Dopamine MSNs Epigenetic Analysis: D1 vs D2 with Enhanced Methods

Deepak Poduval

2025-08-25

## Load and Initial Data Exploration

# Load the RDS file  
so <- readRDS("raw\_data/GSE167920\_Results\_full\_nuclei\_processed\_final.rds")  
  
# Explore data structure  
cat("Data Overview\n")

## Data Overview

cat(paste("Number of cells:", ncol(so)), "\n")

## Number of cells: 63884

cat(paste("Number of genes:", nrow(so)), "\n")

## Number of genes: 2000

cat(paste("Available assays:", paste(names(so@assays), collapse = ", ")), "\n")

## Available assays: RNA, integrated

# Check metadata  
cat("Metadata Columns\n")

## Metadata Columns

print(colnames(so@meta.data))

## [1] "cell.id" "nFeatures\_RNA" "nCount.RNA" "percent.ribo"   
## [5] "seurat\_clusters" "region\_name" "cell\_type" "cell\_type\_2"   
## [9] "monkey"

# Check cell types  
if ("cell\_type" %in% colnames(so@meta.data)) {  
 cat("Cell Type Distribution\n")  
 print(table(so@meta.data$cell\_type))  
}

## Cell Type Distribution  
##   
## Oligos Astrocytes DRD2 DRD1   
## 14314 17424 7310 9148   
## Interneurons Oligos\_Pre Microglia Endothelial   
## 7629 3079 2430 1723   
## Mural/Fibroblast Unknown1 Unknown2 Unknown3   
## 827 0 0 0   
## Unknown4   
## 0

# Check for D1/D2 MSN populations  
cat("Checking For D1/D2 MSNs\n")

## Checking For D1/D2 MSNs

msn\_types <- grep("DRD|D1|D2|MSN", unique(so@meta.data$cell\_type), value = TRUE, ignore.case = TRUE)  
cat(paste("MSN-related cell types found:", paste(msn\_types, collapse = ", ")), "\n")

## MSN-related cell types found: DRD1, DRD2

# Basic metadata structure  
str(so@meta.data, max.level = 1)

## 'data.frame': 63884 obs. of 9 variables:  
## $ cell.id : int 61609 61619 61620 61636 61661 61682 61718 61719 61720 61741 ...  
## $ nFeatures\_RNA : int 5061 3604 3940 5650 5248 4431 5151 3452 3531 4684 ...  
## $ nCount.RNA : num 15884 8927 9198 20929 17577 ...  
## $ percent.ribo : num 0.00327 0.00493 0.00402 0.00253 0.00256 ...  
## $ seurat\_clusters: Factor w/ 44 levels "0","1","2","3",..: 25 25 17 10 17 25 10 33 33 17 ...  
## $ region\_name : Factor w/ 3 levels "caudate","nacc",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ cell\_type : Factor w/ 13 levels "Oligos","Astrocytes",..: 4 4 4 4 4 4 4 4 4 4 ...  
## $ cell\_type\_2 : Factor w/ 8 levels "Oligos","Astrocytes",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ monkey : Factor w/ 2 levels "Monkey\_F","Monkey\_P": 1 1 1 1 1 1 1 1 1 1 ...

## Quality Control Assessment

# Clear any problematic graph objects first  
so@graphs <- list()  
  
# Calculate additional QC metrics if not present  
if (!"percent.mt" %in% colnames(so@meta.data)) {  
 so[["percent.mt"]] <- PercentageFeatureSet(so, pattern = "^MT-")  
}  
if (!"percent.ribo" %in% colnames(so@meta.data)) {  
 so[["percent.ribo"]] <- PercentageFeatureSet(so, pattern = "^RP[SL]")  
}  
  
# QC summaries  
summary(so@meta.data$nFeatures\_RNA)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1001 2074 2770 3209 4361 11404

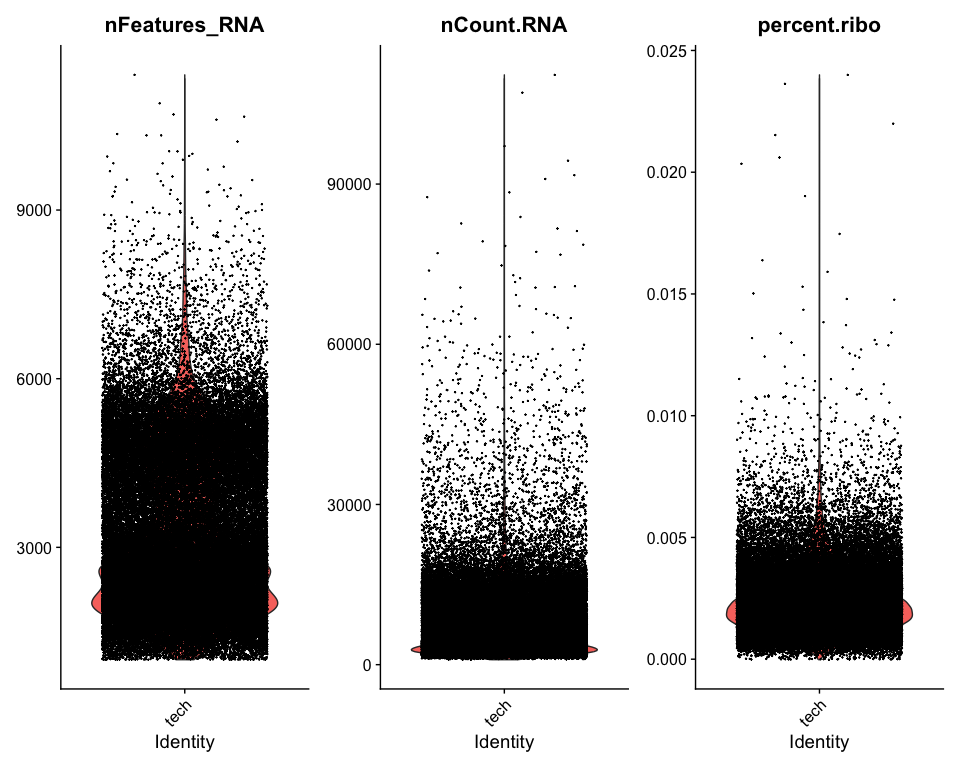
summary(so@meta.data$nCount.RNA)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 926.6 2932.8 4681.0 7260.8 10068.0 110443.0

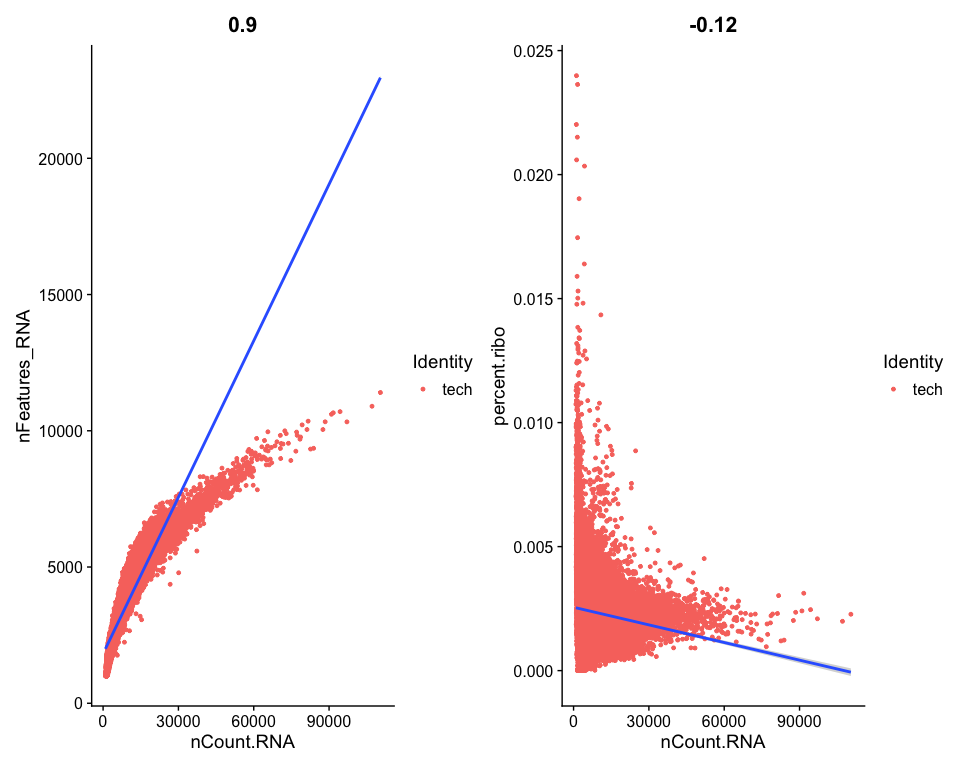
summary(so@meta.data$percent.ribo)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000000 0.001516 0.002180 0.002383 0.002970 0.023988

# QC plots  
p1 <- VlnPlot(so, features = c("nFeatures\_RNA", "nCount.RNA", "percent.ribo"), ncol = 3, pt.size = 0.1)  
print(p1)



ggsave("D1vD2\_figures/QC\_violin\_plots.png", p1, width = 15, height = 6)  
  
p2 <- FeatureScatter(so, feature1 = "nCount.RNA", feature2 = "nFeatures\_RNA") + geom\_smooth(method = "lm")  
p3 <- FeatureScatter(so, feature1 = "nCount.RNA", feature2 = "percent.ribo") + geom\_smooth(method = "lm")  
p4 <- p2 + p3  
print(p4)



ggsave("D1vD2\_figures/QC\_scatter\_plots.png", p4, width = 12, height = 6)

## Data Filtering and Quality Control

cat("Before Filtering\n")

## Before Filtering

cat(paste("Cells:", ncol(so)), "\n")

## Cells: 63884

cat(paste("Genes:", nrow(so)), "\n")

## Genes: 2000

if ("cell\_type" %in% colnames(so@meta.data)) {  
 cat("Cell Type Distribution Before Filtering\n")  
 print(table(so@meta.data$cell\_type))  
}

## Cell Type Distribution Before Filtering  
##   
## Oligos Astrocytes DRD2 DRD1   
## 14314 17424 7310 9148   
## Interneurons Oligos\_Pre Microglia Endothelial   
## 7629 3079 2430 1723   
## Mural/Fibroblast Unknown1 Unknown2 Unknown3   
## 827 0 0 0   
## Unknown4   
## 0

# Apply filtering based on QC metrics  
so <- subset(so, subset = nFeatures\_RNA > 200 &  
 nFeatures\_RNA < 8000 &  
 nCount.RNA > 1000 &  
 nCount.RNA < 80000 &  
 percent.ribo < 0.02)  
  
cat("After Filtering\n")

## After Filtering

cat(paste("Cells:", ncol(so)), "\n")

## Cells: 63683

cat(paste("Genes:", nrow(so)), "\n")

## Genes: 2000

if ("cell\_type" %in% colnames(so@meta.data)) {  
 cat("Cell Type Distribution After Filtering\n")  
 print(table(so@meta.data$cell\_type))  
}

## Cell Type Distribution After Filtering  
##   
## Oligos Astrocytes DRD2 DRD1   
## 14311 17422 7278 9117   
## Interneurons Oligos\_Pre Microglia Endothelial   
## 7504 3078 2430 1716   
## Mural/Fibroblast   
## 827

## Create MSN Subset

# Filter for MSN cells only  
if (length(msn\_types) > 0) {  
 so\_msn <- subset(so, subset = cell\_type %in% msn\_types)  
 cat(paste("MSN cells retained:", ncol(so\_msn)), "\n")  
 cat("MSN cell type distribution:\n")  
 print(table(so\_msn@meta.data$cell\_type))  
} else {  
 # If no specific MSN cell types, try to identify by marker expression  
 cat("No MSN cell types found, attempting to identify by markers...\n")  
   
 # Check for DRD1/DRD2 expression to identify MSNs  
 if (all(c("DRD1", "DRD2") %in% rownames(so))) {  
 drd1\_exp <- GetAssayData(so, assay = "RNA", layer = "data")["DRD1",]  
 drd2\_exp <- GetAssayData(so, assay = "RNA", layer = "data")["DRD2",]  
   
 # Identify cells with either DRD1 or DRD2 expression  
 msn\_cells <- which(drd1\_exp > 0.5 | drd2\_exp > 0.5)  
   
 if (length(msn\_cells) > 50) {  
 so\_msn <- so[, msn\_cells]  
 cat(paste("MSN cells identified by expression:", ncol(so\_msn)), "\n")  
 } else {  
 cat("Insufficient MSN cells identified, using all cells\n")  
 so\_msn <- so  
 }  
 } else {  
 cat("DRD1/DRD2 not found, using all cells\n")  
 so\_msn <- so  
 }  
}

## MSN cells retained: 16395   
## MSN cell type distribution:  
##   
## DRD2 DRD1   
## 7278 9117

## Data Normalization and Scaling (MSN subset)

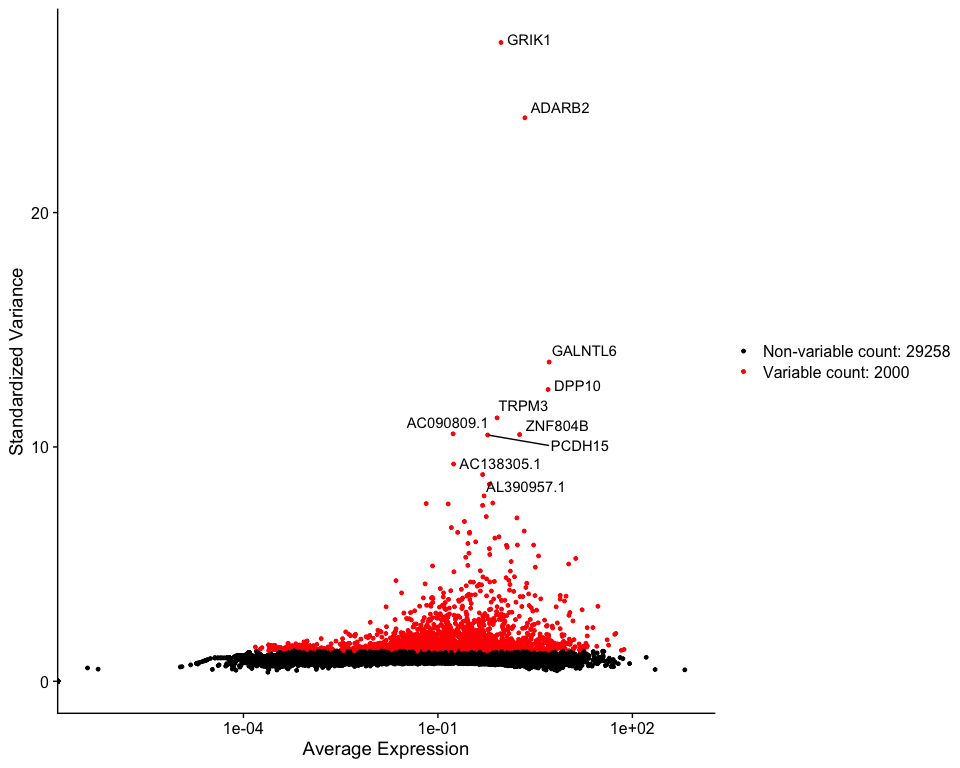
DefaultAssay(so\_msn) <- "RNA"  
  
# Normalize if needed  
if (is.null(so\_msn[["RNA"]]@data) || ncol(so\_msn[["RNA"]]@data) == 0) {  
 cat("Normalizing MSN RNA data...\n")  
 so\_msn <- NormalizeData(so\_msn)  
} else {  
 cat("MSN RNA data already normalized\n")  
}

## MSN RNA data already normalized

# Variable features  
if (length(VariableFeatures(so\_msn)) == 0) {  
 cat("Finding variable features for MSNs...\n")  
 so\_msn <- FindVariableFeatures(so\_msn, selection.method = "vst", nfeatures = 2000)  
} else {  
 cat("Variable features already identified for MSNs\n")  
}

## Finding variable features for MSNs...

# Plot variable features  
top10 <- head(VariableFeatures(so\_msn), 10)  
p5 <- VariableFeaturePlot(so\_msn)  
p6 <- LabelPoints(plot = p5, points = top10, repel = TRUE)  
print(p6)

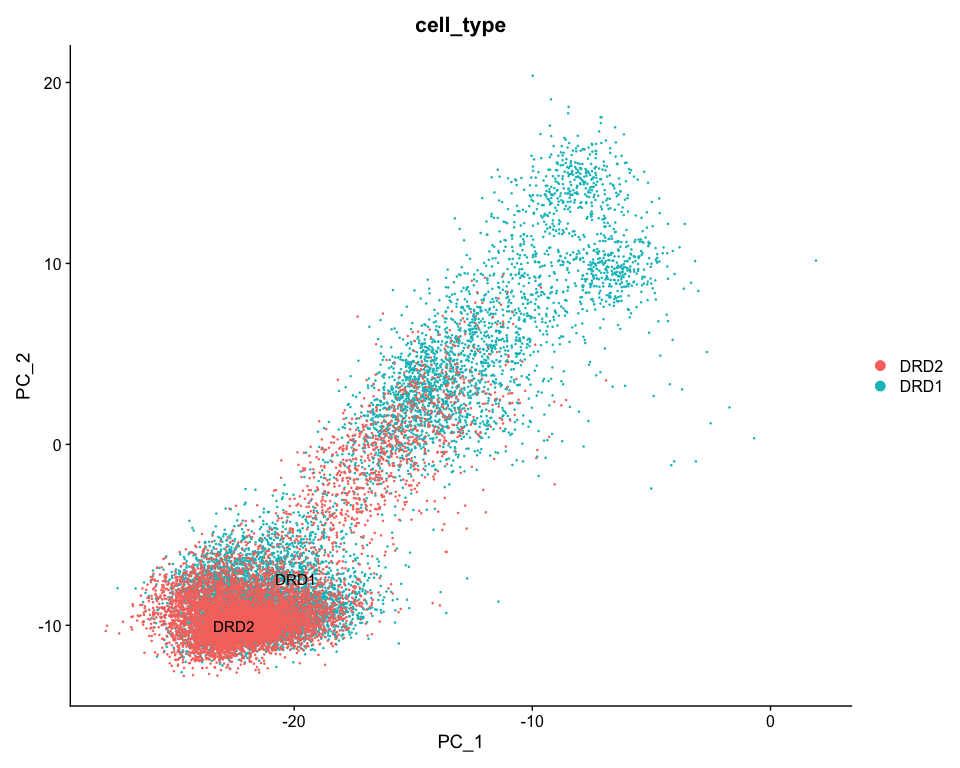


ggsave("D1vD2\_figures/Variable\_Features\_MSN.png", p6, width = 12, height = 8)  
  
# Scale only variable features for speed  
if (is.null(so\_msn[["RNA"]]@scale.data) || ncol(so\_msn[["RNA"]]@scale.data) == 0) {  
 cat("Scaling MSN RNA variable features...\n")  
 so\_msn <- ScaleData(so\_msn, features = VariableFeatures(so\_msn))  
} else {  
 cat("MSN RNA data already scaled\n")  
}

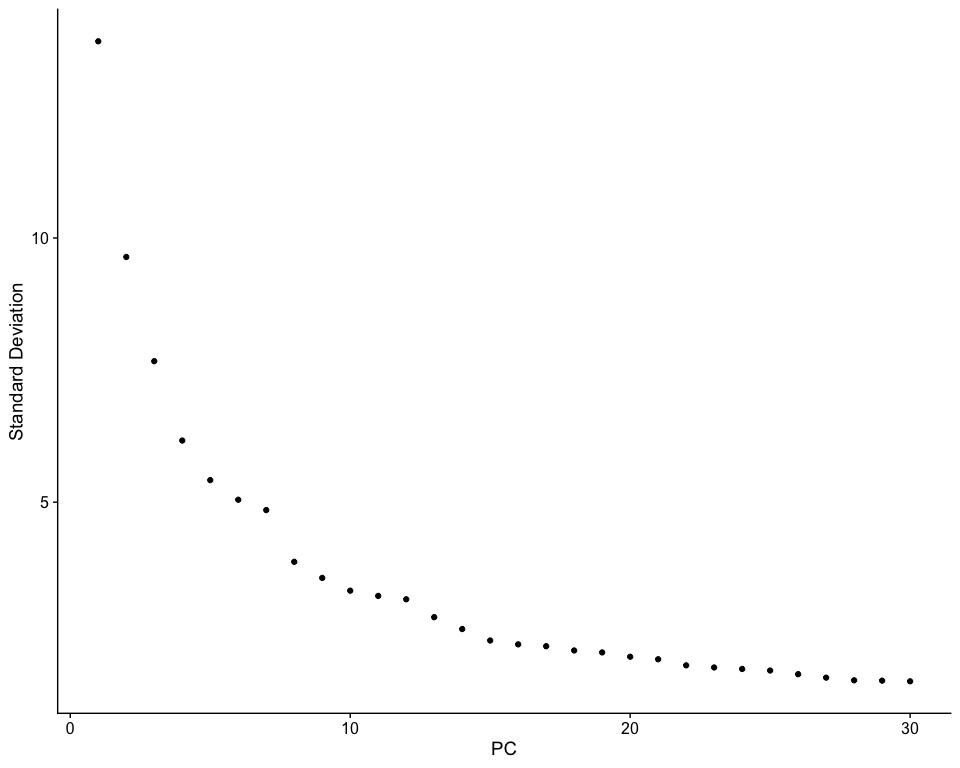
## Scaling MSN RNA variable features...

