Differential Analysis: Alzheimer's Disease in Female Prefrontal Cortex

Nolan Hamilton

March 2025

Contents

Packages

```
library(data.table)
library(DESeq2)
library(stringr)
library(limma)
library(ggplot2)
library(EnhancedVolcano)
library(svglite)
```

Functions

- `ruvg.full.ksvd` is a near exact copy of the implementation available on BioConductor, but preserves access to the vector of singular values returned by base R's `svd()`.

```
ruvg.full.ksvd <- function(x, cIdx, k, drop=0, center=TRUE, round=TRUE,
                           epsilon=1, tolerance=1e-8, isLog=FALSE) {

  if(isLog) {
    Y <- t(x)
  } else {
    Y <- t(log(x+epsilon))
  }
```

```
  if (center) {
    Ycenter <- apply(Y, 2, function(x) scale(x, center = TRUE, scale=FALSE))
  } else {
    Ycenter <- Y
  }
  if (drop >= k) {
    stop("'drop' must be less than 'k'.")
  }
  m <- nrow(Y)
  n <- ncol(Y)
  svdWa <- svd(Ycenter[, cIdx])
  first <- 1 + drop
  k <- min(k, max(which(svdWa$d > tolerance)))
  W <- svdWa$u[, (first:k), drop = FALSE]

  # --> here: diagonal matrix of singular values for each each singular vec.
  delta <- svdWa$d[first:k]
  # --> here: pct. captured with rank k approx of negative control regions
  pct <- sum(delta)/sum(svdWa$d)

  alpha <- solve(t(W) %*% W) %*% t(W) %*% Y
  correctedY <- Y - W %*% alpha
  if(!isLog) {
    if(round) {
      correctedY <- round(exp(correctedY) - epsilon)
      correctedY[correctedY<0] <- 0
    } else {
      correctedY <- exp(correctedY) - epsilon
    }
  }
  colnames(W) <- paste("W", seq(1, ncol(W)), sep="_")
  return(list(W = W, normalizedCounts = t(correctedY), delta=delta, pct=pct))
}
```

Constants/Significance Thresholds

- A peak region will be considered differentially accessible between the disease groups if:
  1. Final DESeq2 FDR-adjusted satisfies $p_{adj} <$ PADJ_THRESH
  2. The (absolute) fold change between patient groups $|\log_2(\text{FC})| >$ LFC_THRESH

```
INIT_COLDATA_FILE = "dnase_ad_metadata.tsv"
INIT_COUNT_MATRIX_FILE = "rocco_consenrich_dnase_ad_signal_v1.6.1.counts.tsv"
PADJ_THRESH = 0.05
LFC_THRESH = 0.25
```

Read and Parse Metadata and Peak-by-Sample Count Matrix

```
coldata <-  read.table(INIT_COLDATA_FILE, sep = "\t",header = TRUE)
cts = read.table(INIT_COUNT_MATRIX_FILE, sep = "\t", header = TRUE,
                 check.names = FALSE)
```

Metadata:

```
knitr::kable(coldata, format = "markdown")
```

sample	status	age	mapped_rlen	tissue	gender
ENCFF502IOV	No_AD	90	76	prefrontal_cortex	female
ENCFF306XXE	No_AD	84	76	prefrontal_cortex	female
ENCFF990EJE	No_AD	82	36	prefrontal_cortex	female
ENCFF195KCN	No_AD	83	76	prefrontal_cortex	female
ENCFF096RBN	No_AD	90	101	prefrontal_cortex	female
ENCFF375NXF	AD	87	76	prefrontal_cortex	female
ENCFF446JDN	AD	90	76	prefrontal_cortex	female
ENCFF617VLY	AD	89	36	prefrontal_cortex	female
ENCFF669HSH	AD	90	36	prefrontal_cortex	female
ENCFF647TPO	AD	89	76	prefrontal_cortex	female

```
rownames(coldata) <- coldata$sample
rownames(cts) <- cts$peak_name
cts = cts[,-1]

if(!all(rownames(coldata) == colnames(cts))) {
  stop('rownames of `coldata` must equal colnames of `cts`')
}

coldata$status <- as.factor(coldata$status)
coldata$status <- relevel(coldata$status, ref = "No_AD")
coldata$age <- as.numeric(coldata$age)
```

