Transcriptomics Data Preparation and Exploratory Data Analysis Note

Data Preprocessing and Exploratory Data Analysis (EDA) are the initial steps of any transcriptomics workflow and they are crucial for transforming raw sequencing counts into reliable biological insights. This note explains the purpose of each major block of code in the R script.

1. Data Preprocessing: Transforming Raw Counts

A. Reading and Cleaning the Count Matrix

This stage prepares the raw count data for statistical analysis by ensuring all samples are comparable and all genes are relevant.

```
read.csv(...)
```

The raw gene count matrix is loaded. It contains Ensembl Gene IDs in the first column (X) and raw counts in the subsequent columns (one per sample).

```
summary(), str(), dim()
```

Initial Data Inspection to check for data types, missing values (NA), and the overall dimensions of the dataset. This is good practice for data integrity.

```
data1 <- my_data[, c("X", "AN13999.BA32", ...)]
```

Subsetting the count matrix to include only the desired samples for the current comparison (e.g., Alzheimer's Disease (AD) vs. Control).

```
groups <- factor(c("AD", ...)) and target <- data.frame(...)
```

Creating the metadata/design matrix. This defines the experimental groups ("AD" and "Control") for each sample. This is essential for all differential expression analysis.

B. Gene Annotation (`biomaRt`)

```
gene_names<-getBM(attributes=c("ensembl_gene_id",

"external_gene_name"),filters = "ensembl_gene_id",

values = genes,mart = ensembl)
```

Gene Annotation is the process of mapping technical identifiers (like Ensembl Gene IDs) to biologically meaningful and readable names (external gene names or common gene symbols, e.g., *APOE*). This is accomplished using the biomaRt package to query public databases (like Ensembl).

Why it is necessary:

- **Interpretability:** Gene IDs are technical and difficult to work with. Annotation makes the final results table (the list of differentially expressed genes) immediately comprehensible to biologists.
- **Standardization:** Ensures that the output uses universally recognized gene symbols.
- **Downstream Tools:** Readable gene names are often required as input for functional analysis tools (like the `clusterProfiler` and `gprofiler2` steps you have later in the analysis).

C. Filtering and Normalization (edgeR)

DGEList Object

```
my_deglist <- DGEList(...)
```

A DGEList is a specialized data object used by the edgeR package. It stores all the essential information for a DGE analysis in a single structured list, including the raw gene count matrix, sample information (like experimental groups), and other meta-data.

Why it is Necessary

- **Compatibility:** A `DGEList` object organizes your count data and library information in a structure that is immediately compatible with all downstream functions in the `edgeR` package.

- **Efficiency and Accuracy:** It ensures that core `edgeR` functions, such as `cpm()` (for count normalization), `calcNormFactors()` (for TMM normalization), and the differential expression analysis itself, can efficiently and correctly access all required components (e.g., raw counts, group definitions, library sizes).
- **Workflow Simplification:** Using `DGEList` streamlines the transcriptomics workflow in R, reducing common errors and the need for complex, manual data management between steps.

D. Normalization (CPM)

```
cpm <- cpm(my_deglist)
```

CPM is a normalization method that scales raw gene counts by the total number of mapped reads in a sample and expresses them as counts per one million reads. The formula is:

$CPM = Raw\ Counts \div Total\ Mapped\ Reads \times 10\text{^}6$

Why it is necessary:

- **Bias in Raw Counts:** Raw gene counts are heavily influenced by the sequencing depth (total number of reads) of each sample, which is a technical factor.
- **The Problem:** A sample with more total reads will naturally have higher raw counts for a gene, even if the gene's true expression level is biologically unchanged. This makes direct comparison misleading.
- **The Solution:** CPM (Counts Per Million) normalization removes this technical bias by scaling the raw counts to a fixed reference (one million total reads).
- **The Benefit:** This allows for a fair and accurate comparison of gene expression levels between all samples, regardless of their original sequencing depth.

E. Filtering Lowly Expressed Genes

```
fil_threshold <- rowSums(cpm > 1) >= 2
```

This step removes genes that are expressed at extremely low levels across all samples. Using the `edgeR` package, we first convert raw counts to Counts Per Million (CPM) to account for library size differences. The common threshold keeps only genes with a minimum of 1 CPM in at least 2 samples.

Why it is necessary:

- **Low Information:** Genes with near-zero counts provide little statistical evidence for differential expression.

- **Noise Reduction:** Removing these genes significantly reduces background noise and the number of multiple testing corrections needed.
- **Statistical Power:** It improves the statistical power of the analysis to detect truly differentially expressed genes.

F. Normalization (TMM)

```
my_deglist.filtered.Norm <- calcNormFactors(my_deglist.filtered, method = "TMM")
```

TMM (Trimmed Mean of M-values) is a robust normalization method implemented in the edgeR package. It corrects for differences in library size (sequencing depth) and for compositional bias, which occurs when a few highly expressed genes dominate a library, skewing the overall read counts.

Why it is necessary:

- **Addressing Compositional Bias:** While simple CPM (Counts Per Million) corrects for library size (sequencing depth), it does not account for compositional bias. This bias occurs when a few highly expressed genes or a subset of consistently differentially expressed genes dominate the total read count, skewing the overall gene expression percentages.
- **Calculating Robust Factors:** TMM (Trimmed Mean of M-values) addresses this by identifying a core set of genes that are assumed to be non-differentially expressed. It then calculates the average fold change (the M-value) among these core genes.
- **Ensuring Comparability:** This calculated average fold change is used to derive a robust normalization factor for each sample. Applying this factor ensures that the expression levels of the majority of genes are accurately comparable between all samples, regardless of the strong expression signals from a few outlier genes

2. Exploratory Data Analysis (EDA): Checking Data Quality

EDA is a quality control process that ensures that the data quality is sufficient and that the primary biological signal is the largest source of variation not technical noise.

A. Hierarchical Clustering

```
hclust(distance, method = "complete")
```

This is a classification method that groups similar samples together based on their gene expression profiles. It visualizes the relationships as a tree-like dendrogram.

- Method: Calculates the Euclidean distance between all samples based on their logCPM expression profiles and then performs hierarchical clustering.
- Purpose: Visualize Sample Relationships. The resulting dendrogram shows which samples are most similar to each other. Ideally, samples from the same biological group (AD or Control) should cluster together.

B. Principal Component Analysis (PCA)

```
pca.res <- prcomp(t(log2.cpm.filtered.norm), scale. = F, retx = T)
```

A statistical procedure that reduces the dimensionality of data while retaining as much variation as possible, often for visualization. It transforms observations into a set of uncorrelated principal components.

- **Method:** PCA is a dimensionality reduction technique that identifies the directions (Principal Components, or PCs) of largest variance in the data. The first few PCs usually capture the most important biological or technical differences.
- **Purpose:** Identify Key Sources of Variation and Check for Outliers.
- PC1 and PC2 plots (and subsequent PC plots): You plot the samples onto the new PC coordinates, colored by the biological group.
- The samples should ideally separate clearly along PC1 or PC2 based on the groups you defined ("AD" vs. "Control"). If they separate based on an unlisted technical factor (like the batch/date they were sequenced), that indicates a potential batch effect.
- The percentage in the axis labels (e.g., PC1(25%)) shows the amount of total variance explained by that component.