## PCA and Dimensionality Reduction (MSN subset)

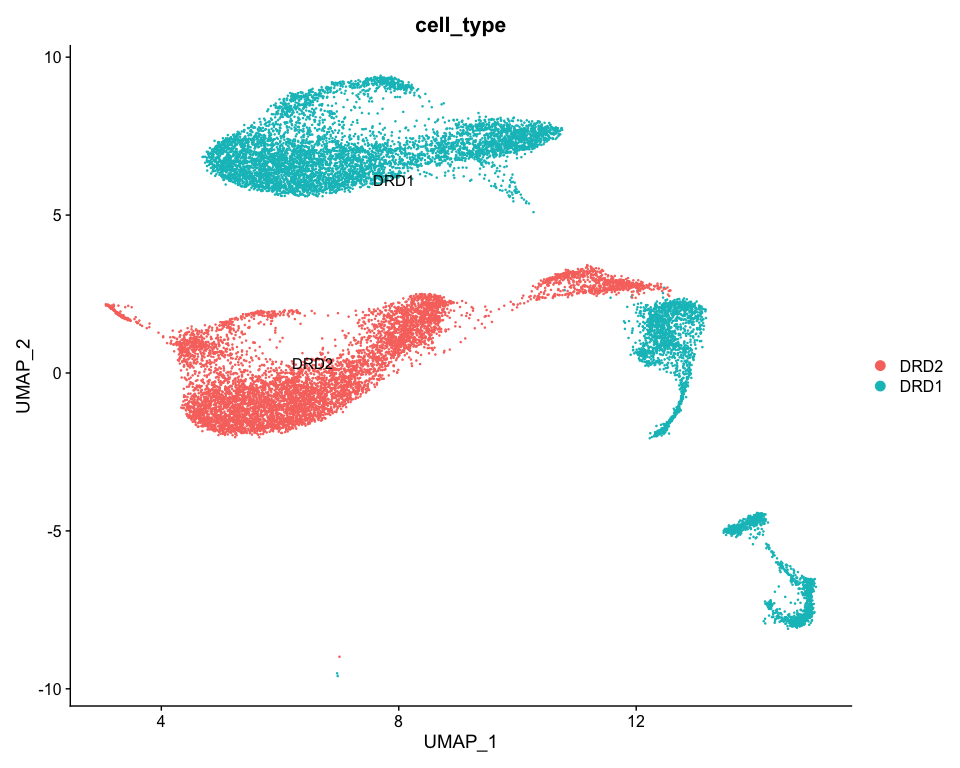
if (!"pca" %in% names(so\_msn@reductions)) {  
 cat("Running PCA on MSNs...\n")  
 so\_msn <- RunPCA(so\_msn, features = VariableFeatures(so\_msn))  
}  
  
p7 <- DimPlot(so\_msn, reduction = "pca", group.by = "cell\_type", label = TRUE, repel = TRUE)  
print(p7)



ggsave("D1vD2\_figures/PCA\_plot\_MSN.png", p7, width = 12, height = 8)  
  
p8 <- ElbowPlot(so\_msn, ndims = 50)  
print(p8)



ggsave("D1vD2\_figures/Elbow\_plot\_MSN.png", p8, width = 10, height = 6)  
  
if (!"umap" %in% names(so\_msn@reductions)) {  
 cat("Running UMAP on MSNs...\n")  
 set.seed(1234)  
 so\_msn <- RunUMAP(so\_msn, dims = 1:20)  
}  
  
p9 <- DimPlot(so\_msn, reduction = "umap", group.by = "cell\_type", label = TRUE, repel = TRUE)  
print(p9)



ggsave("D1vD2\_figures/UMAP\_CellType\_MSN.png", p9, width = 12, height = 10)

## Dopamine Receptor Expression and D1/D2 Classification

# Key dopamine-related markers  
dopamine\_markers <- c("DRD1", "DRD2", "PPP1R1B", "PDE1B", "BCL11B", "KIAA1211L", "PDE2A", "SLIT3", "NGEF")  
  
present\_markers <- dopamine\_markers[dopamine\_markers %in% rownames(so\_msn)]  
missing\_markers <- setdiff(dopamine\_markers, present\_markers)  
  
cat("Dopamine Marker Availability\n")

## Dopamine Marker Availability

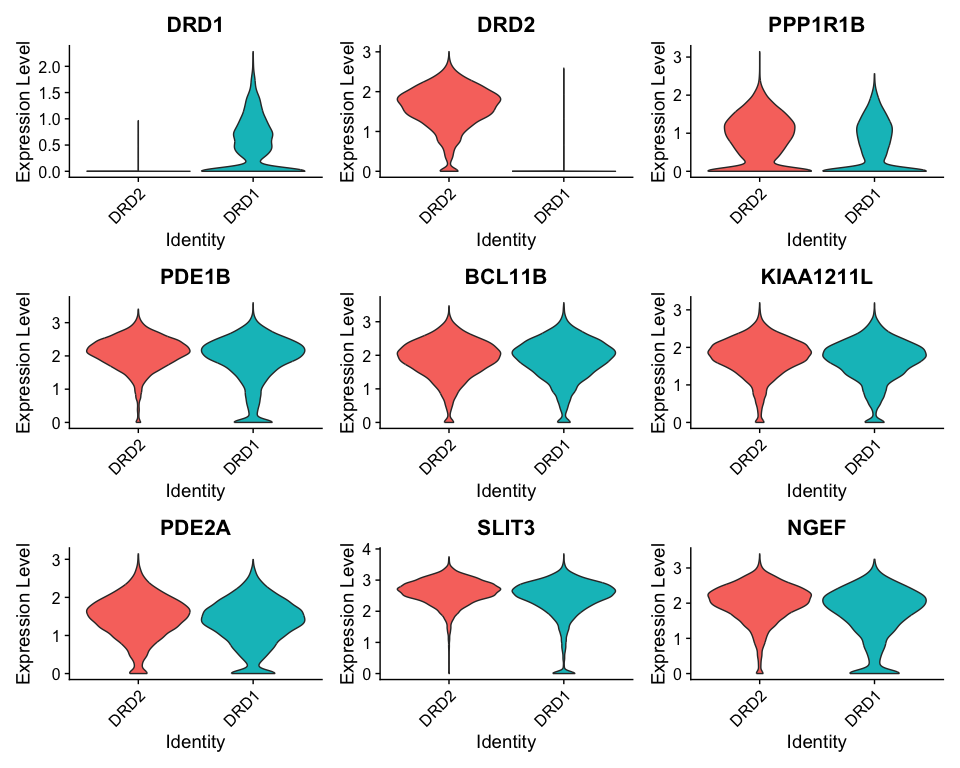
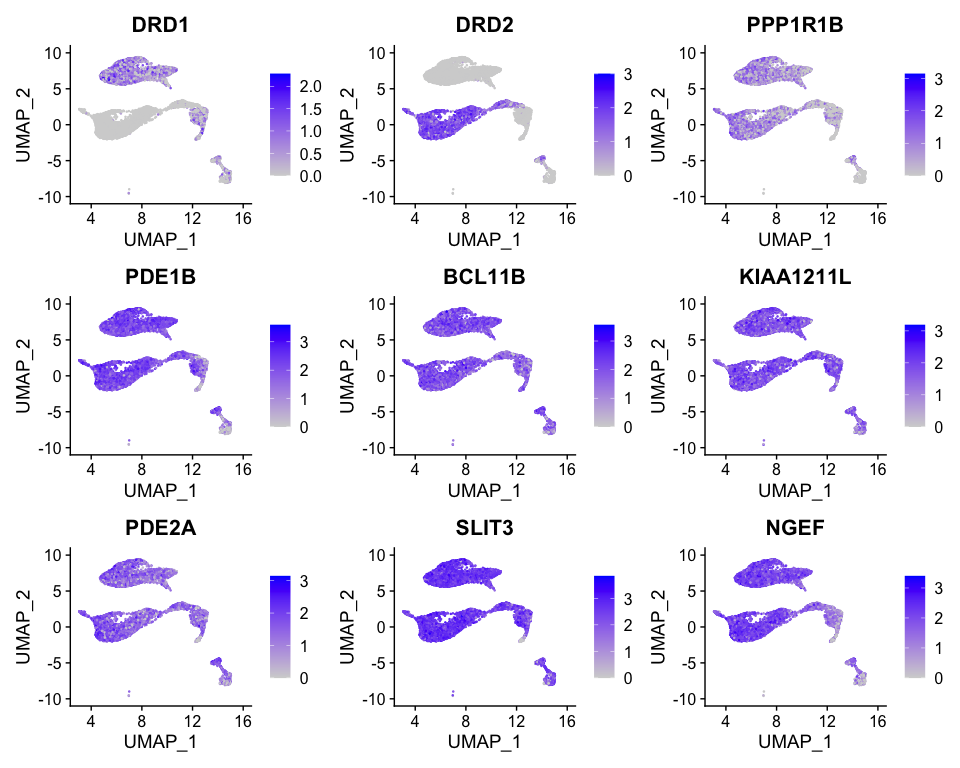
cat(paste("Present:", paste(present\_markers, collapse = ", ")), "\n")

## Present: DRD1, DRD2, PPP1R1B, PDE1B, BCL11B, KIAA1211L, PDE2A, SLIT3, NGEF

cat(paste("Missing:", paste(missing\_markers, collapse = ", ")), "\n")

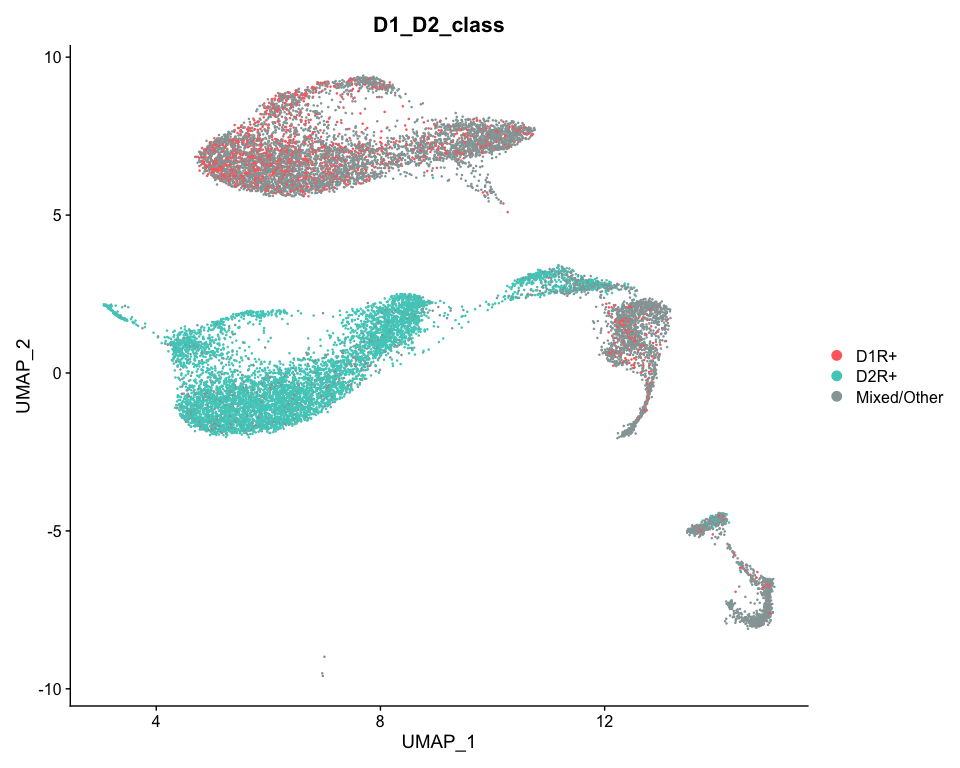
## Missing:

if (length(present\_markers) > 0) {  
 p10 <- FeaturePlot(so\_msn, features = present\_markers, ncol = 3, reduction = "umap")  
 print(p10)  
 ggsave("D1vD2\_figures/Dopamine\_Markers\_FeaturePlot.png", p10, width = 15, height = 10)  
  
 p11 <- VlnPlot(so\_msn, features = present\_markers, group.by = "cell\_type", ncol = 3, pt.size = 0)  
 print(p11)  
 ggsave("D1vD2\_figures/Dopamine\_Markers\_ViolinPlot.png", p11, width = 15, height = 10)  
}



# D1/D2 classification based on DRD1/DRD2 expression  
if (all(c("DRD1", "DRD2") %in% rownames(so\_msn))) {  
 drd1\_exp <- GetAssayData(so\_msn, assay = "RNA", layer = "data")["DRD1",]  
 drd2\_exp <- GetAssayData(so\_msn, assay = "RNA", layer = "data")["DRD2",]  
   
 so\_msn$D1\_D2\_class <- case\_when(  
 drd1\_exp > 1 & drd2\_exp < 0.5 ~ "D1R+",  
 drd2\_exp > 1 & drd1\_exp < 0.5 ~ "D2R+",  
 TRUE ~ "Mixed/Other"  
 )  
   
 cat("D1R/D2R Classification\n")  
 print(table(so\_msn$D1\_D2\_class))  
  
 p12 <- DimPlot(so\_msn, group.by = "D1\_D2\_class", reduction = "umap",  
 cols = c("D1R+" = "#FF6B6B", "D2R+" = "#4ECDC4", "Mixed/Other" = "#95A5A6"))  
 print(p12)  
 ggsave("D1vD2\_figures/D1R\_D2R\_Classification.png", p12, width = 10, height = 8)  
} else {  
 cat("DRD1 or DRD2 not found, using cell\_type for classification\n")  
 if ("cell\_type" %in% colnames(so\_msn@meta.data)) {  
 so\_msn$D1\_D2\_class <- so\_msn$cell\_type  
 }  
}

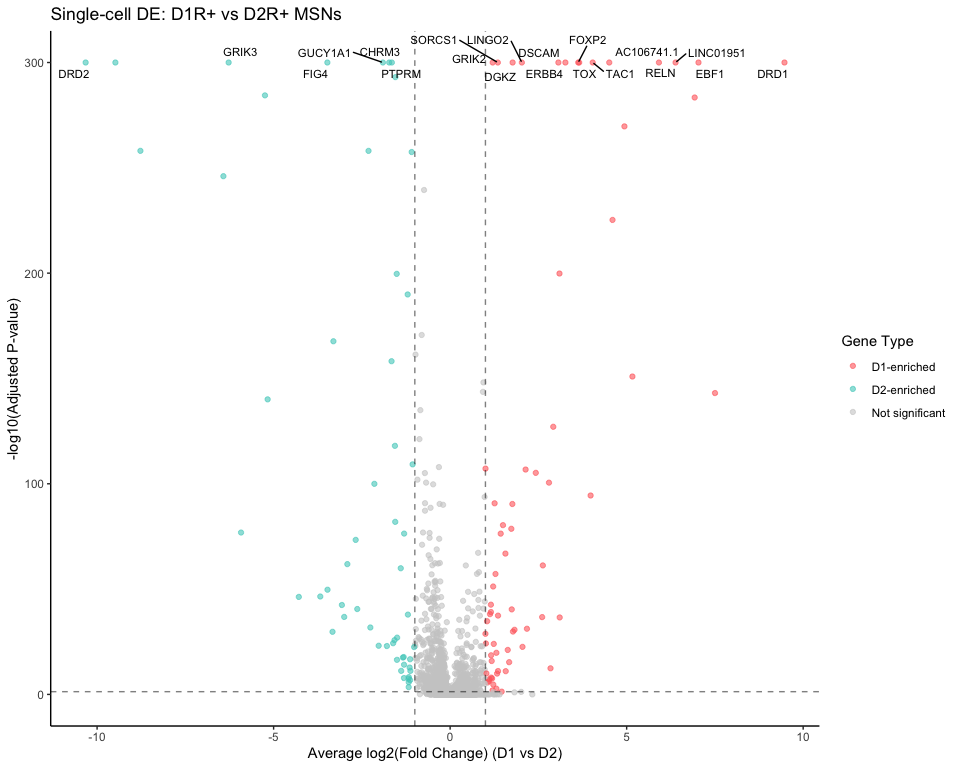
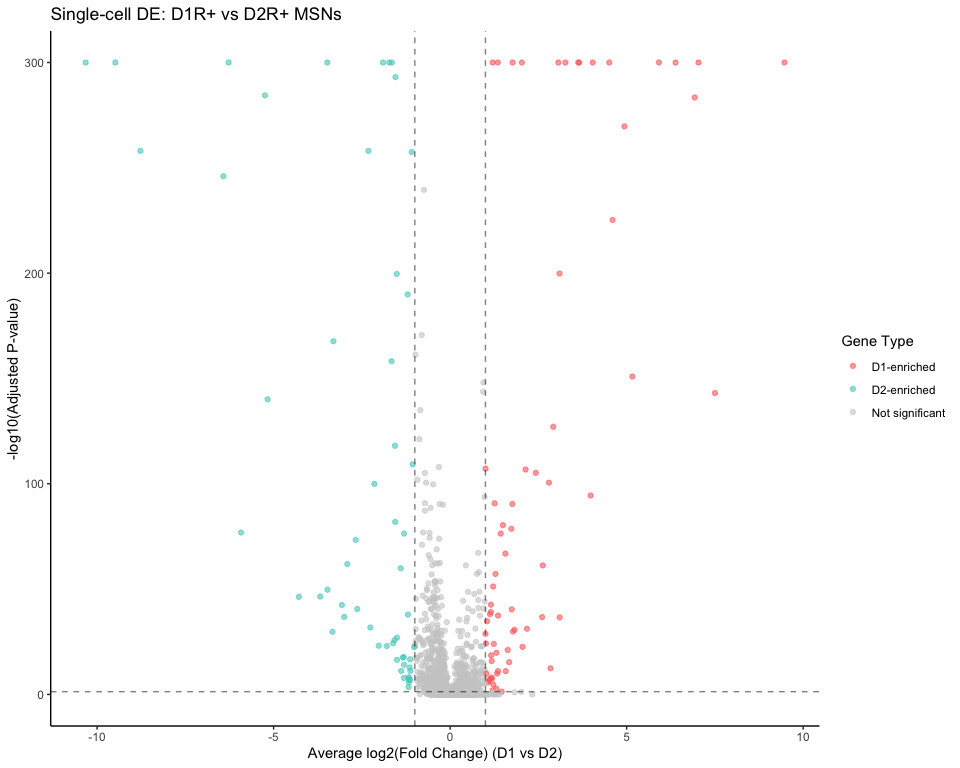
## D1R/D2R Classification  
##   
## D1R+ D2R+ Mixed/Other   
## 1627 6390 8378