Determine Negative/Empirical Control Regions for RUVg

Per guidance in *RNA-seq workflow: gene-level exploratory analysis and differential expression - Section 8.2*

- First, run DEseq2 and obtain 'raw' p-values

```
dds <- DESeqDataSetFromMatrix(countData = cts, colData = coldata,
                              design= ~ age + status)
dds$status <- relevel(dds$status, ref = "No_AD")
dds <- estimateSizeFactors(dds)
dds <- estimateDispersions(dds)
dds <- nbinomLRT(dds, maxit = 1000, reduced = ~1)
uncorrected_results <- results(dds)
uncorrected_results <- na.exclude(uncorrected_results)
regions = rownames(uncorrected_results)
```

- Define a criteria for the negative/empirical control regions

```
start_end <- strsplit(gsub("chr", "", regions), "_")
start <- as.numeric(sapply(start_end, function(x) as.numeric(x[2])))
end <- as.numeric(sapply(start_end, function(x) as.numeric(x[3])))

# We restrict negative/empirical control regions to those
# greater than 100bp and less than 5000bp
small_regions <- regions[end - start < 100 & grepl("^chr[[:alnum:][:punct:]]+_",
                                                   regions)]
```

```
large_regions <- regions[end - start > 5000 & grepl("^chr[[:alnum:][:punct:]]+_",
                                                    regions)]

# We restrict the negative/empirical control regions to autosomal chromosomes
XY_regions <- regions[grepl("chrX|chrY", regions)]

# Per (Love, Anders, Kim, Huber, 2019), we restrict negative/empirical control
# regions to those 'initial' pvalues aboth a threshold (0.50)
nonnull_regions <- rownames(uncorrected_results[uncorrected_results$pvalue < 0.50, ])

ctrlregion=uncorrected_results
ctrlregion = ctrlregion[! match(rownames(ctrlregion), XY_regions, nomatch=0),]
ctrlregion = ctrlregion[! match(rownames(ctrlregion), small_regions, nomatch=0),]
ctrlregion = ctrlregion[! match(rownames(ctrlregion), large_regions, nomatch=0),]
# remove nonnull_regions
ctrlregion = ctrlregion[! match(rownames(ctrlregion), nonnull_regions, nomatch=0),]
ctrl_length = length(rownames(ctrlregion))
ctrlregion_names <- rownames(ctrlregion)
```

- Restrict number of negative/empirical control regions to at most 5000.

```
set.seed(123456)
ctrlregion_names <- ctrlregion_names[sample(length(ctrlregion_names), 5000,
                                            replace = FALSE)]
ctrlregion <- ctrlregion[(rownames(ctrlregion) %in% ctrlregion_names), ]
```

- Apply DESeq2's variance-stabilizing transformation to mitigate heteroskedasticity
  - Note: vst data converges in scale to log2 – which Limma's `removeBatchEffect` expects in later steps

```
vsd <- vst(dds, blind=TRUE)
mm <- model.matrix(~1, colData(vsd))
```

- Remove *known*, easy-to-model effects with a linear mixed model
- We apply Limma's `removeBatchEffect`
- Note: `removeBatchEffect` expects log-scaled counts as input and returns log-scaled data

