
##              <character> <character>   <character>            <character>
## 1 GGAAGGAGCGCTGGCCAGGC..
## 2 AGGTGACTTGAACAGCATTC..
## 3 CGCAACTGCTGTGTAACACA..                                            SRBD1
## 4 CTTCTCTTCTTCTCTCTGCT..                           PLEC1;PLEC1;PLEC1;PL..
## 5 CGGTCTCCTCTGAAACATCA..                                          FAM170B
## 6 CGGCGGAGAAGACTGTGCAG..
##   UCSC_RefGene_Accession     UCSC_RefGene_Group     Phantom         DMR
##              <character>            <character> <character> <character>
## 1
## 2
## 3              NM_018079                   Body
## 4 NM_201378;NM_201384;.. Body;Body;Body;Body;..
## 5           NM_001164484                TSS1500
## 6
##      Enhancer             HMM_Island Regulatory_Feature_Name
##   <character>            <character>             <character>
## 1              2:240069679-240070354
## 2
## 3        TRUE
## 4              8:145062275-145072638
## 5
## 6             12:123151105-123151476
##   Regulatory_Feature_Group         DHS
##                <character> <character>
## 1
## 2
## 3
## 4
## 5
## 6

view(annot_sig)
```

Now annot_sig contains all CpGs with gene annotation, island info, etc

10. Create GRanges for GREAT enrichment (from pre-filtered significant CpGs)

This code prepares our significant CpGs for GREAT enrichment analysis by converting them into a **GRanges object**, which organizes each CpG as a genomic region with its chromosome, position, effect size (deltaBeta), significance (adj.P.Val), and probe ID.

This format is required by GREAT to identify biological functions or pathways associated with these regions. The GRanges object can also be converted to a data frame and saved as a CSV for easy input to the tool.

```
library(GenomicRanges)

# annot_sig: our annotated significant CpGs
# Ensure columns exist: "chr", "pos", "deltaBeta", "adj.P.Val", "CpG"

great_gr <- GRanges(
  seqnames = annot_sig$chr,
  ranges = IRanges(start = annot_sig$pos, end = annot_sig$pos),
  deltaBeta = annot_sig$deltaBeta,      # optional: effect size
  P.Value = annot_sig$adj.P.Val,        # optional: FDR
  CpG = annot_sig$CpG                    # probe IDs
)

# Inspect first few ranges
great_gr

## GRanges object with 6489 ranges and 3 metadata columns:
##          seqnames    ranges strand | deltaBeta     P.Value             CpG
##             <Rle> <IRanges>  <Rle> | <numeric>   <numeric>     <character>
##      [1]     chr2 240404782      * |  0.100338 6.33858e-08     cg00007426
##      [2]     chr9 134247288      * |  0.109087 9.38409e-09     cg00015373
##      [3]     chr2  45797255      * |  0.129595 9.11508e-09     cg00017059
##      [4]     chr8 145000033      * |  0.161070 5.30075e-07     cg00017203
##      [5]    chr10  50343286      * |  0.110412 2.67660e-08     cg00018866
##      ...      ...       ...    ... .       ...         ...            ...
##   [6485]     chr8 103238160      * | -0.117207 2.00726e-06  ch.8.2085179F
##   [6486]     chr8  25284195      * |  0.144019 1.08704e-08 ch.8.25340112R
##   [6487]     chr8  91678943      * |  0.135797 2.21241e-08 ch.8.91748119F
##   [6488]     chr9  88939147      * |  0.160146 2.29656e-08  ch.9.1169222R
##   [6489]     chr9  38731634      * |  0.175067 4.42044e-08 ch.9.38721634R
##   -------
##   seqinfo: 22 sequences from an unspecified genome; no seqlengths

# Convert to data.frame if you want CSV for GREAT
great_table <- as.data.frame(great_gr)
# write.csv(great_table, "GREAT_input_CpGs.csv", row.names = FALSE)
```

12. Build Regions for GREAT

This code prepares genomic regions for GREAT analysis. It creates **GRanges** for either **DMRs** (differentially methylated regions) or, if DMRs aren't available, uses a **±250 bp window around each significant CpG**.

It also defines **background regions** (all tested probes) in the same ±250 bp format. These foreground and background regions are needed for GREAT to assess **enrichment of biological functions or pathways**.