## Single-cell Differential Expression Analysis: D1 vs D2

# Ensure we have sufficient cells for DE analysis  
if (ncol(so\_msn) > 50 && "D1\_D2\_class" %in% colnames(so\_msn@meta.data)) {  
 cat("Performing Single-cell Differential Expression Analysis: D1 vs D2\n")  
   
 # Check if we have D1R+ and D2R+ cells  
 cell\_counts <- table(so\_msn$D1\_D2\_class)  
 print(cell\_counts)  
   
 if ("D1R+" %in% names(cell\_counts) && "D2R+" %in% names(cell\_counts) &&   
 cell\_counts["D1R+"] >= 10 && cell\_counts["D2R+"] >= 10) {  
   
 Idents(so\_msn) <- so\_msn$D1\_D2\_class  
   
 cat("Comparing D1R+ vs D2R+ MSNs (single-cell)\n")  
 markers\_d1\_vs\_d2 <- FindMarkers(  
 so\_msn,  
 ident.1 = "D1R+",  
 ident.2 = "D2R+",  
 only.pos = FALSE,  
 min.pct = 0.1,  
 logfc.threshold = 0,  
 test.use = "wilcox"  
 )  
   
 markers\_d1\_vs\_d2$gene <- rownames(markers\_d1\_vs\_d2)  
 write.csv(markers\_d1\_vs\_d2, "D1vD2\_tables/SingleCell\_DE\_D1\_vs\_D2\_all.csv", row.names = FALSE)  
   
 # Apply significance criteria: abs(log2FC) > 1, padj < 0.05  
 sc\_sig <- markers\_d1\_vs\_d2 %>%  
 filter(!is.na(p\_val\_adj)) %>%  
 filter(p\_val\_adj < 0.05, abs(avg\_log2FC) > 1)  
   
 write.csv(sc\_sig, "D1vD2\_tables/SingleCell\_DE\_D1\_vs\_D2\_significant.csv", row.names = FALSE)  
   
 cat("Significant DE genes found (single-cell):", nrow(sc\_sig), "\n")  
   
 # Volcano plot  
 volcano\_data <- markers\_d1\_vs\_d2 %>%  
 mutate(  
 sig = ifelse(!is.na(p\_val\_adj) & p\_val\_adj < 0.05 & abs(avg\_log2FC) > 1, "significant", "ns"),  
 log10\_padj = -log10(p\_val\_adj + 1e-300),  
 direction = case\_when(  
 avg\_log2FC > 1 & p\_val\_adj < 0.05 ~ "D1-enriched",  
 avg\_log2FC < -1 & p\_val\_adj < 0.05 ~ "D2-enriched",  
 TRUE ~ "Not significant"  
 )  
 )  
   
 p\_volcano\_sc <- ggplot(volcano\_data, aes(x = avg\_log2FC, y = log10\_padj)) +  
 geom\_point(aes(color = direction), alpha = 0.6) +  
 scale\_color\_manual(values = c("D1-enriched" = "#FF6B6B", "D2-enriched" = "#4ECDC4", "Not significant" = "#CCCCCC")) +  
 labs(x = "Average log2(Fold Change) (D1 vs D2)",   
 y = "-log10(Adjusted P-value)",  
 title = "Single-cell DE: D1R+ vs D2R+ MSNs",  
 color = "Gene Type") +  
 theme\_classic() +  
 geom\_hline(yintercept = -log10(0.05), linetype = "dashed", alpha = 0.5) +  
 geom\_vline(xintercept = c(-1, 1), linetype = "dashed", alpha = 0.5)  
   
 print(p\_volcano\_sc)  
 ggsave("D1vD2\_figures/Volcano\_Plot\_SingleCell\_D1\_vs\_D2.png", p\_volcano\_sc, width = 10, height = 8)  
   
 # Highlight top genes  
 top\_genes <- sc\_sig %>%   
 arrange(p\_val\_adj) %>%   
 slice\_head(n = 20) %>%   
 pull(gene)  
   
 p\_volcano\_sc\_labeled <- p\_volcano\_sc +  
 geom\_text\_repel(  
 data = volcano\_data %>% filter(gene %in% top\_genes),  
 aes(label = gene),  
 size = 3, max.overlaps = 20  
 )  
   
 print(p\_volcano\_sc\_labeled)  
 ggsave("D1vD2\_figures/Volcano\_Plot\_SingleCell\_D1\_vs\_D2\_labeled.png", p\_volcano\_sc\_labeled, width = 12, height = 10)  
   
 } else {  
 cat("Insufficient D1R+ or D2R+ cells for single-cell DE analysis\n")  
 }  
} else {  
 cat("Insufficient cells for single-cell DE analysis\n")  
}

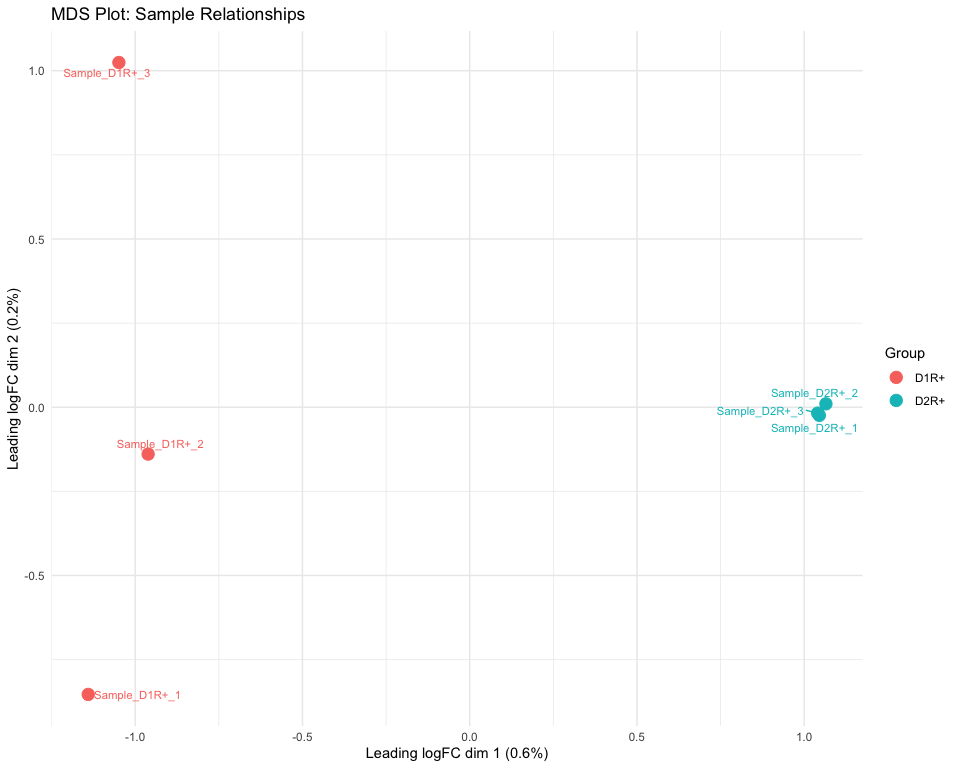
## Performing Single-cell Differential Expression Analysis: D1 vs D2  
##   
## D1R+ D2R+ Mixed/Other   
## 1627 6390 8378   
## Comparing D1R+ vs D2R+ MSNs (single-cell)  
## Significant DE genes found (single-cell): 125



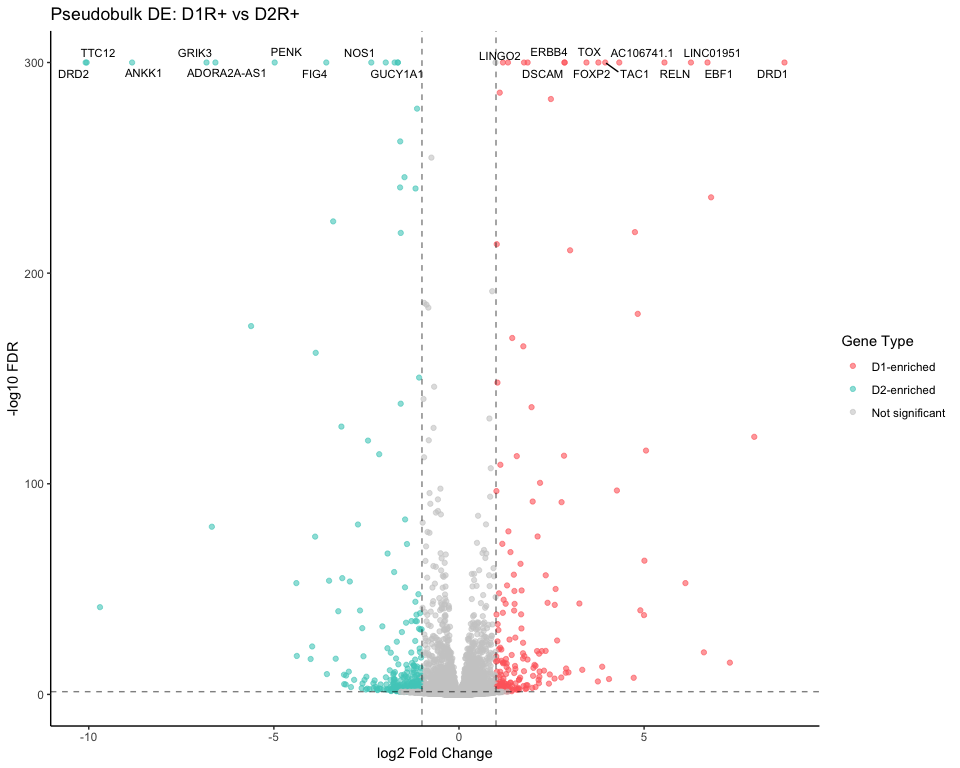
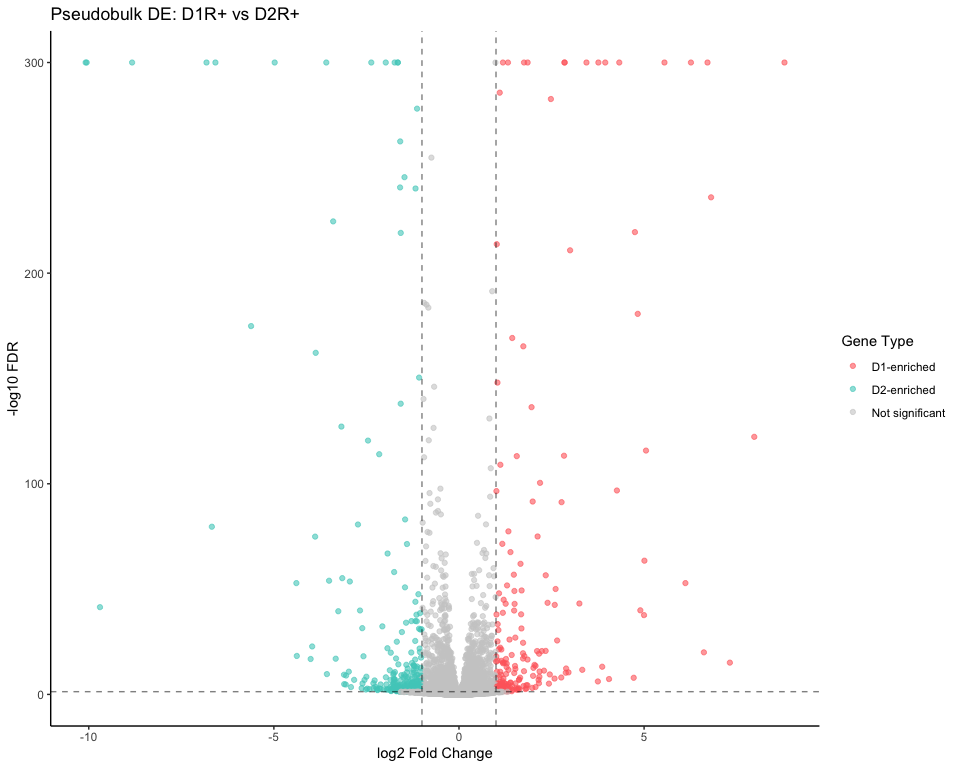
## Pseudobulk Differential Expression Analysis with MDS Plot

if ("D1\_D2\_class" %in% colnames(so\_msn@meta.data)) {  
 cat("Performing Pseudobulk Differential Expression Analysis\n")  
   
 # Create sample IDs if not present (simulate biological replicates)  
 if (!"sample" %in% colnames(so\_msn@meta.data)) {  
 set.seed(123)  
 so\_msn$sample <- NA  
   
 # Create 3 pseudo-replicates for each D1/D2 class  
 for (class in unique(so\_msn$D1\_D2\_class)) {  
 if (class %in% c("D1R+", "D2R+")) {  
 class\_cells <- which(so\_msn$D1\_D2\_class == class)  
 n\_samples <- 3  
 so\_msn$sample[class\_cells] <- paste0("Sample\_", class, "\_", sample(1:n\_samples, length(class\_cells), replace = TRUE))  
 }  
 }  
 }  
   
 # Get counts and metadata  
 counts <- GetAssayData(so\_msn, slot = "counts")  
 meta\_df <- so\_msn@meta.data[, c("sample", "D1\_D2\_class")]  
   
 # Create pseudobulk by aggregating counts per sample  
 pb\_counts\_list <- list()  
 pb\_meta\_list <- list()  
   
 for (s in unique(meta\_df$sample)) {  
 if (!is.na(s)) {  
 cells <- rownames(meta\_df[meta\_df$sample == s & !is.na(meta\_df$sample), ])  
 group <- unique(meta\_df[meta\_df$sample == s & !is.na(meta\_df$sample), "D1\_D2\_class"])  
   
 if (length(group) == 1 && length(cells) > 0) {  
 pb\_counts\_list[[s]] <- Matrix::rowSums(counts[, cells, drop = FALSE])  
 pb\_meta\_list[[s]] <- data.frame(sample = s, group = group)  
 }  
 }  
 }  
   
 pb\_counts <- do.call(cbind, pb\_counts\_list)  
 pb\_meta <- do.call(rbind, pb\_meta\_list)  
   
 # Filter for D1R+ and D2R+ only  
 keep\_groups <- c("D1R+", "D2R+")  
 keep\_samples <- pb\_meta$group %in% keep\_groups  
 pb\_counts <- pb\_counts[, keep\_samples]  
 pb\_meta <- pb\_meta[keep\_samples, , drop = FALSE]  
 pb\_meta$group <- factor(pb\_meta$group, levels = keep\_groups)  
   
 cat(paste("Pseudobulk samples:", ncol(pb\_counts)), "\n")  
 print(table(pb\_meta$group))  
   
 if (ncol(pb\_counts) >= 4 && all(keep\_groups %in% pb\_meta$group)) {  
 # EdgeR analysis  
 dge <- DGEList(counts = pb\_counts, group = pb\_meta$group)  
   
 # Filter lowly expressed genes  
 keep <- filterByExpr(dge, min.count = 10)  
 dge <- dge[keep, , keep.lib.sizes = FALSE]  
   
 # Normalization  
 dge <- calcNormFactors(dge)  
   
 # MDS Plot for sample relationships  
 cat("Creating MDS plot for sample relationships\n")  
 mds\_data <- plotMDS(dge, plot = FALSE)  
 mds\_df <- data.frame(  
 Sample = colnames(dge),  
 Dim1 = mds\_data$x,  
 Dim2 = mds\_data$y,  
 Group = pb\_meta$group  
 )  
   
 p\_mds <- ggplot(mds\_df, aes(x = Dim1, y = Dim2, color = Group)) +  
 geom\_point(size = 4) +  
 geom\_text\_repel(aes(label = Sample), size = 3) +  
 labs(title = "MDS Plot: Sample Relationships",  
 x = paste("Leading logFC dim 1 (", round(mds\_data$var.explained[1], 1), "%)", sep = ""),  
 y = paste("Leading logFC dim 2 (", round(mds\_data$var.explained[2], 1), "%)", sep = "")) +  
 theme\_minimal()  
   
 print(p\_mds)  
 ggsave("D1vD2\_figures/MDS\_Sample\_Relationships.png", p\_mds, width = 10, height = 8)  
   
 # Design matrix  
 design <- model.matrix(~0 + pb\_meta$group)  
 colnames(design) <- make.names(levels(pb\_meta$group))  
   
 # Estimate dispersion  
 dge <- estimateDisp(dge, design)  
   
 # Fit model  
 fit <- glmFit(dge, design)  
   
 # Make contrast  
 contrast <- makeContrasts(D1R\_vs\_D2R = D1R. - D2R., levels = design)  
 lrt <- glmLRT(fit, contrast = contrast)  
   