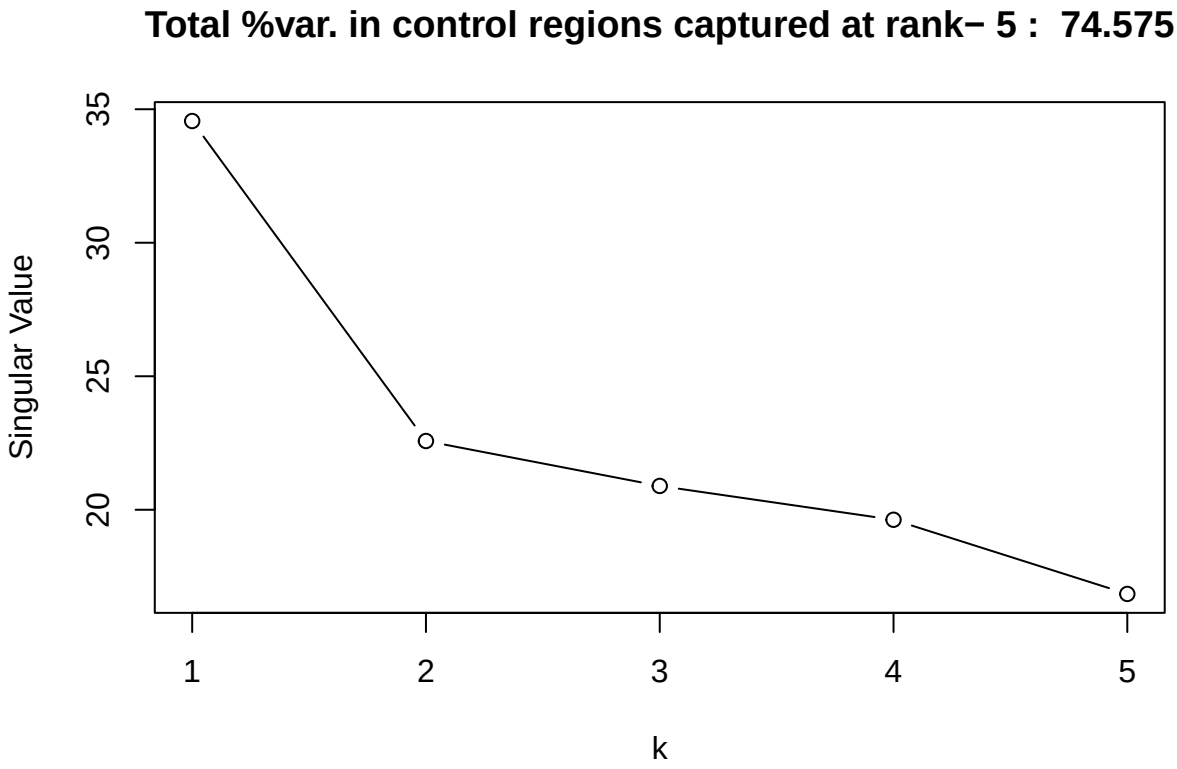
```
assay(vsd) <- removeBatchEffect(assay(vsd), covariates = coldata$age, design = mm)
```

Determine Number of RUV Factors to Represent Negative Control Regions

- RUV factors: left-singular vectors of the count matrix defined over negative/empirical control regions.
- Here, we plot the corresponding singular values of each left-singular vector.
  - Using too few RUV factors → Poor representation of unwanted variation in negative control regions
  - Using too many RUV factors → Increased model complexity (consider sample size in relation to K) for minor reduction in unwanted variation.

```
eval_K = 5

RUVg_results <- ruvg.full.ksvd(assay(vsd), ctrlregion_names, k = eval_K, isLog=TRUE)
ruv_singular_values <- RUVg_results$d
pct = RUVg_results$pct
plot(1:length(ruv_singular_values), ruv_singular_values, type = "b", xlab = "k", ylab = "Singular Value
       main=paste("Total %var. in control regions captured at rank-",eval_K,": ",
```

```
                round(pct*100,digits=3)))
```