```
# BiocManager::install("ChIPseeker")

library(GenomicRanges)
library(dplyr)
library(ggplot2)
library(ChIPseeker) # for GREAT submission functions

# Note: make sure rGREAT is installed and loaded for submitGreatJob()

# If DMRs exist, expand them; otherwise use +/-250 bp around significant CpGs
if (exists("dmr_ranges") && length(dmr_ranges) > 0) {
  gr_for_great <- resize(dmr_ranges, width = pmax(width(dmr_ranges), 500), fi
x = "center")
} else {
  gr_for_great <- GRanges(
    seqnames = annot_sig$chr,
    ranges = IRanges(start = annot_sig$pos - 250, end = annot_sig$pos + 250),
    deltaBeta = annot_sig$deltaBeta,
    P.Value = annot_sig$adj.P.Val,
    CpG = annot_sig$CpG
  )
}

# Background regions (all tested probes)
tested_probes <- rownames(topCpGs)
anno_tested <- anno[tested_probes, ]
bg_gr <- GRanges(
  seqnames = anno_tested$chr,
  ranges = IRanges(start = anno_tested$pos - 250, end = anno_tested$pos + 250
)
)
```

13. GREAT Functional Enrichment

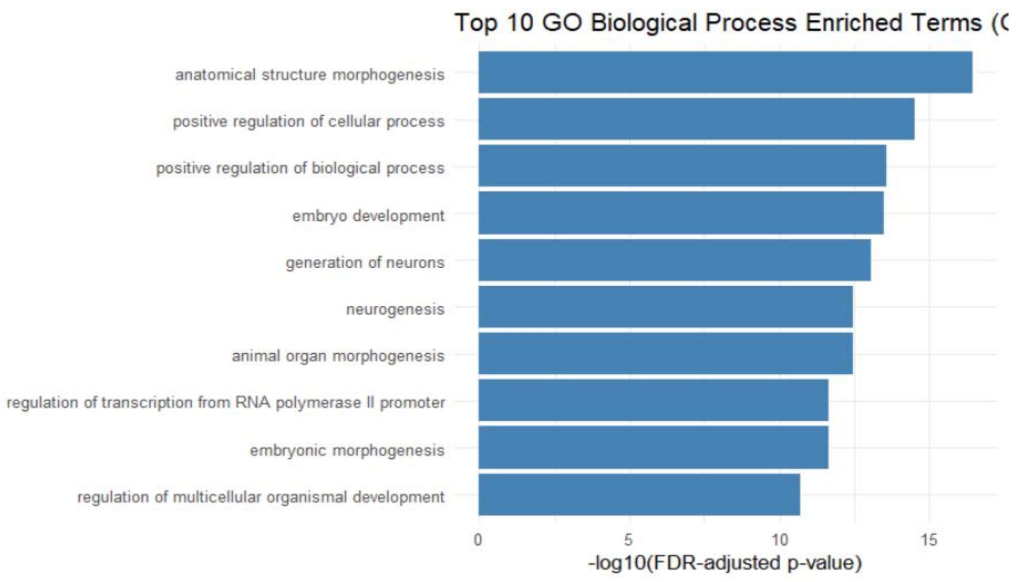
This chunk performs functional enrichment analysis using **GREAT (Genomic Regions Enrichment of Annotations Tool)** via the **rGREAT** package. It submits our genomic regions or CpGs (gr_for_great) to GREAT, which links each region to nearby genes and tests for enrichment in biological pathways and annotations. The results are returned as tables for the three **Gene Ontology (GO)** categories: **Biological Process (BP)**, **Molecular Function (MF)**, and **Cellular Component (CC)**, showing which functions or processes are statistically overrepresented in your regions.

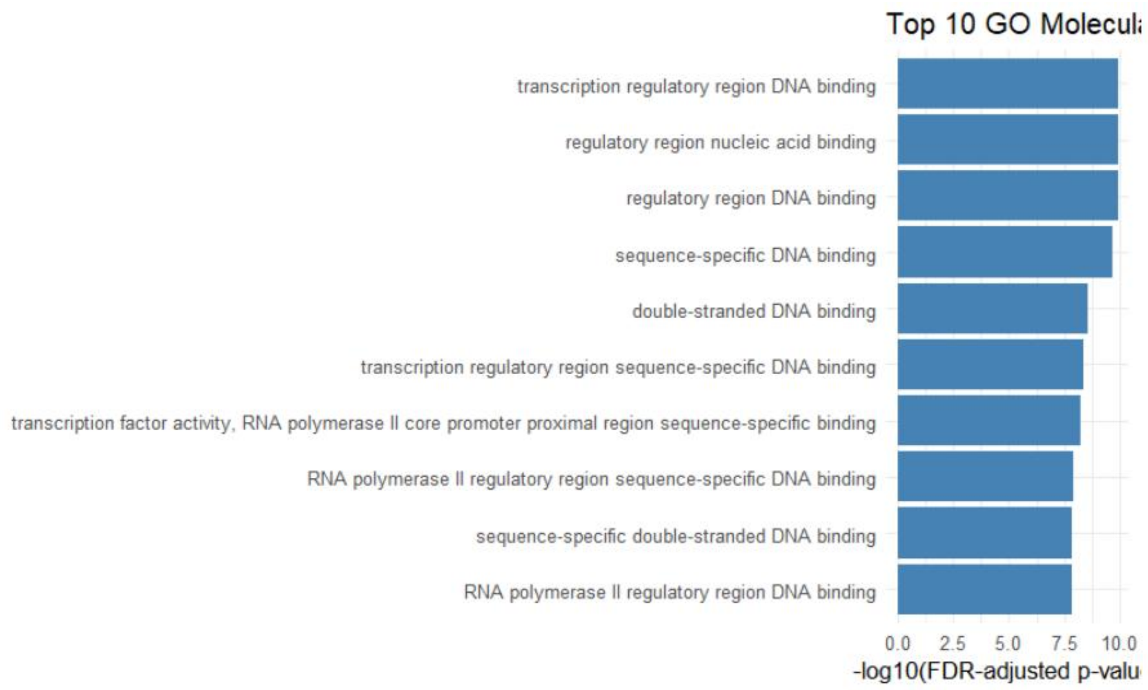
```
library(rGREAT)