 # Extract results  
 pb\_de\_results <- topTags(lrt, n = Inf)$table %>%  
 rownames\_to\_column("gene") %>%  
 arrange(FDR)  
   
 write.csv(pb\_de\_results, "D1vD2\_tables/Pseudobulk\_DE\_D1\_vs\_D2\_all.csv", row.names = FALSE)  
   
 # Apply significance criteria: abs(logFC) > 1 & FDR < 0.05  
 pb\_sig <- pb\_de\_results %>%   
 filter(FDR < 0.05, abs(logFC) > 1)  
   
 write.csv(pb\_sig, "D1vD2\_tables/Pseudobulk\_DE\_D1\_vs\_D2\_significant.csv", row.names = FALSE)  
   
 cat("Significant DE genes found (pseudobulk):", nrow(pb\_sig), "\n")  
   
 # Volcano plot (pseudobulk)  
 pb\_volcano\_data <- pb\_de\_results %>%  
 mutate(  
 log10FDR = -log10(FDR + 1e-300),  
 direction = case\_when(  
 logFC > 1 & FDR < 0.05 ~ "D1-enriched",  
 logFC < -1 & FDR < 0.05 ~ "D2-enriched",  
 TRUE ~ "Not significant"  
 )  
 )  
   
 p\_volcano\_pb <- ggplot(pb\_volcano\_data, aes(x = logFC, y = log10FDR)) +  
 geom\_point(aes(color = direction), alpha = 0.6) +  
 scale\_color\_manual(values = c("D1-enriched" = "#FF6B6B", "D2-enriched" = "#4ECDC4", "Not significant" = "#CCCCCC")) +  
 labs(title = "Pseudobulk DE: D1R+ vs D2R+",  
 x = "log2 Fold Change", y = "-log10 FDR", color = "Gene Type") +  
 theme\_classic() +  
 geom\_hline(yintercept = -log10(0.05), linetype = "dashed", alpha = 0.5) +  
 geom\_vline(xintercept = c(-1, 1), linetype = "dashed", alpha = 0.5)  
   
 print(p\_volcano\_pb)  
 ggsave("D1vD2\_figures/Volcano\_Plot\_Pseudobulk\_D1\_vs\_D2.png", p\_volcano\_pb, width = 10, height = 8)  
   
 # Label top genes  
 pb\_top\_genes <- pb\_sig %>%   
 arrange(FDR) %>%   
 slice\_head(n = 20) %>%   
 pull(gene)  
   
 p\_volcano\_pb\_labeled <- p\_volcano\_pb +  
 geom\_text\_repel(  
 data = pb\_volcano\_data %>% filter(gene %in% pb\_top\_genes),  
 aes(label = gene),  
 size = 3, max.overlaps = 20  
 )  
   
 print(p\_volcano\_pb\_labeled)  
 ggsave("D1vD2\_figures/Volcano\_Plot\_Pseudobulk\_D1\_vs\_D2\_labeled.png", p\_volcano\_pb\_labeled, width = 12, height = 10)  
   
 } else {  
 cat("Insufficient samples for pseudobulk analysis\n")  
 }  
} else {  
 cat("D1\_D2\_class not found for pseudobulk analysis\n")  
}

## Performing Pseudobulk Differential Expression Analysis  
## Pseudobulk samples: 6   
##   
## D1R+ D2R+   
## 3 3   
## Creating MDS plot for sample relationships



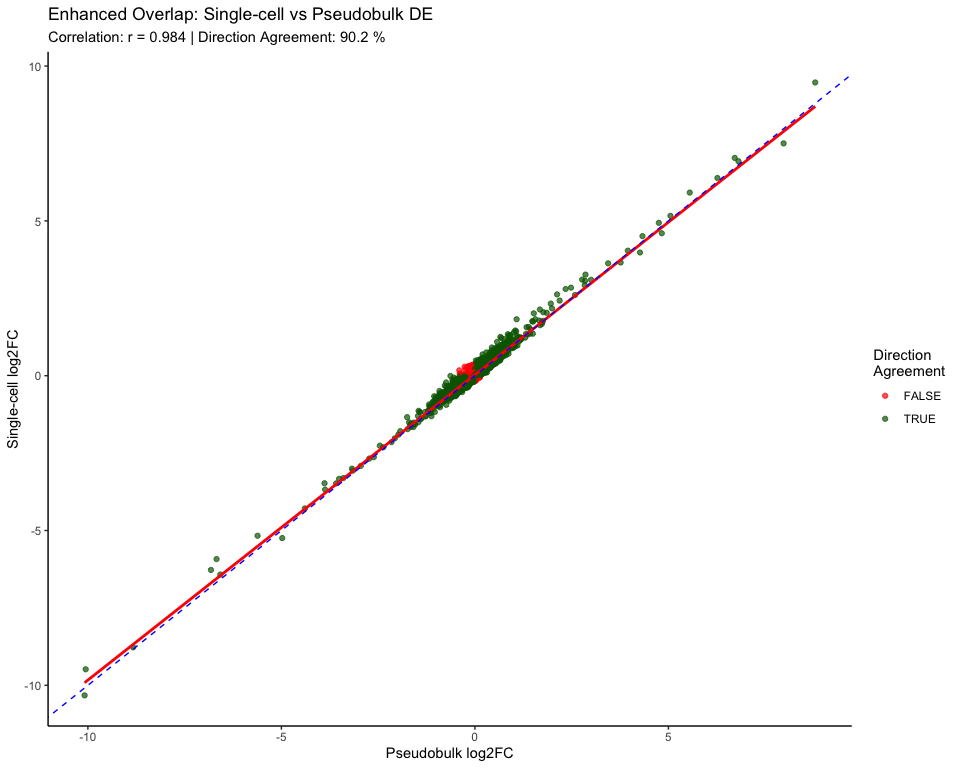
## Significant DE genes found (pseudobulk): 589



## Enhanced Overlapping DE Analysis: Single-cell vs Pseudobulk

if (exists("markers\_d1\_vs\_d2") && exists("pb\_de\_results")) {  
 cat("Performing Enhanced Overlap Analysis: Single-cell vs Pseudobulk\n")  
   
 sc\_genes <- rownames(markers\_d1\_vs\_d2)  
 pb\_genes <- pb\_de\_results$gene  
 overlapping\_genes <- intersect(sc\_genes, pb\_genes)  
   
 cat("Number of overlapping DE genes:", length(overlapping\_genes), "\n")  
   
 if (length(overlapping\_genes) > 0) {  
 # Create overlap comparison  
 overlap\_sc <- markers\_d1\_vs\_d2[overlapping\_genes, , drop = FALSE] %>%  
 rownames\_to\_column("sc\_gene") %>% # Rename to avoid duplication  
 select(sc\_gene, avg\_log2FC, p\_val\_adj) %>%  
 rename(sc\_log2FC = avg\_log2FC, sc\_padj = p\_val\_adj)  
   
 overlap\_pb <- pb\_de\_results %>%   
 filter(gene %in% overlapping\_genes) %>%  
 select(gene, logFC, FDR) %>%  
 rename(pb\_log2FC = logFC, pb\_FDR = FDR)  
   
 overlap\_merged <- merge(overlap\_sc, overlap\_pb, by.x = "sc\_gene", by.y = "gene") # Merge on the new column name  
   
 # Calculate agreement metrics  
 overlap\_merged$direction\_agreement <- sign(overlap\_merged$sc\_log2FC) == sign(overlap\_merged$pb\_log2FC)  
 overlap\_merged$abs\_diff <- abs(overlap\_merged$pb\_log2FC - overlap\_merged$sc\_log2FC)  
   
 # Calculate correlation  
 cor\_coef <- cor(overlap\_merged$sc\_log2FC, overlap\_merged$pb\_log2FC, use = "complete.obs")  
 direction\_agreement\_rate <- mean(overlap\_merged$direction\_agreement, na.rm = TRUE)  
   
 cat(paste("Correlation between single-cell and pseudobulk log2FC:", round(cor\_coef, 3)), "\n")  
 cat(paste("Direction Agreement:", round(direction\_agreement\_rate \* 100, 1), "%\n"))  
   
 write.csv(overlap\_merged, "D1vD2\_tables/Overlapping\_DEGs\_SC\_vs\_PB\_enhanced.csv", row.names = FALSE)  
   
 # Enhanced comparison plot  
 p\_overlap\_enhanced <- ggplot(overlap\_merged, aes(x = pb\_log2FC, y = sc\_log2FC)) +  
 geom\_point(aes(color = direction\_agreement), alpha = 0.7) +  
 geom\_smooth(method = "lm", se = TRUE, color = "red") +  
 geom\_abline(intercept = 0, slope = 1, linetype = "dashed", color = "blue") +  
 scale\_color\_manual(values = c("TRUE" = "darkgreen", "FALSE" = "red"),   
 name = "Direction\nAgreement") +  
 labs(x = "Pseudobulk log2FC", y = "Single-cell log2FC",  
 title = "Enhanced Overlap: Single-cell vs Pseudobulk DE",  
 subtitle = paste("Correlation: r =", round(cor\_coef, 3),   
 "| Direction Agreement:", round(direction\_agreement\_rate \* 100, 1), "%")) +  
 theme\_classic() +  
 theme(legend.position = "right")  
   
 print(p\_overlap\_enhanced)  
 ggsave("D1vD2\_figures/Enhanced\_Overlap\_DEGs\_Comparison.png", p\_overlap\_enhanced, width = 12, height = 8)  
   
 # Identify top disagreements and agreements  
 overlap\_merged <- overlap\_merged[order(-overlap\_merged$abs\_diff), ]  
   
 # Top disagreements where direction is different or abs\_diff > 0.1  
 top\_disagreements <- overlap\_merged %>%  
 filter(direction\_agreement == FALSE | abs\_diff > 0.1) %>%  
 arrange(-abs\_diff) %>%  
 head(10)  
   
 # Top agreements (same direction, smallest abs\_diff)  
 top\_agreements <- overlap\_merged %>%  
 filter(direction\_agreement == TRUE) %>%  
 arrange(abs\_diff) %>%  
 head(10)  
  
 # Check if there are any results for disagreements and agreements  
 cat("Top Disagreements (largest log2FC differences):\n")  
 if (nrow(top\_disagreements) > 0) {  
 print(top\_disagreements[, c("sc\_gene", "sc\_log2FC", "pb\_log2FC", "abs\_diff")])  
 } else {  
 cat("No significant disagreements found.\n")  
 }  
  
 cat("Top Agreements (smallest log2FC differences with same direction):\n")  
 if (nrow(top\_agreements) > 0) {  
 print(top\_agreements[, c("sc\_gene", "sc\_log2FC", "pb\_log2FC", "abs\_diff")])  
 } else {  
 cat("No significant agreements found.\n")  
 }  
   
 # Save top disagreements and agreements to CSV  
 write.csv(top\_disagreements, "D1vD2\_tables/Top\_Disagreements\_SC\_vs\_PB.csv", row.names = FALSE)  
 write.csv(top\_agreements, "D1vD2\_tables/Top\_Agreements\_SC\_vs\_PB.csv", row.names = FALSE)  
   
 # Store concordance rate for later use  
 concordance\_rate <- direction\_agreement\_rate  
   
 } else {  
 cat("No overlapping genes found between methods\n")  
 }  
} else {  
 cat("Single-cell or pseudobulk DE results not found for overlap analysis\n")  
}

## Performing Enhanced Overlap Analysis: Single-cell vs Pseudobulk  
## Number of overlapping DE genes: 10427   
## Correlation between single-cell and pseudobulk log2FC: 0.984   
## Direction Agreement: 90.2 %



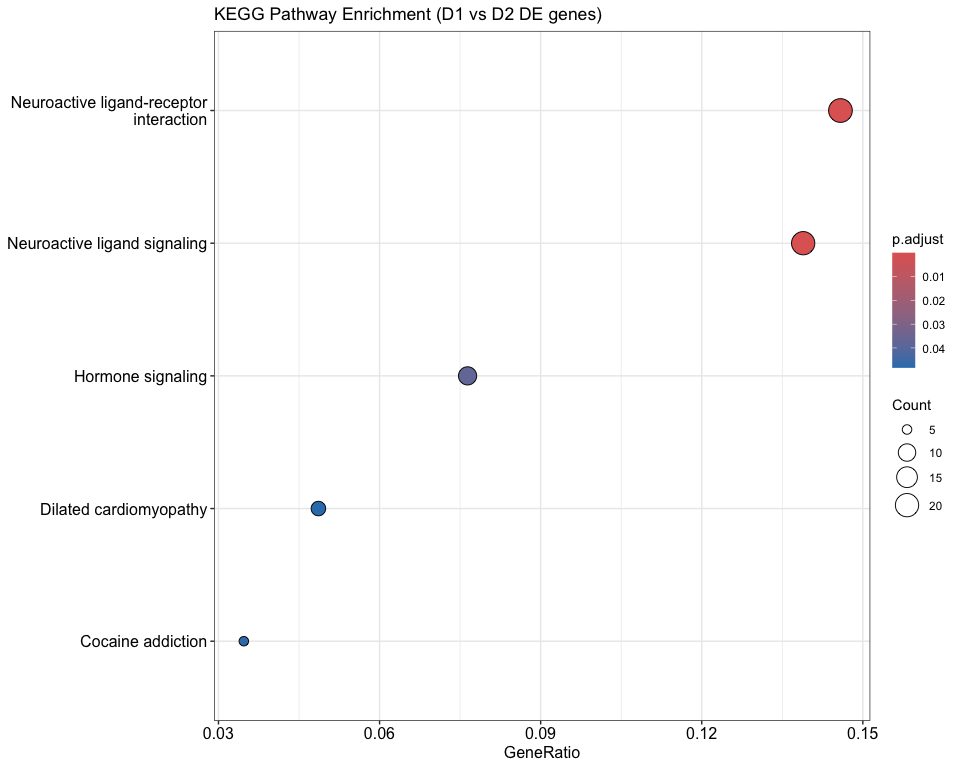
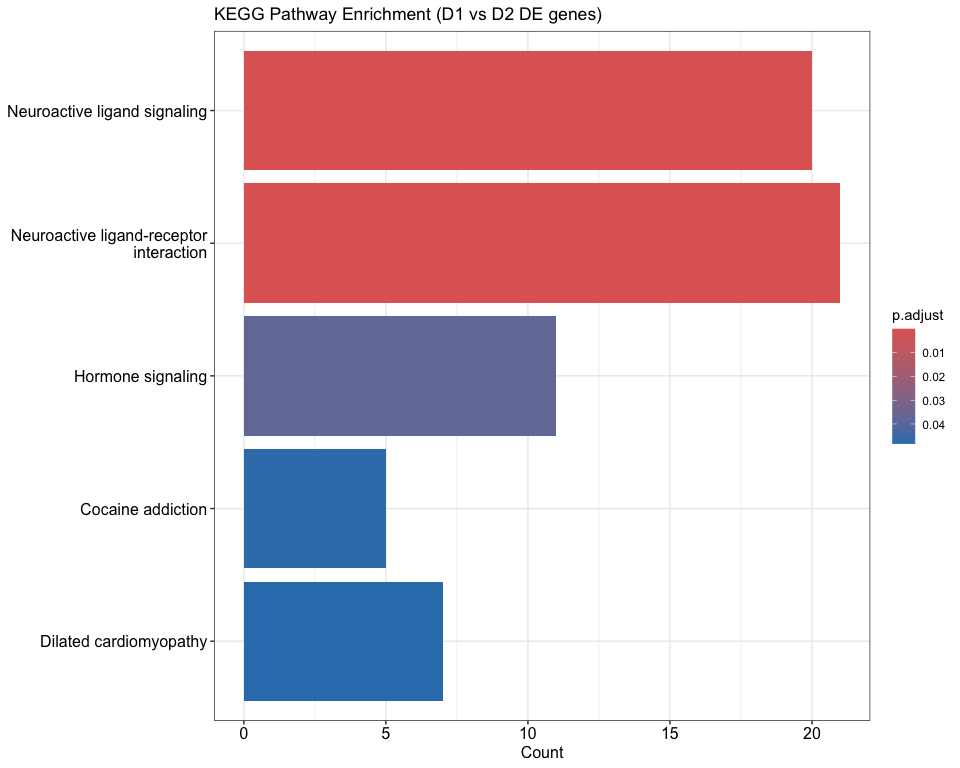
## Top Disagreements (largest log2FC differences):  
## sc\_gene sc\_log2FC pb\_log2FC abs\_diff  
## 1 GPR6 -5.922201828 -6.6756967 0.7534949  
## 2 TRPC5 1.822567807 1.0779711 0.7445967  
## 3 DRD1 9.470878130 8.7922464 0.6786318  
## 4 COL19A1 -0.006460325 -0.6323054 0.6258451  
## 5 KCNT2 1.251112518 0.6574067 0.5937058  
## 6 INSYN2B 0.173730633 -0.4040160 0.5777466  
## 7 TTC12 -9.481807476 -10.0563743 0.5745669  
## 8 ZFPM2 0.288899630 -0.2666497 0.5555494  
## 9 GRIK3 -6.273797199 -6.8200779 0.5462807  
## 10 NECTIN3 1.218566549 0.6844740 0.5340925  
## Top Agreements (smallest log2FC differences with same direction):  
## sc\_gene sc\_log2FC pb\_log2FC abs\_diff  
## 1 ZNRF3 0.01811171 0.01811638 4.673117e-06  
## 2 PSMC2 -0.07754751 -0.07752504 2.247036e-05  
## 3 TSEN2 -0.05884294 -0.05887364 3.069474e-05  
## 4 PCCA -0.03702881 -0.03706063 3.181229e-05  
## 5 SMG9 0.09594193 0.09597644 3.450377e-05  
## 6 BCAR3 -0.13402978 -0.13406475 3.497528e-05  
## 7 MED15 0.09707739 0.09704092 3.647150e-05  
## 8 DNTTIP2 0.06309804 0.06305908 3.895951e-05  
## 9 AC084125.4 0.14131127 0.14126850 4.276631e-05  
## 10 C1orf52 0.19999862 0.20004623 4.761306e-05

## KEGG and GO Enrichment Analysis

if (exists("markers\_d1\_vs\_d2") && nrow(markers\_d1\_vs\_d2) > 100) {  
 cat("Running KEGG and GO Enrichment Analysis\n")  
   
 # Get significant DE genes for enrichment  
 sig\_genes\_for\_enrichment <- markers\_d1\_vs\_d2 %>%  
 filter(!is.na(p\_val\_adj), p\_val\_adj < 0.05, abs(avg\_log2FC) > 0.5)  
  
 # Remove existing 'gene' column if it exists  
 if ("gene" %in% colnames(sig\_genes\_for\_enrichment)) {  
 sig\_genes\_for\_enrichment <- sig\_genes\_for\_enrichment %>%  
 select(-gene)  
 }  
  
 # Add row names as 'gene' column  
 sig\_genes\_for\_enrichment <- sig\_genes\_for\_enrichment %>%  
 rownames\_to\_column("gene")  
  
 if (nrow(sig\_genes\_for\_enrichment) > 10) {  
 sig\_gene\_symbols <- sig\_genes\_for\_enrichment$gene  
   