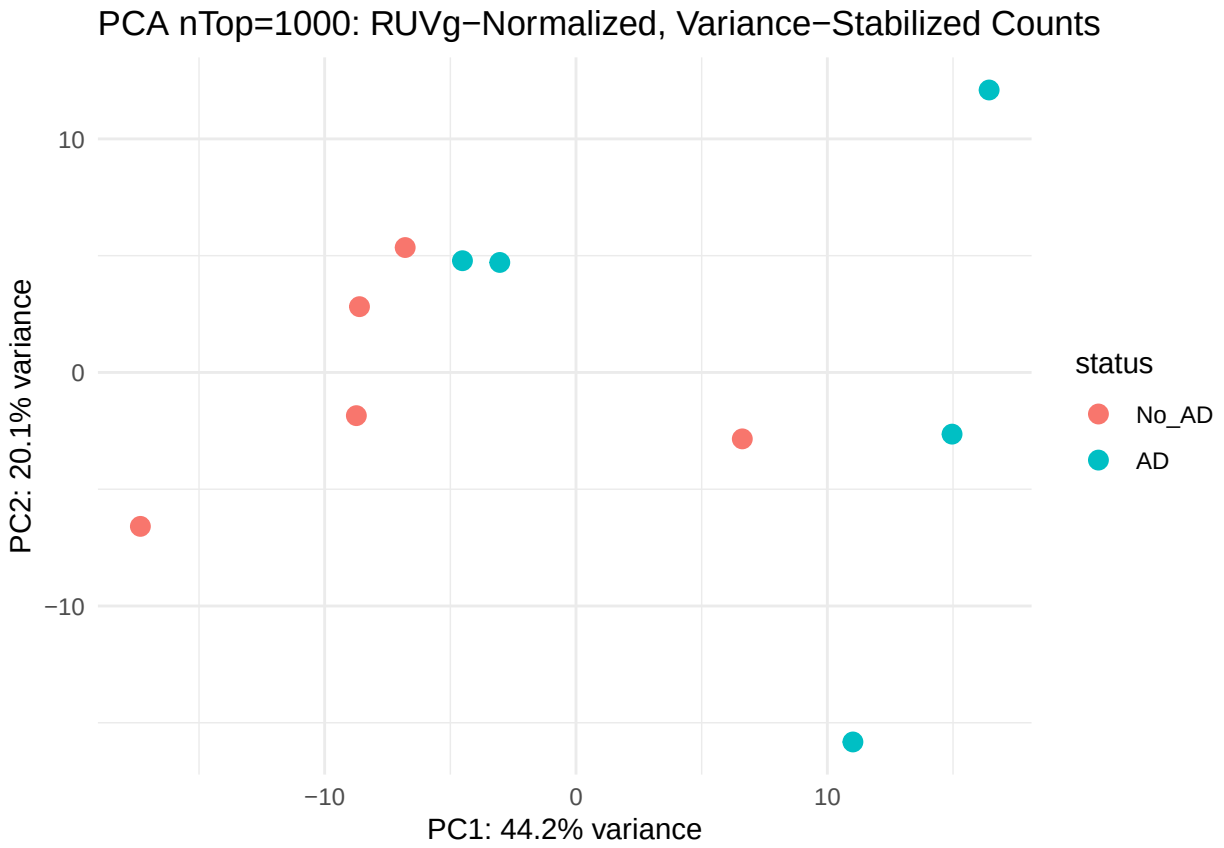
In light of the above, we choose $K = 2$ as the number of RUV factors to represent the negative control regions.

```
K=2
RUVg_results <- ruvg.full.ksvd(assay(vsd), ctrlregion_names, k = K, isLog=TRUE)
ruvg_normalized_counts <- RUVg_results$normalizedCounts
ruv_factors  <- RUVg_results$W
```

PCAs: RUVg-Normalized, Batch-Corrected, Variance-Stabilized Count Data

- Colored by disease status

```
assay(vsd) <- ruvg_normalized_counts
pcaData <- DESeq2::plotPCA(vsd, intgroup = "status", ntop=1000,
                           returnData = TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"), 1)
ggplot(pcaData, aes(x = PC1, y = PC2, color = status)) +
  geom_point(size = 3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance")) +
  theme_minimal() + ggtitle("PCA nTop=1000: RUVg-Normalized, Variance-Stabilized Counts")
```




```
ggsave(paste0("dnase_ad_k", K, "_pca_nTop1000.svg"))
```

Apply DESeq2 with RUVg-Augmented Design Formula on Raw Counts

- Note that the previous normalization/transformation steps were only to determine the RUV factors and generate a PCA plot for informal visualization.
- Now, DESeq2 is supplied the *raw* count data that it expects–but with an augmented design formula that ensures effects due to the unwanted variation observed in our approximation of negative controls does not influence testing differential analysis