# gr_for_great: your GRanges object of regions or CpGs
job <- submitGreatJob(gr_for_great, species = "hg19")


go_tables <- getEnrichmentTables(job)
names(go_tables)

## [1] "GO Molecular Function" "GO Biological Process" "GO Cellular Component
"
```

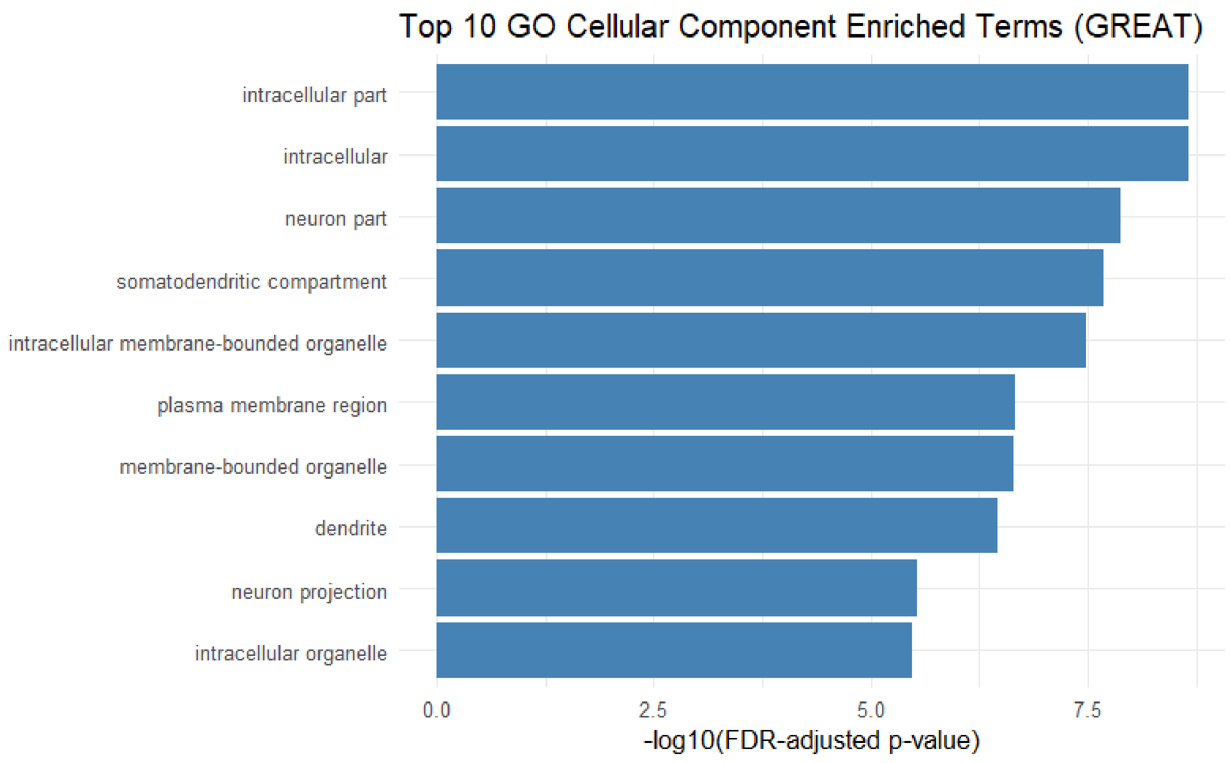
14. Visualize GO Enrichment (BP, MF, CC)

This chunk takes the enrichment results from GREAT and creates visualizations for each GO category (BP, MF, CC). For each category, it sorts the GO terms by their **FDR-adjusted p-values**, selects the **top 10 most significant terms**, and plots a **horizontal bar chart** using ggplot2 with -log10(FDR) as the y-axis. This makes it easy to quickly see the most enriched biological processes, molecular functions, and cellular locations associated with your significant CpGs or regions.