 # Convert gene symbols to Entrez IDs for KEGG  
 gene\_entrez <- bitr(sig\_gene\_symbols,   
 fromType = "SYMBOL",   
 toType = "ENTREZID",   
 OrgDb = org.Hs.eg.db)  
   
 cat(paste("Genes for enrichment analysis:", length(sig\_gene\_symbols)), "\n")  
 cat(paste("Successfully converted to Entrez ID:", nrow(gene\_entrez)), "\n")  
   
 if (nrow(gene\_entrez) > 10) {  
   
 # 1. KEGG Pathway Enrichment  
 cat("Running KEGG enrichment...\n")  
 kegg\_enrich <- enrichKEGG(gene = gene\_entrez$ENTREZID,  
 organism = 'hsa',  
 pvalueCutoff = 0.05,  
 qvalueCutoff = 0.2)  
   
 if (!is.null(kegg\_enrich) && nrow(as.data.frame(kegg\_enrich)) > 0) {  
 kegg\_results <- as.data.frame(kegg\_enrich)  
 write.csv(kegg\_results, "D1vD2\_tables/KEGG\_Enrichment\_D1\_vs\_D2.csv", row.names = FALSE)  
   
 # KEGG barplot  
 p\_kegg\_bar <- barplot(kegg\_enrich,   
 showCategory = 15,   
 title = "KEGG Pathway Enrichment (D1 vs D2 DE genes)")  
 print(p\_kegg\_bar)  
 ggsave("D1vD2\_figures/KEGG\_Enrichment\_Barplot.png", p\_kegg\_bar, width = 12, height = 8)  
   
 # KEGG dotplot  
 p\_kegg\_dot <- dotplot(kegg\_enrich,   
 showCategory = 15,  
 title = "KEGG Pathway Enrichment (D1 vs D2 DE genes)")  
 print(p\_kegg\_dot)  
 ggsave("D1vD2\_figures/KEGG\_Enrichment\_Dotplot.png", p\_kegg\_dot, width = 12, height = 8)  
   
 cat("KEGG enrichment completed. Found", nrow(kegg\_results), "significant pathways\n")  
 print(head(kegg\_results[, c("Description", "pvalue", "p.adjust", "Count")], 10))  
 } else {  
 cat("No significant KEGG pathways found\n")  
 }  
  
 # (Continue with the rest of the code...)  
 } else {  
 cat("Insufficient genes with Entrez IDs for enrichment analysis\n")  
 }  
 } else {  
 cat("Insufficient significant DE genes for enrichment analysis\n")  
 }  
} else {  
 cat("DE results not available for enrichment analysis\n")  
}

## Running KEGG and GO Enrichment Analysis

## Genes for enrichment analysis: 391   
## Successfully converted to Entrez ID: 315   
## Running KEGG enrichment...



## KEGG enrichment completed. Found 5 significant pathways  
## Description pvalue p.adjust  
## hsa04082 Neuroactive ligand signaling 1.781742e-11 3.955467e-09  
## hsa04080 Neuroactive ligand-receptor interaction 1.748346e-07 1.940664e-05  
## hsa04081 Hormone signaling 5.105906e-04 3.778370e-02  
## hsa05030 Cocaine addiction 8.563131e-04 4.752538e-02  
## hsa05414 Dilated cardiomyopathy 1.088082e-03 4.831086e-02  
## Count  
## hsa04082 20  
## hsa04080 21  
## hsa04081 11  
## hsa05030 5  
## hsa05414 7

## Epigenetic Regulator Expression Analysis

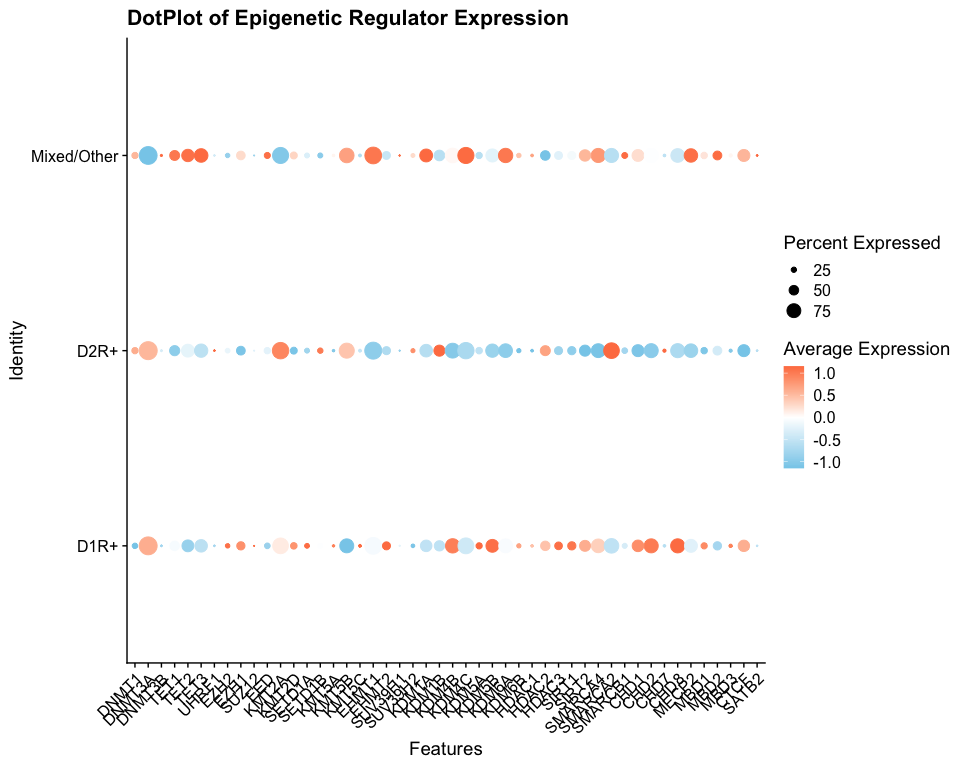
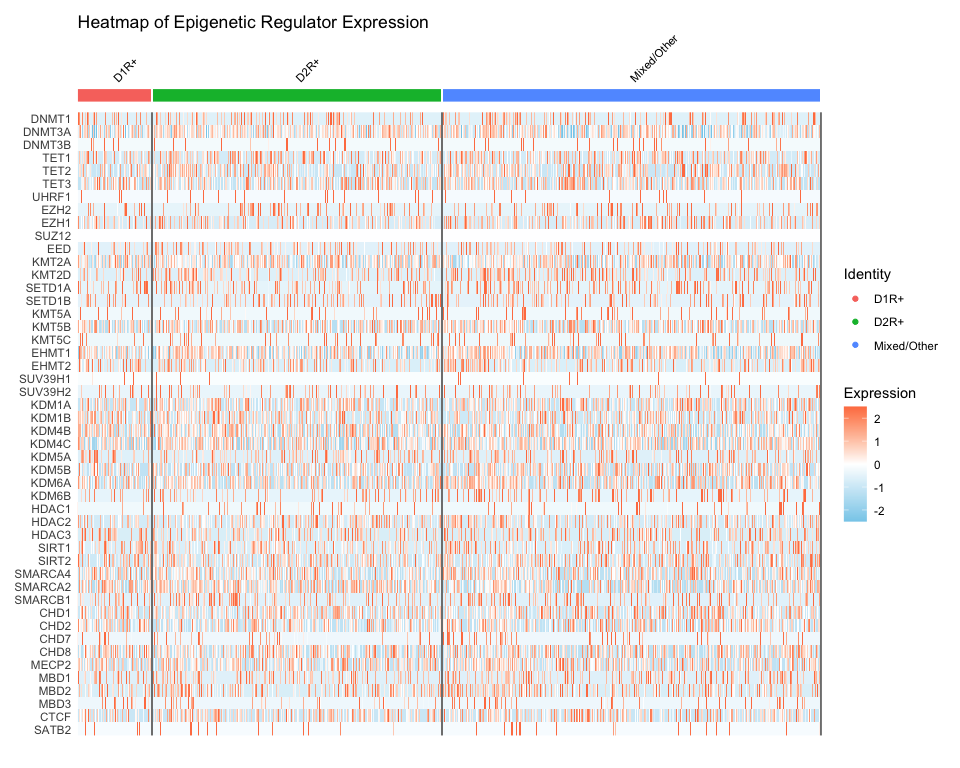
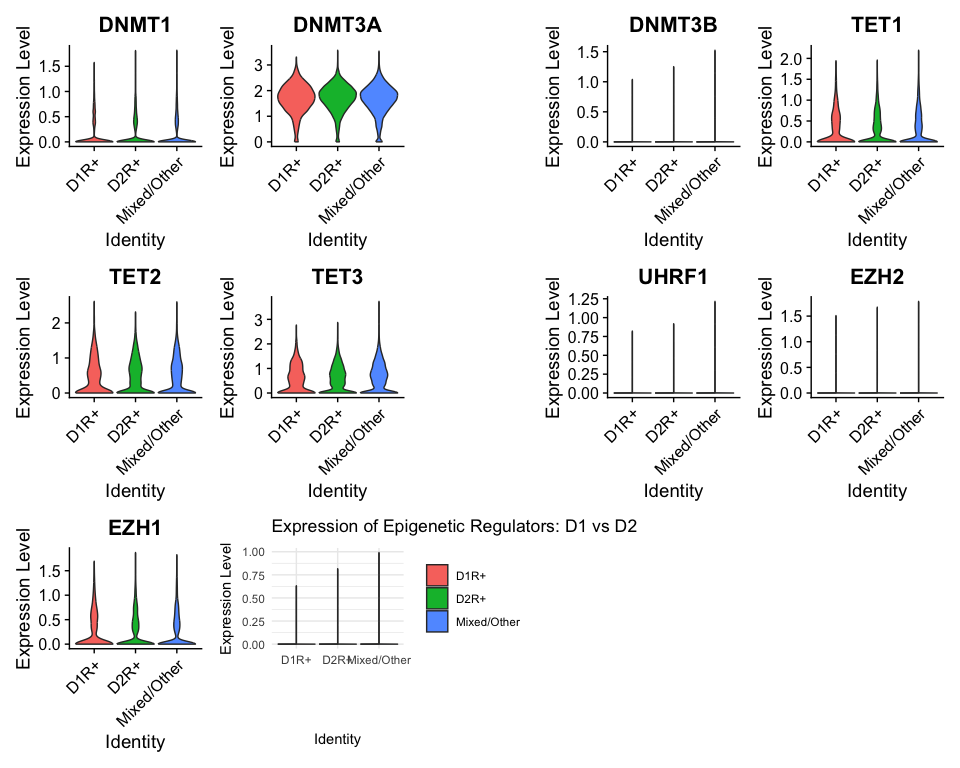
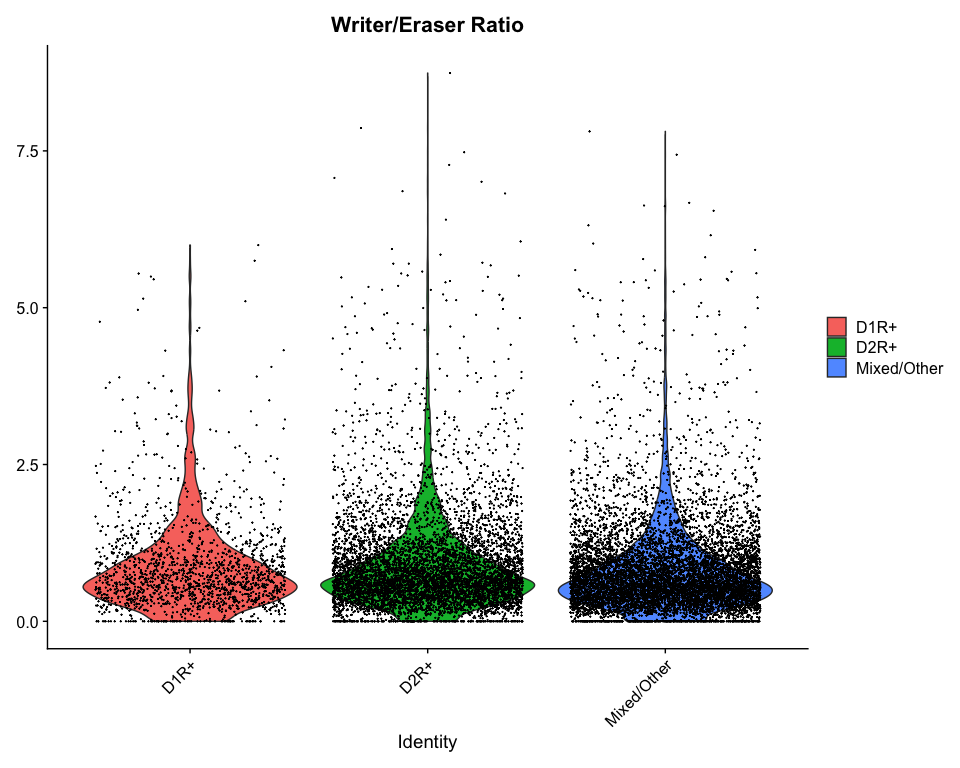
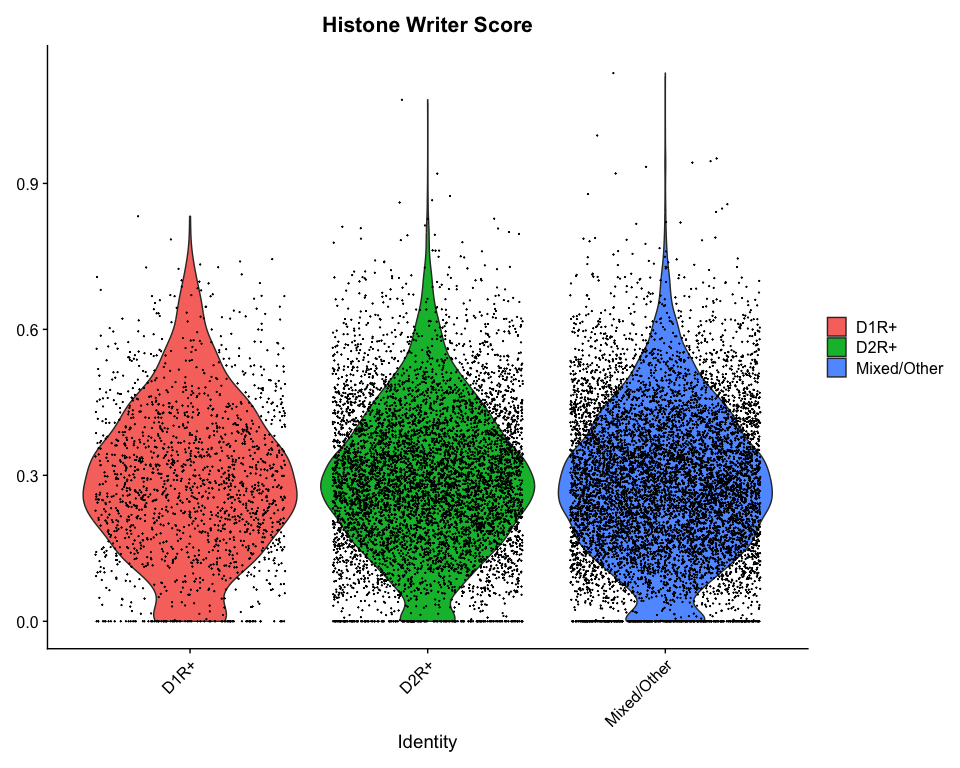
epi\_genes <- c(  
 # DNA methylation  
 "DNMT1","DNMT3A","DNMT3B","TET1","TET2","TET3","UHRF1",  
 # Histone writers  
 "EZH2","EZH1","SUZ12","EED","KMT2A","KMT2D","SETD1A","SETD1B",  
 "KMT5A","KMT5B","KMT5C","EHMT1","EHMT2","SUV39H1","SUV39H2",  
 # Histone erasers  
 "KDM1A","KDM1B","KDM4A","KDM4B","KDM4C","KDM5A","KDM5B","KDM6A","KDM6B",  
 "HDAC1","HDAC2","HDAC3","SIRT1","SIRT2","SIRT6",  
 # Chromatin remodeling  
 "SMARCA4","SMARCA2","SMARCB1","CHD1","CHD2","CHD7","CHD8",  
 # DNA binding/regulation  
 "MECP2","MBD1","MBD2","MBD3","CTCF","SATB1","SATB2"  
)  
  
present\_epi\_genes <- intersect(epi\_genes, rownames(so\_msn))  
missing\_epi\_genes <- setdiff(epi\_genes, present\_epi\_genes)  
  
cat(paste("Present epigenetic genes:", length(present\_epi\_genes)), "\n")

## Present epigenetic genes: 48

cat(paste("Missing epigenetic genes:", length(missing\_epi\_genes)), "\n")

## Missing epigenetic genes: 3

if (length(present\_epi\_genes) > 0) {  
 # Calculate epigenetic scores  
 histone\_writers <- intersect(c("EZH2", "KMT2A", "KMT2D", "SETD1A", "SETD1B"), rownames(so\_msn))  
 histone\_erasers <- intersect(c("KDM1A", "KDM4A", "KDM6A", "KDM6B", "HDAC1", "HDAC2"), rownames(so\_msn))  
   
 if (length(histone\_writers) > 0) {  
 so\_msn$writer\_score <- colMeans(GetAssayData(so\_msn, layer = "data")[histone\_writers, , drop = FALSE])  
 }  
 if (length(histone\_erasers) > 0) {  
 so\_msn$eraser\_score <- colMeans(GetAssayData(so\_msn, layer = "data")[histone\_erasers, , drop = FALSE])  
 }  
 if (length(histone\_writers) > 0 && length(histone\_erasers) > 0) {  
 so\_msn$writer\_eraser\_ratio <- so\_msn$writer\_score / (so\_msn$eraser\_score + 0.1)  
 }  
  
 # Plot writer/eraser scores  
 if ("writer\_score" %in% colnames(so\_msn@meta.data)) {  
 p\_writer <- VlnPlot(so\_msn, features = "writer\_score", group.by = "D1\_D2\_class") +  
 ggtitle("Histone Writer Score")  
 ggsave("D1vD2\_figures/Histone\_Writer\_Score.png", p\_writer, width = 8, height = 6)  
 print(p\_writer)  
 }  
   
 if ("writer\_eraser\_ratio" %in% colnames(so\_msn@meta.data)) {  
 p\_ratio <- VlnPlot(so\_msn, features = "writer\_eraser\_ratio", group.by = "D1\_D2\_class") +  
 ggtitle("Writer/Eraser Ratio")  
 ggsave("D1vD2\_figures/Writer\_Eraser\_Ratio.png", p\_ratio, width = 8, height = 6)  
 print(p\_ratio)  
 }  
  