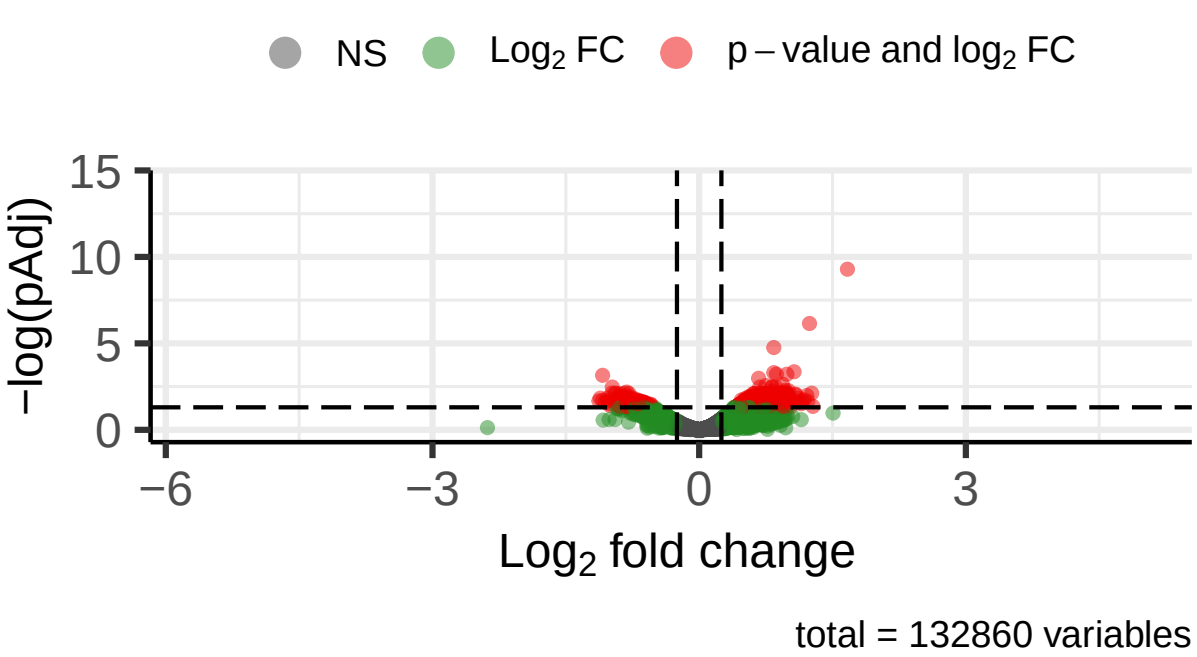
```
# restart with raw counts/metadata
coldata <-  read.table(INIT_COLDATA_FILE, sep = "\t",header = TRUE)
cts = read.table(INIT_COUNT_MATRIX_FILE, sep = "\t", header = TRUE,
                 check.names = FALSE)

rownames(coldata) <- coldata$sample
rownames(cts) <- cts$peak_name
cts = cts[,-1]

if(!all(rownames(coldata) == colnames(cts))) {
  stop('rownames of `coldata` must equal colnames of `cts`')
}
coldata$status <- as.factor(coldata$status)
coldata$status <- relevel(dds$status, ref = "No_AD")
coldata$age <- as.numeric(coldata$age)

w_terms <- paste0("W_", 1:K)
```

```
coldata <- cbind(coldata, ruv_factors)
design_formula <- as.formula(paste0("~ age +", paste0(w_terms, collapse = "+"),
                                    "+ status"))
dds_final <- DESeqDataSetFromMatrix(countData = cts, colData = coldata,
                                    design = design_formula)
dds_final <- DESeq(dds_final)
results <- results(dds_final)

design_formula <- as.formula(
  paste0("~ age + ", paste0(w_terms, collapse = "+"),
         "+ status")
)

dds_final <- DESeqDataSetFromMatrix(
  countData = cts,
  colData = coldata,
  design = design_formula
)
dds_final <- DESeq(dds_final)
results <- results(dds_final)

significant_results <- results[
  complete.cases(results) &
    (results$padj < PADJ_THRESH),]

significant_results <- significant_results[
  abs(significant_results$log2FoldChange) > LFC_THRESH,]

significant_regions <- rownames(significant_results)
```