```r
library(dplyr)
library(ggplot2)

# List of GO categories to visualize
go_categories <- c("GO Biological Process", "GO Molecular Function", "GO Cell
ular Component")

for (cat in go_categories) {
  if (cat %in% names(go_tables)) {
    go_df <- go_tables[[cat]] %>% arrange(Hyper_Adjp_BH)

    top10 <- head(go_df, 10)

    p <- ggplot(top10, aes(x = reorder(name, -Hyper_Adjp_BH), y = -log10(Hype
r_Adjp_BH))) +
      geom_col(fill = "steelblue") +
      coord_flip() +
      theme_minimal() +
      labs(title = paste0("Top 10 ", cat, " Enriched Terms (GREAT)"),
           x = NULL,
           y = "-log10(FDR-adjusted p-value)")

    print(p)
  }
}
```



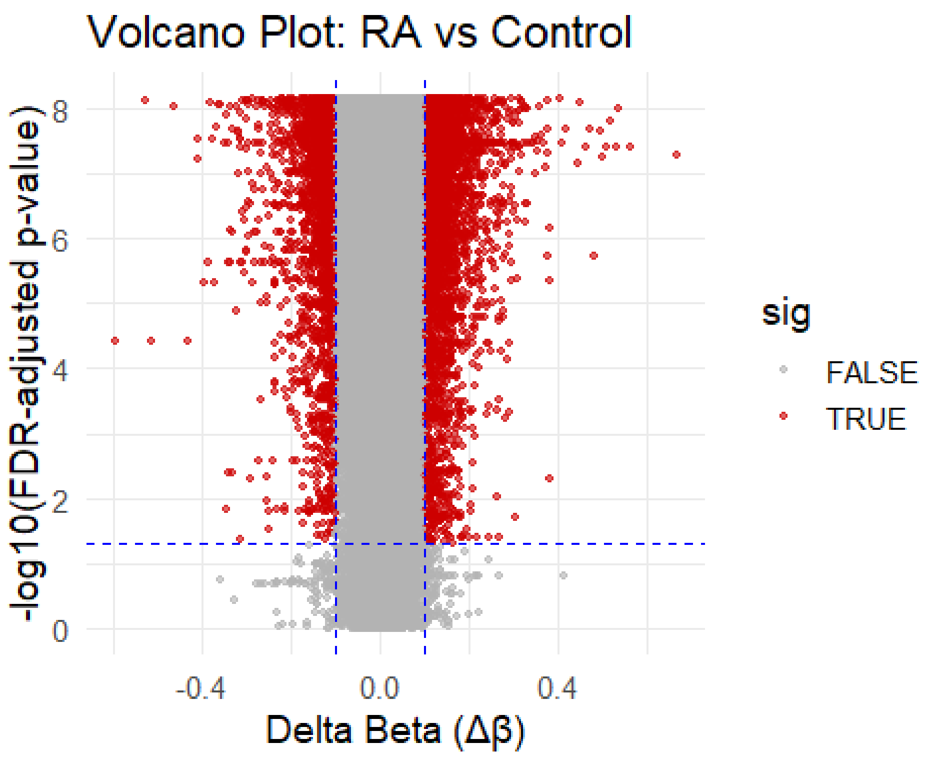

Top 10 GO Molecul
transcription regulatory region DNA binding
regulatory region nucleic acid binding
regulatory region DNA binding
sequence-specific DNA binding
double-stranded DNA binding
transcription regulatory region sequence-specific DNA binding
transcription factor activity, RNA polymerase II core promoter proximal region sequence-specific binding
RNA polymerase II regulatory region sequence-specific DNA binding
sequence-specific double-stranded DNA binding
RNA polymerase II regulatory region DNA binding
0.0
2.5
5.0
7.5
10.0
-log10(FDR-adjusted p-valu


Top 10 GO Cellular Component Enriched Terms (GREAT)
intracellular part
intracellular
neuron part
somatodendritic compartment
intracellular membrane-bounded organelle
plasma membrane region
membrane-bounded organelle
dendrite
neuron projection
intracellular organelle
0.0
2.5
5.0
7.5
-log10(FDR-adjusted p-value)

15. Volcano Plot

This R code creates a **volcano plot** to visualize differential DNA methylation between rheumatoid arthritis (RA) samples and controls. It first processes the topCpGs dataset by converting the row names (CpG probe IDs) into a column and then defines significance based on two criteria: an FDR-adjusted p-value less than 0.05 and an absolute deltaBeta (methylation difference) of at least 0.10.