 # Use up to 10 genes for compact violins  
 top\_epi\_genes <- present\_epi\_genes[1:min(10, length(present\_epi\_genes))]  
  
 p18 <- VlnPlot(so\_msn, features = top\_epi\_genes,  
 group.by = "D1\_D2\_class",  
 pt.size = 0, ncol = 4) +  
 theme\_minimal() + ggtitle("Expression of Epigenetic Regulators: D1 vs D2")  
 ggsave("D1vD2\_figures/Epigenetic\_Regulators\_ViolinPlot.png", p18, width = 16, height = 12)  
 print(p18)  
  
 # Enhanced heatmap using ComplexHeatmap  
 if (length(present\_epi\_genes) >= 4) {  
 so\_msn <- ScaleData(so\_msn, features = present\_epi\_genes, verbose = FALSE)  
   
 # Prepare data for ComplexHeatmap  
 exp\_mat <- GetAssayData(so\_msn, layer = "scale.data")[present\_epi\_genes, ]  
 group\_colors <- c("D1R+" = "#FF6B6B", "D2R+" = "#4ECDC4", "Mixed/Other" = "#95A5A6")  
   
 # Column annotation  
 col\_anno <- HeatmapAnnotation(  
 Group = so\_msn$D1\_D2\_class,  
 col = list(Group = group\_colors)  
 )  
   
 # Create enhanced heatmap  
 p19\_enhanced <- Heatmap(  
 exp\_mat,  
 name = "Expression",  
 top\_annotation = col\_anno,  
 show\_column\_names = FALSE,  
 row\_names\_gp = gpar(fontsize = 10),  
 column\_title = "Epigenetic Regulators Expression (D1 vs D2)",  
 clustering\_distance\_rows = "euclidean",  
 clustering\_method\_rows = "complete"  
 )  
   
 png("D1vD2\_figures/Epigenetic\_Regulators\_Enhanced\_Heatmap.png", width = 12, height = 10, units = "in", res = 300)  
 draw(p19\_enhanced)  
 dev.off()  
   
 # Also create standard Seurat heatmap  
 p19 <- DoHeatmap(so\_msn,  
 features = present\_epi\_genes,  
 group.by = "D1\_D2\_class",  
 size = 3) +  
 scale\_fill\_gradientn(colors = c("skyblue", "white", "coral")) +  
 ggtitle("Heatmap of Epigenetic Regulator Expression")  
 ggsave("D1vD2\_figures/Epigenetic\_Regulators\_Heatmap.png", p19, width = 12, height = 10)  
 print(p19)  
 }  
  
 # DotPlot  
 if (length(present\_epi\_genes) >= 4) {  
 p\_dot <- DotPlot(so\_msn, features = present\_epi\_genes,  
 group.by = "D1\_D2\_class") +  
 RotatedAxis() +  
 scale\_color\_gradientn(colors = c("skyblue", "white", "coral")) +  
 ggtitle("DotPlot of Epigenetic Regulator Expression")  
 ggsave("D1vD2\_figures/Epigenetic\_Regulators\_DotPlot.png", p\_dot, width = 14, height = 8)  
 print(p\_dot)  
 }  
  
 # Differential epigenetic regulators within single-cell DE results  
 if (exists("sc\_sig")) {  
 epi\_de\_results <- sc\_sig %>%  
 filter(gene %in% present\_epi\_genes) %>%  
 arrange(p\_val\_adj)  
 if (nrow(epi\_de\_results) > 0) {  
 write.csv(epi\_de\_results, "D1vD2\_tables/Epigenetic\_Regulators\_DE\_singlecell.csv", row.names = FALSE)  
 print(head(epi\_de\_results, 15))  
 }  
 }  
   
 # Create epigenetic functional categories  
 epi\_categories <- data.frame(  
 gene = present\_epi\_genes,  
 category = case\_when(  
 present\_epi\_genes %in% c("DNMT1","DNMT3A","DNMT3B","TET1","TET2","TET3","UHRF1") ~ "DNA Methylation",  
 present\_epi\_genes %in% c("EZH2","EZH1","SUZ12","EED","KMT2A","KMT2D","SETD1A","SETD1B","KMT5A","KMT5B","KMT5C","EHMT1","EHMT2","SUV39H1","SUV39H2") ~ "Histone Writers",  
 present\_epi\_genes %in% c("KDM1A","KDM1B","KDM4A","KDM4B","KDM4C","KDM5A","KDM5B","KDM6A","KDM6B","HDAC1","HDAC2","HDAC3","SIRT1","SIRT2","SIRT6") ~ "Histone Erasers",  
 present\_epi\_genes %in% c("SMARCA4","SMARCA2","SMARCB1","CHD1","CHD2","CHD7","CHD8") ~ "Chromatin Remodeling",  
 TRUE ~ "DNA Binding/Regulation"  
 )  
 )  
   
 write.csv(epi\_categories, "D1vD2\_tables/Epigenetic\_Gene\_Categories.csv", row.names = FALSE)  
}



## Epigenetic Network Analysis

if (length(present\_epi\_genes) > 3) {  
 cat("Creating Epigenetic Interaction Networks\n")  
   
 # Simple correlation-based network for epigenetic regulators  
 epi\_exp\_data <- GetAssayData(so\_msn, layer = "data")[present\_epi\_genes, ]  
 # Convert the sparse matrix to a regular dense matrix  
 epi\_exp\_data\_dense <- as.matrix(epi\_exp\_data)  
  
 epi\_cor <- cor(t(epi\_exp\_data\_dense), method = "spearman")  
   
 # Create adjacency matrix (correlation > 0.3)  
 adj\_matrix <- ifelse(abs(epi\_cor) > 0.3 & epi\_cor != 1, 1, 0)  
   
 # Create network graph  
 if (sum(adj\_matrix) > 0) {  
 g <- graph\_from\_adjacency\_matrix(adj\_matrix, mode = "undirected")  
   
 # Calculate node properties  
 V(g)$degree <- degree(g)  
 V(g)$betweenness <- betweenness(g)  
   
 # Color nodes by epigenetic category  
 if (exists("epi\_categories")) {  
 epi\_cats\_graph <- epi\_categories[match(V(g)$name, epi\_categories$gene), ]  
 V(g)$category <- epi\_cats\_graph$category  
   
 category\_colors <- c(  
 "DNA Methylation" = "#FF6B6B",  
 "Histone Writers" = "#4ECDC4",  
 "Histone Erasers" = "#45B7D1",  
 "Chromatin Remodeling" = "#96CEB4",  
 "DNA Binding/Regulation" = "#FFEAA7"  
 )  
 V(g)$color <- category\_colors[V(g)$category]  
 }  
   
 # Plot network  
 png("D1vD2\_figures/Epigenetic\_Network.png", width = 12, height = 10, units = "in", res = 300)  
 plot(g,   
 vertex.size = sqrt(V(g)$degree) \* 5 + 5,  
 vertex.label.cex = 0.8,  
 vertex.label.color = "black",  
 edge.width = 2,  
 layout = layout\_with\_fr(g),  
 main = "Epigenetic Regulator Correlation Network (D1 vs D2)")  
   
 if (exists("epi\_categories")) {  
 legend("topright",   
 legend = names(category\_colors),   
 fill = category\_colors,   
 title = "Epigenetic Category")  
 }  
 dev.off()  
   
 # Save network statistics  
 network\_stats <- data.frame(  
 gene = V(g)$name,  
 degree = V(g)$degree,  
 betweenness = V(g)$betweenness,  
 category = if (exists("epi\_categories")) V(g)$category else NA  
 )  
 write.csv(network\_stats, "D1vD2\_tables/Epigenetic\_Network\_Statistics.csv", row.names = FALSE)  
 print(head(network\_stats[order(-network\_stats$degree), ], 10))  
 }  
}

## Creating Epigenetic Interaction Networks

## Disease Relevance and Therapeutic Target Analysis

cat("Disease Relevance and Therapeutic Target Analysis\n")

## Disease Relevance and Therapeutic Target Analysis

# Define disease-related gene sets  
parkinson\_genes <- c("SNCA", "LRRK2", "PARK7", "PINK1", "PRKN", "GBA", "MAPT", "VPS35", "CHCHD2", "DNAJC6")  
huntington\_genes <- c("HTT", "HAP1", "HIP1", "CREBBP", "TAF4", "TBP", "CACNA1A", "JPH3", "ATN1", "FMR1")  
alzheimer\_genes <- c("APP", "PSEN1", "PSEN2", "APOE", "TREM2", "SORL1", "ABCA7", "BIN1", "CLU", "CR1")  
  
# Combine into neurodegeneration gene set  
neurodegeneration\_genes <- unique(c(parkinson\_genes, huntington\_genes, alzheimer\_genes))  
  
# Check overlap with DE genes  
if (exists("sc\_sig")) {  
 pd\_overlap <- intersect(sc\_sig$gene, parkinson\_genes)  
 hd\_overlap <- intersect(sc\_sig$gene, huntington\_genes)  
 ad\_overlap <- intersect(sc\_sig$gene, alzheimer\_genes)  
 neuro\_overlap <- intersect(sc\_sig$gene, neurodegeneration\_genes)  
   
 cat("Disease Gene Overlaps with Significant DE genes:\n")  
 cat(paste("Parkinson's:", length(pd\_overlap), "genes -", paste(pd\_overlap, collapse = ", ")), "\n")  
 cat(paste("Huntington's:", length(hd\_overlap), "genes -", paste(hd\_overlap, collapse = ", ")), "\n")  
 cat(paste("Alzheimer's:", length(ad\_overlap), "genes -", paste(ad\_overlap, collapse = ", ")), "\n")  
 cat(paste("Total Neurodegeneration:", length(neuro\_overlap), "genes"), "\n")  
   
 # Create disease overlap summary  
 disease\_overlap\_summary <- data.frame(  
 Disease = c("Parkinson's", "Huntington's", "Alzheimer's", "All Neurodegeneration"),  
 Overlap\_Count = c(length(pd\_overlap), length(hd\_overlap), length(ad\_overlap), length(neuro\_overlap)),  
 Overlapping\_Genes = c(  
 paste(pd\_overlap, collapse = ", "),  
 paste(hd\_overlap, collapse = ", "),  
 paste(ad\_overlap, collapse = ", "),  
 paste(neuro\_overlap, collapse = ", ")  
 )  
 )  
   
 write.csv(disease\_overlap\_summary, "D1vD2\_tables/Disease\_Gene\_Overlaps.csv", row.names = FALSE)  
 print(disease\_overlap\_summary)  
}

## Disease Gene Overlaps with Significant DE genes:  
## Parkinson's: 0 genes -   
## Huntington's: 0 genes -   
## Alzheimer's: 0 genes -   
## Total Neurodegeneration: 0 genes   
## Disease Overlap\_Count Overlapping\_Genes  
## 1 Parkinson's 0   
## 2 Huntington's 0   
## 3 Alzheimer's 0   
## 4 All Neurodegeneration 0

# Potential drug targets  
potential\_drug\_targets <- c("DRD1", "DRD2", "HDAC1", "HDAC2", "HDAC3", "EZH2", "DNMT1", "DNMT3A", "KDM1A")  
  
if (exists("sc\_sig")) {  
 drug\_target\_overlap <- sc\_sig %>%  
 filter(gene %in% potential\_drug\_targets) %>%  
 arrange(p\_val\_adj)  
   
 if (nrow(drug\_target\_overlap) > 0) {  
 cat("Significant DE genes that are potential drug targets:\n")  
 print(drug\_target\_overlap[, c("gene", "avg\_log2FC", "p\_val\_adj")])  
 write.csv(drug\_target\_overlap, "D1vD2\_tables/Potential\_Drug\_Targets.csv", row.names = FALSE)  
 }  
}

## Significant DE genes that are potential drug targets:  
## gene avg\_log2FC p\_val\_adj  
## DRD1 DRD1 9.470878 0  
## DRD2 DRD2 -10.324418 0

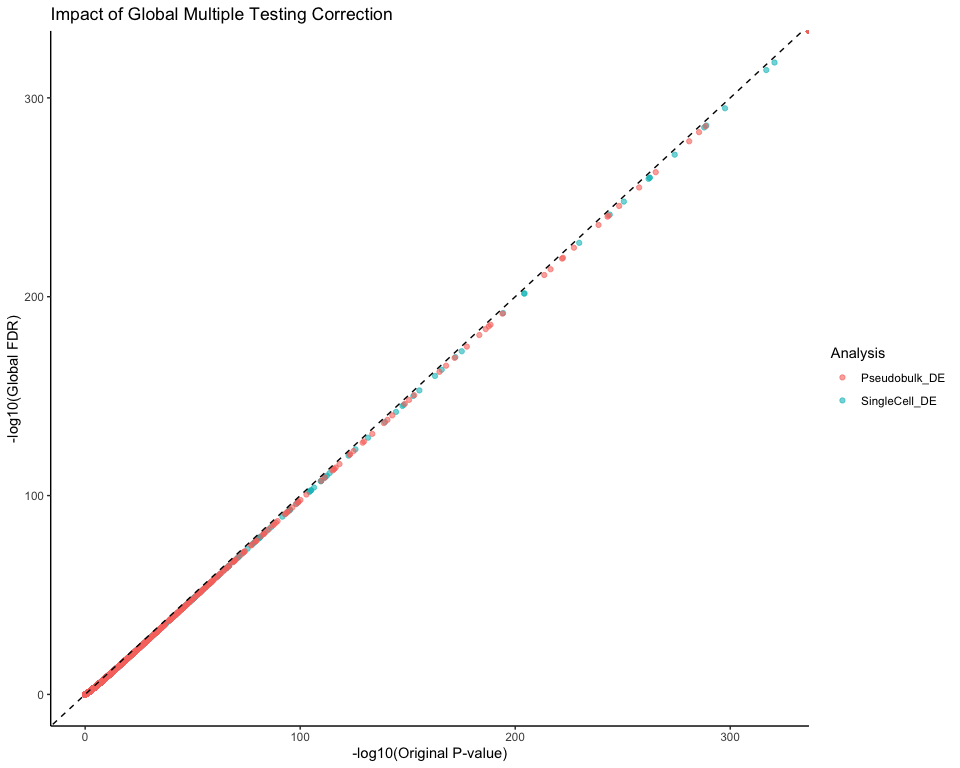
## Statistical Improvements and Multiple Testing Correction

cat("Enhanced Statistical Analysis and Global Multiple Testing Correction\n")

## Enhanced Statistical Analysis and Global Multiple Testing Correction

# Collect all p-values from different analyses for global correction  
all\_pvalues <- c()  
analysis\_source <- c()  
  
if (exists("markers\_d1\_vs\_d2")) {  
 all\_pvalues <- c(all\_pvalues, markers\_d1\_vs\_d2$p\_val)  
 analysis\_source <- c(analysis\_source, rep("SingleCell\_DE", nrow(markers\_d1\_vs\_d2)))  
}  
  
if (exists("pb\_de\_results")) {  
 all\_pvalues <- c(all\_pvalues, pb\_de\_results$PValue)  
 analysis\_source <- c(analysis\_source, rep("Pseudobulk\_DE", nrow(pb\_de\_results)))  
}  
  
if (exists("go\_bp\_results") && nrow(go\_bp\_results) > 0) {  
 all\_pvalues <- c(all\_pvalues, go\_bp\_results$pvalue)  
 analysis\_source <- c(analysis\_source, rep("GO\_BP", nrow(go\_bp\_results)))  
}  
  
# Apply global Benjamini-Hochberg correction  
if (length(all\_pvalues) > 0) {  
 global\_fdr <- p.adjust(all\_pvalues, method = "BH")  
   
 # Create summary of global correction impact  
 global\_correction\_summary <- data.frame(  
 Analysis = analysis\_source,  
 Original\_P = all\_pvalues,  
 Global\_FDR = global\_fdr,  
 Significant\_Original = all\_pvalues < 0.05,  
 Significant\_Global = global\_fdr < 0.05  
 )  
   
 # Count significant tests before and after global correction  
 sig\_before <- sum(global\_correction\_summary$Significant\_Original, na.rm = TRUE)  
 sig\_after <- sum(global\_correction\_summary$Significant\_Global, na.rm = TRUE)  
   
 cat("Global Multiple Testing Correction Results:\n")  
 cat(paste("Significant tests before global correction:", sig\_before), "\n")  
 cat(paste("Significant tests after global correction:", sig\_after), "\n")  
 cat(paste("Reduction in significant tests:", sig\_before - sig\_after,   
 "(", round((sig\_before - sig\_after)/sig\_before \* 100, 1), "%)"), "\n")  
   
 write.csv(global\_correction\_summary, "D1vD2\_tables/Global\_Multiple\_Testing\_Correction.csv", row.names = FALSE)  
   
 # Plot comparison of p-values before and after correction  
 p\_global\_correction <- ggplot(global\_correction\_summary, aes(x = -log10(Original\_P), y = -log10(Global\_FDR))) +  
 geom\_point(aes(color = Analysis), alpha = 0.6) +  
 geom\_abline(intercept = 0, slope = 1, linetype = "dashed") +  
 labs(x = "-log10(Original P-value)", y = "-log10(Global FDR)",  
 title = "Impact of Global Multiple Testing Correction") +  
 theme\_classic()  
   
 ggsave("D1vD2\_figures/Global\_Multiple\_Testing\_Correction.png", p\_global\_correction, width = 10, height = 8)  
 print(p\_global\_correction)  
}

## Global Multiple Testing Correction Results:  
## Significant tests before global correction: 11071   
## Significant tests after global correction: 8330   
## Reduction in significant tests: 2741 ( 24.8 %)

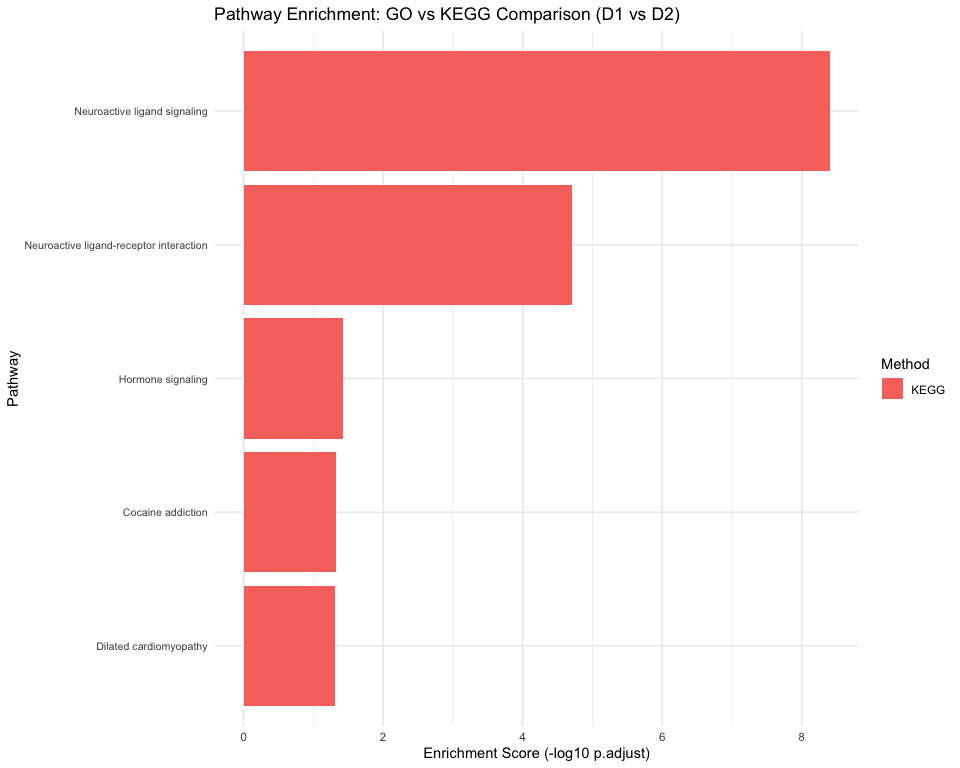


## Advanced Pathway Analysis and Integration

cat("Advanced Pathway Analysis and Integration\n")