Volcano Plot

- Regions satisfying $p_{adj} <$ PADJ_THRESH and |lg2fc| $>$ LFC_THRESH are considered differentially accessible (red).

```
# EnhancedVolcano plot
EnhancedVolcano(results,
                lab = '',
                x = 'log2FoldChange',
                y = 'padj',
                ylab = '-log(pAdj)',
                title = 'DARs between AD and non-AD',
                pCutoff = PADJ_THRESH,
                FCcutoff = LFC_THRESH,
                cutoffLineWidth = 0.75)
```




```
  ggsave(paste0("dnase_ad_DESeq_results_k", K, "_volcano.svg"))
```

Save DAR results in BED format

- A bed file of regions satisfying the criteria above is saved for downstream analysis.

```
start_end <- strsplit(gsub("chr", "", significant_regions), "_")
chromosome <- sapply(start_end, function(x) x[1])
start <- as.numeric(sapply(start_end, function(x) x[2]))
end <- as.numeric(sapply(start_end, function(x) x[3]))
bed_output <- data.frame(
  chrom = paste0('chr',chromosome),
  chromStart = start,
  chromEnd = end
  )

  write.table(bed_output,
              file = paste0("dnase_ad_DESeq_results_k", K, ".bed"),
              sep = "\t", quote = FALSE, row.names = FALSE, col.names = FALSE)
```

Per-Chromosome Differential Results

```
write.table(significant_results,
            file = paste0("dnase_ad_DESeq_results_k", K, ".tsv"),
            sep = "\t", quote = FALSE, row.names = TRUE)
chromosomes <- unique(chromosome)
positive_counts <- sapply(chromosomes,
                          function(x) sum(significant_results[chromosome == x,]$log2FoldChange > 0))
```

```
negative_counts <- sapply(chromosomes,
                          function(x) sum(significant_results[chromosome == x,]$log2FoldChange < 0))
total_counts <- sapply(chromosomes,
                       function(x) sum(significant_results[chromosome == x,]$log2FoldChange != 0))
chromosome_counts <- data.frame(chromosome = chromosomes,
                                total=total_counts,
                                positive = positive_counts,
                                negative = negative_counts)
chromosome_counts <- chromosome_counts[order(chromosome_counts$total,
                                             decreasing = TRUE),]
knitr::kable(chromosome_counts,
             format = "markdown",
             col.names = c("Chromosome", "Total DARS", "Positive LFC", "Negative LFC"), row.names=FALSE)
```

Chromosome	Total DARS	Positive LFC	Negative LFC
19	125	119	6
1	108	94	14
17	83	71	12
2	80	66	14
11	78	70	8
7	66	55	11
16	61	57	4
9	57	52	5
3	56	43	13
12	52	44	8
8	51	48	3
6	44	35	9
5	42	35	7
10	42	38	4
22	40	40	0
14	39	34	5
15	39	30	9
20	39	31	8
4	28	23	5
X	16	15	1
21	15	14	1
13	12	10	2
18	10	6	4

Distribution: Approximated Regulatory Roles of Differentially Accessible Regions

```
library(TxDb.Hsapiens.UCSC.hg38.knownGene)
library(ChIPseeker)
library(clusterProfiler)
library(org.Hs.eg.db)

txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene
peaks <- readPeakFile(paste0("dnase_ad_DESeq_results_k", K, ".bed"))

peakAnno <- annotatePeak(peaks, tssRegion=c(-1000, 1000),
                         TxDb=txdb, annoDb="org.Hs.eg.db")

## >> preparing features information...      2025-03-20 08:51:05
```