```r
library(ggplot2)
library(tibble)
library(dplyr)

# Prepare data
res_df <- topCpGs %>%
  rownames_to_column("CpG") %>%
  mutate(
    sig = adj.P.Val < 0.05 & abs(deltaBeta) >= 0.10  # significance based on
FDR and effect size
  )

# Volcano plot using deltaBeta
ggplot(res_df, aes(x = deltaBeta, y = -log10(adj.P.Val))) +
  geom_point(aes(color = sig), alpha = 0.6, size = 1) +          # points
  scale_color_manual(values = c("grey70", "red3")) +             # non-signif
icant vs significant
  theme_minimal(base_size = 14) +
  labs(
    title = "Volcano Plot: RA vs Control",
    x = "Delta Beta (Δβ)",
    y = "-log10(FDR-adjusted p-value)"
  ) +
  geom_vline(xintercept = c(-0.10, 0.10), linetype = "dashed", color = "blue"
) +  # effect size cutoffs
  geom_hline(yintercept = -log10(0.05), linetype = "dashed", color = "blue")
+    # significance cutoff
  theme(legend.position = "right")
```



The ggplot2 plot maps deltaBeta on the x-axis and the negative log10 of the adjusted p-value on the y-axis, so that points higher up are more statistically significant and points further from zero on the x-axis have larger methylation differences. Each CpG is represented as a dot, colored red if significant and grey if not, with vertical dashed lines at ±0.10 showing the effect size threshold and a horizontal dashed line at -log10(0.05) showing the significance cutoff. The result is a clear visual summary highlighting which CpGs are both statistically significant and biologically meaningful in methylation change, allowing quick identification of hyper- or hypomethylated sites associated with RA.

16. Save Key Outputs

```r
out_dir <- file.path(base_dir, "results_revised")
dir.create(out_dir, recursive = TRUE, showWarnings = FALSE)

write.csv(topCpGs, file.path(out_dir, "DMP_all.csv"))
write.csv(sig_probes, file.path(out_dir, "DMP_significant.csv"))
write.csv(annot_sig, file.path(out_dir, "DMP_annotated.csv"))

# Only save DMRs if they exist
if (exists("dmr_ranges") && length(dmr_ranges) > 0) {
  write.csv(as.data.frame(dmr_ranges), file.path(out_dir, "DMRs.csv"))
  saveRDS(dmr_ranges, file.path(out_dir, "DMRs.rds"))
}
```

```r
saveRDS(go_tables, file.path(out_dir, "GREAT_results.rds"))
```

17. Session Info

```r
# Capture sessionInfo() output as text
si_text <- capture.output(sessionInfo())

# Print first 10 lines
cat(si_text[1:20], sep = "\n")
```

```
## R version 4.3.3 (2024-02-29 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 26200)
## 
## Matrix products: default
## 
## 
## locale:
## [1] LC_COLLATE=English_India.utf8  LC_CTYPE=English_India.utf8   
## [3] LC_MONETARY=English_India.utf8 LC_NUMERIC=C                  
## [5] LC_TIME=English_India.utf8    
## 
## time zone: Asia/Calcutta
## tzcode source: internal
## 
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets 
## [8] methods   base     
## 
## other attached packages:
```