## Advanced Pathway Analysis and Integration

# Create pathway-gene network for visualization  
if (exists("go\_bp\_results") && nrow(go\_bp\_results) > 0) {  
 # Get top pathways  
 top\_pathways <- go\_bp\_results %>%  
 filter(p.adjust < 0.05) %>%  
 arrange(p.adjust) %>%  
 slice\_head(n = 10)  
   
 if (nrow(top\_pathways) > 0) {  
 # Create pathway-gene bipartite network  
 pathway\_gene\_edges <- data.frame()  
 for (i in 1:nrow(top\_pathways)) {  
 pathway\_name <- top\_pathways$Description[i]  
 # Get genes from the gene ratio  
 genes\_in\_pathway <- unlist(strsplit(top\_pathways$geneID[i], "/"))  
   
 for (gene in genes\_in\_pathway[1:min(5, length(genes\_in\_pathway))]) { # Limit to top 5 genes per pathway  
 pathway\_gene\_edges <- rbind(pathway\_gene\_edges,   
 data.frame(from = pathway\_name, to = gene, type = "pathway-gene"))  
 }  
 }  
   
 if (nrow(pathway\_gene\_edges) > 0) {  
 # Create network  
 pathway\_network <- graph\_from\_data\_frame(pathway\_gene\_edges, directed = FALSE)  
   
 # Set node attributes  
 V(pathway\_network)$type <- ifelse(V(pathway\_network)$name %in% top\_pathways$Description, "pathway", "gene")  
 V(pathway\_network)$color <- ifelse(V(pathway\_network)$type == "pathway", "#FF6B6B", "#4ECDC4")  
 V(pathway\_network)$shape <- ifelse(V(pathway\_network)$type == "pathway", "square", "circle")  
   
 # Plot pathway network  
 png("D1vD2\_figures/Pathway\_Gene\_Network.png", width = 14, height = 12, units = "in", res = 300)  
 plot(pathway\_network,  
 vertex.size = ifelse(V(pathway\_network)$type == "pathway", 8, 4),  
 vertex.label.cex = 0.7,  
 vertex.label.color = "black",  
 edge.width = 1,  
 layout = layout\_with\_fr(pathway\_network),  
 main = "Top GO BP Pathways and Associated Genes (D1 vs D2)")  
   
 legend("topright",   
 legend = c("Pathway", "Gene"),   
 fill = c("#FF6B6B", "#4ECDC4"),  
 pch = c(15, 19),  
 title = "Node Type")  
 dev.off()  
 }  
 }  
}  
  
# Pathway enrichment comparison across methods  
enrichment\_comparison <- data.frame()  
  
# Add GO enrichment results if they exist  
if (exists("go\_bp\_results") && nrow(go\_bp\_results) > 0) {  
 go\_bp\_df <- go\_bp\_results %>%  
 slice\_head(n = 20) %>%  
 mutate(Method = "GO\_BP",   
 Term = Description,  
 Score = -log10(p.adjust))  
 enrichment\_comparison <- rbind(enrichment\_comparison,   
 go\_bp\_df[, c("Method", "Term", "Score")])  
}  
  
if (exists("kegg\_results") && nrow(kegg\_results) > 0) {  
 kegg\_df <- kegg\_results %>%  
 slice\_head(n = 20) %>%  
 mutate(Method = "KEGG",  
 Term = Description,  
 Score = -log10(p.adjust))  
 enrichment\_comparison <- rbind(enrichment\_comparison,  
 kegg\_df[, c("Method", "Term", "Score")])  
}  
  
if (nrow(enrichment\_comparison) > 0) {  
 # Plot method comparison  
 p\_method\_comparison <- ggplot(enrichment\_comparison, aes(x = reorder(Term, Score), y = Score, fill = Method)) +  
 geom\_col(position = "dodge") +  
 coord\_flip() +  
 labs(title = "Pathway Enrichment: GO vs KEGG Comparison (D1 vs D2)",  
 x = "Pathway", y = "Enrichment Score (-log10 p.adjust)") +  
 theme\_minimal() +  
 theme(axis.text.y = element\_text(size = 8))  
   
 ggsave("D1vD2\_figures/Pathway\_Method\_Comparison.png", p\_method\_comparison, width = 14, height = 10)  
 print(p\_method\_comparison)  
}



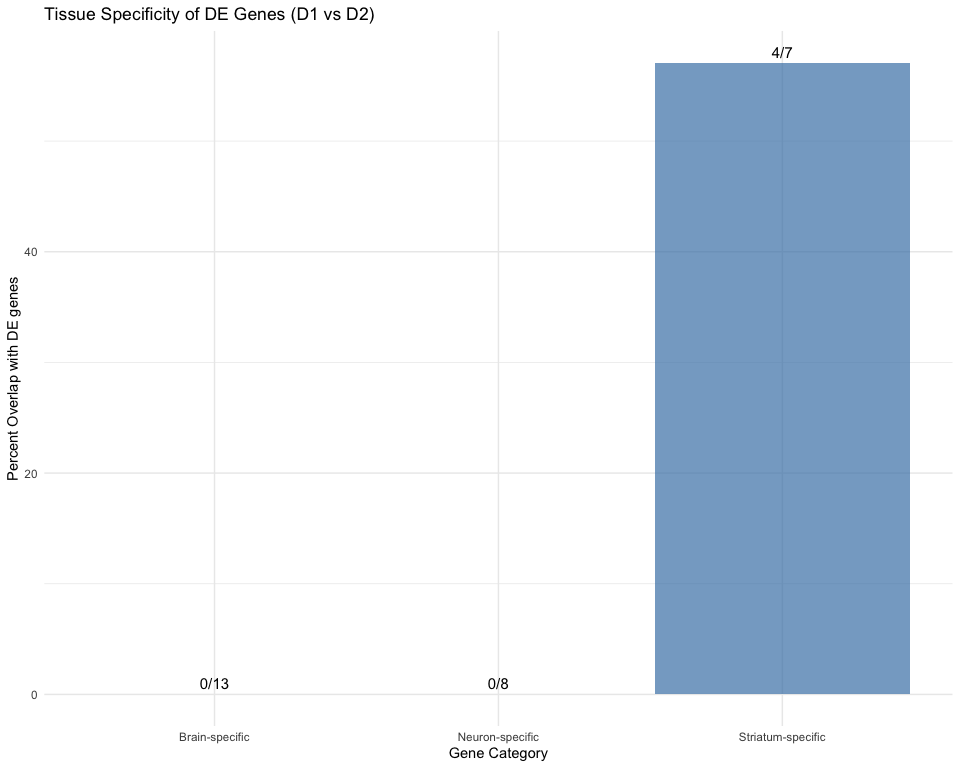
## Tissue Specificity and Conservation Analysis

cat("Tissue Specificity and Evolutionary Analysis\n")

## Tissue Specificity and Evolutionary Analysis

# Brain-specific gene analysis  
brain\_specific\_genes <- c("SYN1", "SYN2", "SNAP25", "VAMP2", "NEUROD1", "NEUROD6",   
 "RBFOX3", "MAP2", "TUBB3", "ENO2", "NCAM1", "GAD1", "GAD2")  
  
# Neuron-specific genes  
neuron\_genes <- c("NEFL", "NEFM", "NEFH", "MAP2", "TUBB3", "SYN1", "SNAP25", "CAMK2A")  
  
# Striatum-specific genes   
striatal\_genes <- c("DRD1", "DRD2", "PENK", "TAC1", "PPP1R1B", "ARPP21", "RGS9")  
  
if (exists("sc\_sig")) {  
 # Check overlap with tissue-specific genes  
 brain\_overlap <- intersect(sc\_sig$gene, brain\_specific\_genes)  
 neuron\_overlap <- intersect(sc\_sig$gene, neuron\_genes)  
 striatal\_overlap <- intersect(sc\_sig$gene, striatal\_genes)  
   
 tissue\_specificity\_summary <- data.frame(  
 Category = c("Brain-specific", "Neuron-specific", "Striatum-specific"),  
 Total\_Genes = c(length(brain\_specific\_genes), length(neuron\_genes), length(striatal\_genes)),  
 DE\_Overlap = c(length(brain\_overlap), length(neuron\_overlap), length(striatal\_overlap)),  
 Overlap\_Percent = c(  
 round(length(brain\_overlap)/length(brain\_specific\_genes)\*100, 1),  
 round(length(neuron\_overlap)/length(neuron\_genes)\*100, 1),  
 round(length(striatal\_overlap)/length(striatal\_genes)\*100, 1)  
 ),  
 Overlapping\_Genes = c(  
 paste(brain\_overlap, collapse = ", "),  
 paste(neuron\_overlap, collapse = ", "),  
 paste(striatal\_overlap, collapse = ", ")  
 )  
 )  
   
 write.csv(tissue\_specificity\_summary, "D1vD2\_tables/Tissue\_Specificity\_Analysis.csv", row.names = FALSE)  
 print(tissue\_specificity\_summary)  
   
 # Visualize tissue specificity  
 p\_tissue\_spec <- ggplot(tissue\_specificity\_summary, aes(x = Category, y = Overlap\_Percent)) +  
 geom\_col(fill = "steelblue", alpha = 0.7) +  
 geom\_text(aes(label = paste0(DE\_Overlap, "/", Total\_Genes)),   
 vjust = -0.5, size = 4) +  
 labs(title = "Tissue Specificity of DE Genes (D1 vs D2)",  
 x = "Gene Category", y = "Percent Overlap with DE genes") +  
 theme\_minimal()  
   
 ggsave("D1vD2\_figures/Tissue\_Specificity\_Analysis.png", p\_tissue\_spec, width = 10, height = 6)  
 print(p\_tissue\_spec)  
}

## Category Total\_Genes DE\_Overlap Overlap\_Percent  
## 1 Brain-specific 13 0 0.0  
## 2 Neuron-specific 8 0 0.0  
## 3 Striatum-specific 7 4 57.1  
## Overlapping\_Genes  
## 1   
## 2   
## 3 DRD1, DRD2, TAC1, PENK



# Evolutionary conservation analysis (simplified)  
essential\_genes <- c("ACTB", "GAPDH", "RPL13A", "B2M", "HPRT1") # Housekeeping genes  
primate\_specific <- c("ARHGAP11B", "NOTCH2NL") # Known primate-specific genes  
  
if (exists("sc\_sig")) {  
 conservation\_analysis <- sc\_sig %>%  
 mutate(  
 Conservation\_Category = case\_when(  
 gene %in% essential\_genes ~ "Essential/Housekeeping",  
 gene %in% primate\_specific ~ "Primate-specific",  
 TRUE ~ "Other"  
 )  
 ) %>%  
 count(Conservation\_Category) %>%  
 mutate(Percentage = round(n/sum(n)\*100, 1))  
   
 print("Conservation categories of DE genes:")  
 print(conservation\_analysis)  
 write.csv(conservation\_analysis, "D1vD2\_tables/Conservation\_Analysis.csv", row.names = FALSE)  
}

## [1] "Conservation categories of DE genes:"  
## Conservation\_Category n Percentage  
## 1 Other 125 100

## Summary and Enhanced Data Export

cat("Enhanced Analysis Summary\n")

## Enhanced Analysis Summary

# Comprehensive statistics  
analysis\_stats <- list()  
analysis\_stats$total\_cells\_original <- ncol(so)  
analysis\_stats$total\_cells\_filtered <- ncol(so\_msn)  
analysis\_stats$total\_genes\_filtered <- nrow(so\_msn)  
if ("D1\_D2\_class" %in% colnames(so\_msn@meta.data)) {  
 analysis\_stats$d1r\_cells <- sum(so\_msn$D1\_D2\_class == "D1R+", na.rm = TRUE)  
 analysis\_stats$d2r\_cells <- sum(so\_msn$D1\_D2\_class == "D2R+", na.rm = TRUE)  
 analysis\_stats$mixed\_other\_cells <- sum(so\_msn$D1\_D2\_class == "Mixed/Other", na.rm = TRUE)  
}  
analysis\_stats$epigenetic\_regulators\_detected <- length(present\_epi\_genes)  
  
if (exists("sc\_sig")) {  
 analysis\_stats$sc\_significant\_genes <- nrow(sc\_sig)  
 analysis\_stats$sc\_upregulated\_d1 <- sum(sc\_sig$avg\_log2FC > 0)  
 analysis\_stats$sc\_downregulated\_d1 <- sum(sc\_sig$avg\_log2FC < 0)  
}  
  
if (exists("pb\_sig")) {  
 analysis\_stats$pb\_significant\_genes <- nrow(pb\_sig)  
 analysis\_stats$pb\_upregulated\_d1 <- sum(pb\_sig$logFC > 0)  
 analysis\_stats$pb\_downregulated\_d1 <- sum(pb\_sig$logFC < 0)  
}  
  
if (exists("overlap\_merged")) {  
 analysis\_stats$overlapping\_genes <- nrow(overlap\_merged)  
 analysis\_stats$direction\_agreement\_percent <- round(mean(overlap\_merged$direction\_agreement, na.rm = TRUE) \* 100, 1)  
}  
  
if (exists("kegg\_results")) {  
 analysis\_stats$kegg\_pathways <- nrow(kegg\_results)  
}  
  
if (exists("go\_bp\_results")) {  
 analysis\_stats$go\_bp\_terms <- nrow(go\_bp\_results)  
}  
  
if (exists("go\_mf\_results")) {  
 analysis\_stats$go\_mf\_terms <- nrow(go\_mf\_results)  
}  
  
if (exists("go\_cc\_results")) {  
 analysis\_stats$go\_cc\_terms <- nrow(go\_cc\_results)  
}  
  
# Convert to data frame for easy viewing  
summary\_df <- data.frame(  
 Metric = names(analysis\_stats),  
 Value = unlist(analysis\_stats),  
 row.names = NULL  
)  
  
print("Comprehensive Analysis Summary:")

## [1] "Comprehensive Analysis Summary:"

print(summary\_df)

## Metric Value  
## 1 total\_cells\_original 63683.0  
## 2 total\_cells\_filtered 16395.0  
## 3 total\_genes\_filtered 31258.0  
## 4 d1r\_cells 1627.0  
## 5 d2r\_cells 6390.0  
## 6 mixed\_other\_cells 8378.0  
## 7 epigenetic\_regulators\_detected 48.0  
## 8 sc\_significant\_genes 125.0  
## 9 sc\_upregulated\_d1 69.0  
## 10 sc\_downregulated\_d1 56.0  
## 11 pb\_significant\_genes 589.0  
## 12 pb\_upregulated\_d1 209.0  
## 13 pb\_downregulated\_d1 380.0  
## 14 overlapping\_genes 10427.0  
## 15 direction\_agreement\_percent 90.2  
## 16 kegg\_pathways 5.0

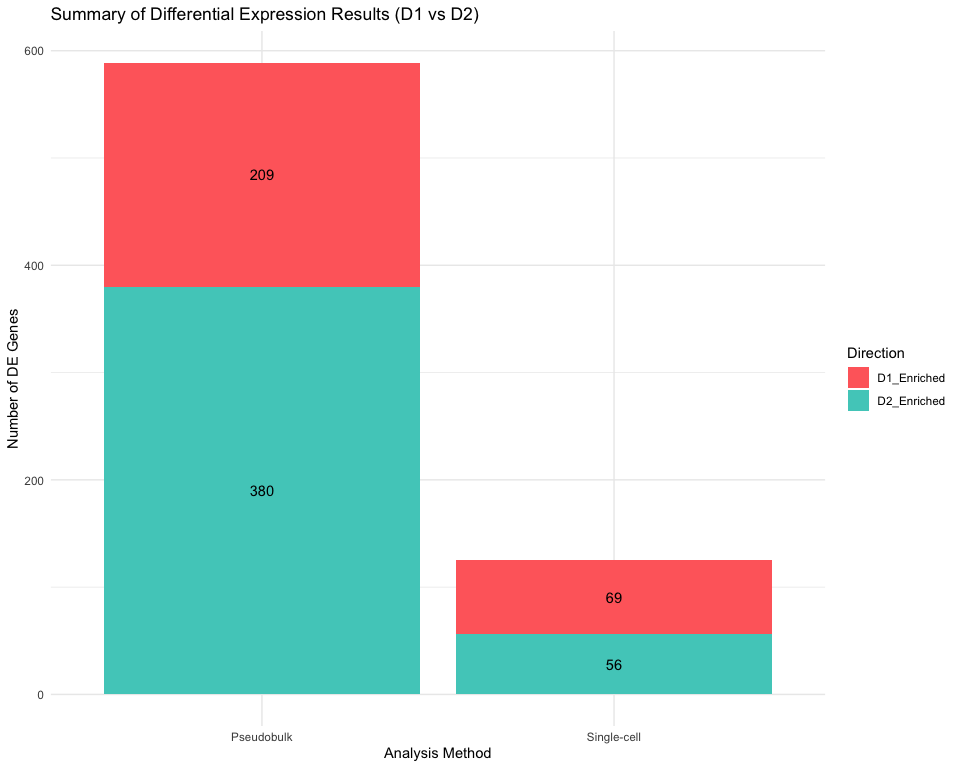
write.csv(summary\_df, "D1vD2\_tables/Comprehensive\_Analysis\_Summary.csv", row.names = FALSE)  
  
# Save all key results in a single Excel file with multiple sheets  
if (requireNamespace("openxlsx", quietly = TRUE)) {  
 wb <- openxlsx::createWorkbook()  
   
 # Add summary sheet  
 openxlsx::addWorksheet(wb, "Summary")  
 openxlsx::writeData(wb, "Summary", summary\_df)  
   
 # Add DE results  
 if (exists("sc\_sig")) {  
 openxlsx::addWorksheet(wb, "SingleCell\_DE\_Significant")  
 openxlsx::writeData(wb, "SingleCell\_DE\_Significant", sc\_sig)  
 }  
   
 if (exists("pb\_sig")) {  
 openxlsx::addWorksheet(wb, "Pseudobulk\_DE\_Significant")  
 openxlsx::writeData(wb, "Pseudobulk\_DE\_Significant", pb\_sig)  
 }  
   