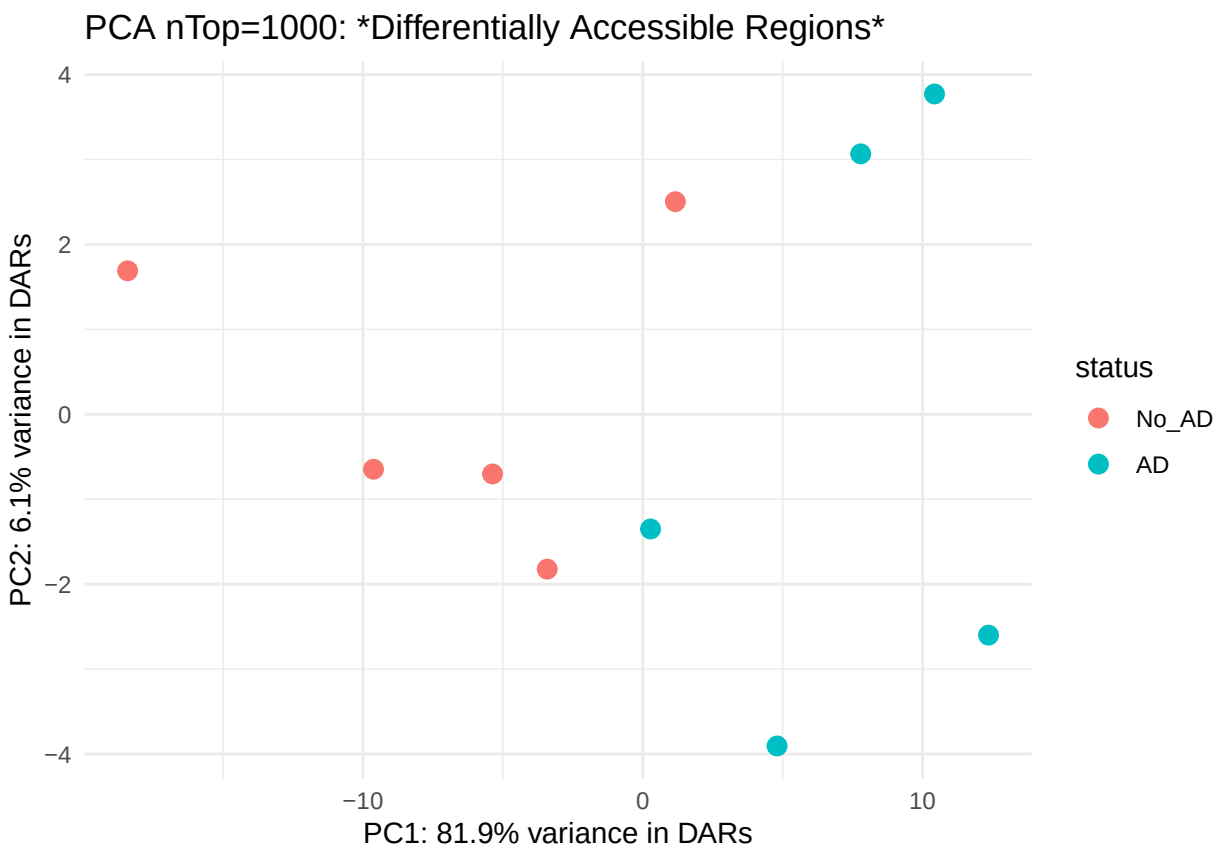
```
## >> identifying nearest features...        2025-03-20 08:51:05
## >> calculating distance from peak to TSS...   2025-03-20 08:51:06
## >> assigning genomic annotation...        2025-03-20 08:51:06
## >> adding gene annotation...          2025-03-20 08:51:18
## >> assigning chromosome lengths           2025-03-20 08:51:18
## >> done...                    2025-03-20 08:51:18
```

```
knitr::kable(peakAnno@annoStat, format = "markdown",
             col.names = c("Feature Type", "Frequency"), row.names = FALSE)
```

Feature Type	Frequency
Promoter	42.4344886
5' UTR	0.6762468
3' UTR	3.2967033
1st Exon	1.9442096
Other Exon	4.4801352
1st Intron	13.3558749
Other Intron	17.9205410
Downstream (<=300)	0.1690617
Distal Intergenic	15.7227388

PCA over Differentially Accessible Regions

```
dar <- vsd[significant_regions,]
pcaData <- DESeq2::plotPCA(dar, intgroup = "status", ntop=1000,
                           returnData = TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"), 1)

ggplot(pcaData, aes(x = PC1, y = PC2, color = status)) +
  geom_point(size = 3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance in DARs")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance in DARs")) + theme_minimal() +
  ggtitle("PCA nTop=1000: *Differentially Accessible Regions*")
```




```
ggsave(paste0("dnase_ad_DESeq_results", K, "_pca_dar_nTop1000.svg"))
```