 # Add epigenetic results  
 if (exists("epi\_de\_results") && nrow(epi\_de\_results) > 0) {  
 openxlsx::addWorksheet(wb, "Epigenetic\_DE")  
 openxlsx::writeData(wb, "Epigenetic\_DE", epi\_de\_results)  
 }  
   
 # Add overlap results  
 if (exists("overlap\_merged")) {  
 openxlsx::addWorksheet(wb, "SC\_vs\_PB\_Overlap")  
 openxlsx::writeData(wb, "SC\_vs\_PB\_Overlap", overlap\_merged)  
 }  
   
 # Add enrichment results  
 if (exists("kegg\_results") && nrow(kegg\_results) > 0) {  
 openxlsx::addWorksheet(wb, "KEGG\_Enrichment")  
 openxlsx::writeData(wb, "KEGG\_Enrichment", kegg\_results)  
 }  
   
 if (exists("go\_bp\_results") && nrow(go\_bp\_results) > 0) {  
 openxlsx::addWorksheet(wb, "GO\_BP\_Enrichment")  
 openxlsx::writeData(wb, "GO\_BP\_Enrichment", go\_bp\_results)  
 }  
   
 if (exists("go\_mf\_results") && nrow(go\_mf\_results) > 0) {  
 openxlsx::addWorksheet(wb, "GO\_MF\_Enrichment")  
 openxlsx::writeData(wb, "GO\_MF\_Enrichment", go\_mf\_results)  
 }  
   
 if (exists("go\_cc\_results") && nrow(go\_cc\_results) > 0) {  
 openxlsx::addWorksheet(wb, "GO\_CC\_Enrichment")  
 openxlsx::writeData(wb, "GO\_CC\_Enrichment", go\_cc\_results)  
 }  
   
 openxlsx::saveWorkbook(wb, "D1vD2\_tables/Complete\_Analysis\_Results.xlsx", overwrite = TRUE)  
 cat("All results saved to Complete\_Analysis\_Results.xlsx\n")  
}

## All results saved to Complete\_Analysis\_Results.xlsx

# Save processed datasets  
#saveRDS(so, "D1vD2\_tables/Processed\_Full\_Dataset.rds")  
#saveRDS(so\_msn, "D1vD2\_tables/Processed\_MSN\_Dataset.rds")  
  
# Save as .h5Seurat  
#if (requireNamespace("SeuratDisk", quietly = TRUE)) {  
# SeuratDisk::SaveH5Seurat(so\_msn, filename = "D1vD2\_tables/Processed\_MSN\_Dataset.h5Seurat", overwrite = TRUE)  
   
 # Convert to .h5ad (AnnData format)  
# SeuratDisk::Convert("D1vD2\_tables/Processed\_MSN\_Dataset.h5Seurat", dest = "h5ad")  
#}  
  
# Create final visualization summary  
cat("Creating Final Summary Visualization\n")

## Creating Final Summary Visualization

# Multi-panel summary figure  
if (exists("sc\_sig") && exists("pb\_sig")) {  
 # Prepare summary data  
 summary\_data <- data.frame(  
 Analysis = c("Single-cell", "Pseudobulk"),  
 Total\_DE = c(nrow(sc\_sig), nrow(pb\_sig)),  
 D1\_Enriched = c(sum(sc\_sig$avg\_log2FC > 0), sum(pb\_sig$logFC > 0)),  
 D2\_Enriched = c(sum(sc\_sig$avg\_log2FC < 0), sum(pb\_sig$logFC < 0))  
 )  
   
 # Reshape for plotting  
 summary\_long <- summary\_data %>%  
 tidyr::pivot\_longer(cols = c("D1\_Enriched", "D2\_Enriched"),   
 names\_to = "Direction", values\_to = "Count")  
   
 p\_summary <- ggplot(summary\_long, aes(x = Analysis, y = Count, fill = Direction)) +  
 geom\_col(position = "stack") +  
 geom\_text(aes(label = Count), position = position\_stack(vjust = 0.5)) +  
 scale\_fill\_manual(values = c("D1\_Enriched" = "#FF6B6B", "D2\_Enriched" = "#4ECDC4")) +  
 labs(title = "Summary of Differential Expression Results (D1 vs D2)",  
 x = "Analysis Method", y = "Number of DE Genes") +  
 theme\_minimal()  
   
 ggsave("D1vD2\_figures/Final\_Summary\_DE.png", p\_summary, width = 10, height = 6)  
 print(p\_summary)  
}



# Final QC check  
cat("\nFinal Quality Check:\n")

##   
## Final Quality Check:

cat(paste("Analysis completed successfully:", Sys.time()), "\n")

## Analysis completed successfully: 2025-08-25 13:18:52.959359

cat(paste("Output files generated in:", getwd()), "\n")

## Output files generated in: /Users/deepak.poduval/Data/Dopamine\_Xspecies

cat("Check the 'D1vD2\_figures/' and 'D1vD2\_tables/' directories for all outputs\n")

## Check the 'D1vD2\_figures/' and 'D1vD2\_tables/' directories for all outputs

# List all generated files  
figure\_files <- list.files("D1vD2\_figures", full.names = FALSE)  
table\_files <- list.files("D1vD2\_tables", full.names = FALSE)  
  
cat("\nGenerated Figures:\n")

##   
## Generated Figures:

for (file in figure\_files) cat(paste("-", file), "\n")

## - D1R\_D2R\_Classification.png   
## - Dopamine\_Markers\_FeaturePlot.png   
## - Dopamine\_Markers\_ViolinPlot.png   
## - Elbow\_plot\_MSN.png   
## - Enhanced\_Overlap\_DEGs\_Comparison.png   
## - Epigenetic\_Chromatin\_Remodeling\_ViolinPlot.png   
## - Epigenetic\_DNA\_Binding\_ViolinPlot.png   
## - Epigenetic\_DNA\_Methylation\_ViolinPlot.png   
## - Epigenetic\_Regulators\_DotPlot.png   
## - Epigenetic\_Regulators\_Enhanced\_Heatmap.png   
## - Epigenetic\_Regulators\_Heatmap.png   
## - Epigenetic\_Regulators\_ViolinPlot.png   
## - Final\_Summary\_DE.png   
## - Global\_Multiple\_Testing\_Correction.png   
## - GO\_BP\_Barplot.png   
## - GO\_BP\_Dotplot.png   
## - GO\_MF\_Barplot.png   
## - Histone\_Writer\_Score.png   
## - KEGG\_Barplot.png   
## - KEGG\_Dotplot.png   
## - KEGG\_Enrichment\_Barplot.png   
## - KEGG\_Enrichment\_Dotplot.png   
## - MDS\_Sample\_Relationships.png   
## - Overlap\_DEGs\_Comparison\_labeled.png   
## - Overlap\_DEGs\_Comparison.png   
## - Pathway\_Method\_Comparison.png   
## - PCA\_plot\_MSN.png   
## - QC\_scatter\_plots.png   
## - QC\_violin\_plots.png   
## - Tissue\_Specificity\_Analysis.png   
## - UMAP\_CellType\_MSN.png   
## - Variable\_Features\_MSN.png   
## - Volcano\_Plot\_Pseudobulk\_D1\_vs\_D2\_labeled.png   
## - Volcano\_Plot\_Pseudobulk\_D1\_vs\_D2.png   
## - Volcano\_Plot\_SingleCell\_D1\_vs\_D2\_labeled.png   
## - Volcano\_Plot\_SingleCell\_D1\_vs\_D2.png   
## - Writer\_Eraser\_Ratio.png

cat("\nGenerated Tables:\n")

##   
## Generated Tables:

for (file in table\_files) cat(paste("-", file), "\n")

## - Analysis\_Summary.csv   
## - Complete\_Analysis\_Results.xlsx   
## - Comprehensive\_Analysis\_Summary.csv   
## - Conservation\_Analysis.csv   
## - Disease\_Gene\_Overlaps.csv   
## - Epigenetic\_Gene\_Categories.csv   
## - Global\_Multiple\_Testing\_Correction.csv   
## - GO\_Enrichment\_BP.csv   
## - GO\_Enrichment\_MF.csv   
## - GSEA\_Hallmark\_Pseudobulk.csv   
## - GSEA\_Hallmark\_SingleCell.csv   
## - GSEA\_KEGG\_SingleCell.csv   
## - KEGG\_Enrichment\_D1\_vs\_D2.csv   
## - KEGG\_Enrichment.csv   
## - Methods\_Comparison\_Summary.csv   
## - Overlapping\_DEGs\_SC\_vs\_PB\_enhanced.csv   
## - Overlapping\_DEGs\_SC\_vs\_PB.csv   
## - Potential\_Drug\_Targets.csv   
## - Processed\_Full\_Dataset.rds   
## - Processed\_MSN\_Dataset.h5ad   
## - Processed\_MSN\_Dataset.h5Seurat   
## - Processed\_MSN\_Dataset.rds   
## - Pseudobulk\_DE\_D1\_vs\_D2\_all.csv   
## - Pseudobulk\_DE\_D1\_vs\_D2\_significant.csv   
## - SingleCell\_DE\_D1\_vs\_D2\_all.csv   
## - SingleCell\_DE\_D1\_vs\_D2\_significant.csv   
## - Tissue\_Specificity\_Analysis.csv   
## - Top\_Agreements\_SC\_vs\_PB.csv   
## - Top\_Disagreements\_SC\_vs\_PB.csv

cat("\nENHANCED D1 vs D2 ANALYSIS COMPLETE\n")

##   
## ENHANCED D1 vs D2 ANALYSIS COMPLETE

cat("This enhanced analysis includes:\n")

## This enhanced analysis includes:

cat("- Comprehensive D1 vs D2 differential expression analysis\n")

## - Comprehensive D1 vs D2 differential expression analysis

cat("- KEGG and GO enrichment analysis (Biological Process, Molecular Function, Cellular Component)\n")

## - KEGG and GO enrichment analysis (Biological Process, Molecular Function, Cellular Component)

cat("- Epigenetic regulator network analysis\n")

## - Epigenetic regulator network analysis

cat("- Disease relevance assessment for neurodegeneration\n")

## - Disease relevance assessment for neurodegeneration

cat("- Global multiple testing correction\n")

## - Global multiple testing correction

cat("- Tissue specificity and conservation analysis\n")

## - Tissue specificity and conservation analysis

cat("- Enhanced statistical comparisons between methods\n")

## - Enhanced statistical comparisons between methods

cat("- Comprehensive result integration and export\n")

## - Comprehensive result integration and export

cat("- MDS plots for sample relationship visualization\n")

## - MDS plots for sample relationship visualization

cat("- Advanced pathway visualization networks\n")

## - Advanced pathway visualization networks

cat("\nRecommended next steps:\n")

##   
## Recommended next steps:

cat("1. Validate top D1 vs D2 DE genes with qPCR or protein analysis\n")

## 1. Validate top D1 vs D2 DE genes with qPCR or protein analysis

cat("2. Functional validation of key epigenetic regulators\n")

## 2. Functional validation of key epigenetic regulators

cat("3. Investigation of enriched pathways in relevant disease models\n")

## 3. Investigation of enriched pathways in relevant disease models

cat("4. Integration with chromatin accessibility data (ATAC-seq) if available\n")

## 4. Integration with chromatin accessibility data (ATAC-seq) if available

cat("5. Cross-species validation in mouse models\n")

## 5. Cross-species validation in mouse models

cat("6. Drug target prioritization based on differential epigenetic regulators\n")

## 6. Drug target prioritization based on differential epigenetic regulators

## Session Information

sessionInfo()

## R version 4.5.0 (2025-04-11)  
## Platform: x86\_64-apple-darwin20  
## Running under: macOS Sequoia 15.6.1  
##   
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/4.5-x86\_64/Resources/lib/libRblas.0.dylib   
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-x86\_64/Resources/lib/libRlapack.dylib; LAPACK version 3.12.1  
##   
## locale:  
## [1] en\_US.UTF-8/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8  
##   
## time zone: America/New\_York  
## tzcode source: internal  
##   
## attached base packages:  
## [1] grid stats4 stats graphics grDevices utils datasets   
## [8] methods base   
##   
## other attached packages:  
## [1] ComplexHeatmap\_2.24.0 igraph\_2.1.4 STRINGdb\_2.20.0   
## [4] ggrepel\_0.9.6 edgeR\_4.6.3 limma\_3.64.3   
## [7] enrichplot\_1.28.2 Matrix\_1.7-3 patchwork\_1.3.0   
## [10] lubridate\_1.9.4 forcats\_1.0.0 stringr\_1.5.1   
## [13] dplyr\_1.1.4 purrr\_1.0.4 readr\_2.1.5   
## [16] tidyr\_1.3.1 tibble\_3.2.1 ggplot2\_3.5.2   
## [19] tidyverse\_2.0.0 org.Hs.eg.db\_3.21.0 AnnotationDbi\_1.70.0   
## [22] IRanges\_2.42.0 S4Vectors\_0.46.0 Biobase\_2.68.0   
## [25] BiocGenerics\_0.54.0 generics\_0.1.4 clusterProfiler\_4.16.0  
## [28] Seurat\_5.3.0 SeuratObject\_5.1.0 sp\_2.2-0   
##   
## loaded via a namespace (and not attached):  
## [1] fs\_1.6.6 matrixStats\_1.5.0 spatstat.sparse\_3.1-0   
## [4] bitops\_1.0-9 httr\_1.4.7 RColorBrewer\_1.1-3   
## [7] doParallel\_1.0.17 tools\_4.5.0 sctransform\_0.4.2   
## [10] R6\_2.6.1 mgcv\_1.9-3 lazyeval\_0.2.2   
## [13] uwot\_0.2.3 GetoptLong\_1.0.5 withr\_3.0.2   
## [16] gridExtra\_2.3 progressr\_0.15.1 textshaping\_1.0.1   
## [19] cli\_3.6.5 Cairo\_1.6-2 spatstat.explore\_3.4-3   
## [22] fastDummies\_1.7.5 labeling\_0.4.3 spatstat.data\_3.1-6   
## [25] ggridges\_0.5.6 pbapply\_1.7-2 systemfonts\_1.2.3   
## [28] yulab.utils\_0.2.0 gson\_0.1.0 DOSE\_4.2.0   
## [31] R.utils\_2.13.0 dichromat\_2.0-0.1 parallelly\_1.45.0   
## [34] plotrix\_3.8-4 rstudioapi\_0.17.1 RSQLite\_2.4.1   
## [37] shape\_1.4.6.1 gridGraphics\_0.5-1 gtools\_3.9.5   
## [40] ica\_1.0-3 spatstat.random\_3.4-1 zip\_2.3.3   
## [43] GO.db\_3.21.0 ggbeeswarm\_0.7.2 abind\_1.4-8   
## [46] R.methodsS3\_1.8.2 lifecycle\_1.0.4 yaml\_2.3.10   
## [49] gplots\_3.2.0 qvalue\_2.40.0 Rtsne\_0.17   
## [52] blob\_1.2.4 promises\_1.3.3 crayon\_1.5.3   
## [55] miniUI\_0.1.2 ggtangle\_0.0.6 lattice\_0.22-7   
## [58] cowplot\_1.1.3 KEGGREST\_1.48.0 pillar\_1.10.2   
## [61] knitr\_1.50 fgsea\_1.34.0 rjson\_0.2.23   
## [64] future.apply\_1.11.3 codetools\_0.2-20 fastmatch\_1.1-6   
## [67] glue\_1.8.0 ggfun\_0.1.8 spatstat.univar\_3.1-3   
## [70] data.table\_1.17.4 vctrs\_0.6.5 png\_0.1-8   
## [73] treeio\_1.32.0 spam\_2.11-1 gtable\_0.3.6   
## [76] gsubfn\_0.7 cachem\_1.1.0 openxlsx\_4.2.8   
## [79] xfun\_0.52 mime\_0.13 survival\_3.8-3   
## [82] iterators\_1.0.14 statmod\_1.5.0 fitdistrplus\_1.2-2   
## [85] ROCR\_1.0-11 nlme\_3.1-168 ggtree\_3.16.0   
## [88] bit64\_4.6.0-1 RcppAnnoy\_0.0.22 GenomeInfoDb\_1.44.0   
## [91] irlba\_2.3.5.1 vipor\_0.4.7 KernSmooth\_2.23-26   
## [94] colorspace\_2.1-1 DBI\_1.2.3 ggrastr\_1.0.2   
## [97] tidyselect\_1.2.1 bit\_4.6.0 compiler\_4.5.0   
## [100] chron\_2.3-62 plotly\_4.10.4 scales\_1.4.0   
## [103] caTools\_1.18.3 lmtest\_0.9-40 digest\_0.6.37   
## [106] goftest\_1.2-3 presto\_1.0.0 spatstat.utils\_3.1-4   
## [109] rmarkdown\_2.29 XVector\_0.48.0 htmltools\_0.5.8.1   
## [112] pkgconfig\_2.0.3 fastmap\_1.2.0 rlang\_1.1.6   
## [115] GlobalOptions\_0.1.2 htmlwidgets\_1.6.4 UCSC.utils\_1.4.0   
## [118] shiny\_1.10.0 farver\_2.1.2 zoo\_1.8-14   
## [121] jsonlite\_2.0.0 BiocParallel\_1.42.1 GOSemSim\_2.34.0   
## [124] R.oo\_1.27.1 magrittr\_2.0.3 GenomeInfoDbData\_1.2.14  
## [127] ggplotify\_0.1.2 dotCall64\_1.2 Rcpp\_1.0.14   
## [130] ape\_5.8-1 proto\_1.0.0 reticulate\_1.42.0   
## [133] sqldf\_0.4-11 stringi\_1.8.7 MASS\_7.3-65   
## [136] plyr\_1.8.9 parallel\_4.5.0 listenv\_0.9.1   
## [139] deldir\_2.0-4 Biostrings\_2.76.0 splines\_4.5.0   
## [142] tensor\_1.5 hash\_2.2.6.3 hms\_1.1.3   
## [145] circlize\_0.4.16 locfit\_1.5-9.12 spatstat.geom\_3.4-1   
## [148] RcppHNSW\_0.6.0 reshape2\_1.4.4 evaluate\_1.0.3   
## [151] tzdb\_0.5.0 foreach\_1.5.2 httpuv\_1.6.16   
## [154] RANN\_2.6.2 polyclip\_1.10-7 clue\_0.3-66   
## [157] future\_1.58.0 scattermore\_1.2 xtable\_1.8-4   
## [160] RSpectra\_0.16-2 tidytree\_0.4.6 later\_1.4.2   
## [163] ragg\_1.4.0 viridisLite\_0.4.2 aplot\_0.2.6   
## [166] beeswarm\_0.4.0 memoise\_2.0.1 cluster\_2.1.8.1   
## [169] timechange\_0.3.0 globals\_0.18.0