R# Load required packages  
library(Seurat)  
library(ggplot2)  
library(dplyr)  
library(patchwork)  
library(viridis)  
library(RColorBrewer)  
library(scales)  
library(ggpubr)  
  
# Set seed for reproducibility  
set.seed(1234)  
  
# -------------------------- Data Loading --------------------------  
# 1. Define sample paths (update these paths before execution)  
sample\_paths <- list(  
 MPR\_in\_PR = "path/to/MPR\_in\_PR/filtered\_feature\_bc\_matrix.h5",  
 non\_MPR\_in\_PR = "path/to/non-MPR\_in\_PR/filtered\_feature\_bc\_matrix.h5",  
 non\_MPR\_in\_SD = "path/to/non-MPR\_in\_SD/filtered\_feature\_bc\_matrix.h5",  
 non\_MPR\_in\_PD = "path/to/non-MPR\_in\_PD/filtered\_feature\_bc\_matrix.h5"  
)  
  
image\_paths <- list(  
 MPR\_in\_PR = "path/to/MPR\_in\_PR/spatial",  
 non\_MPR\_in\_PR = "path/to/non-MPR\_in\_PR/spatial",  
 non\_MPR\_in\_SD = "path/to/non-MPR\_in\_SD/spatial",  
 non\_MPR\_in\_PD = "path/to/non-MPR\_in\_PD/spatial"  
)  
  
sc\_path <- "path/to/scRNAseq/filtered\_feature\_bc\_matrix"  
  
# ------------------------- Data Processing -------------------------  
# 2. Process spatial transcriptomics data  
process\_spatial\_sample <- function(sample\_id) {  
 # Load counts and spatial image  
 counts <- Read10X\_h5(filename = sample\_paths[[sample\_id]])  
 obj <- CreateSeuratObject(counts = counts, project = sample\_id)  
 obj@images[[sample\_id]] <- Load10X\_Spatial(  
 data.dir = image\_paths[[sample\_id]],  
 slice = sample\_id,  
 filter.matrix = FALSE  
 )@images[[sample\_id]]  
   
 # Add metadata  
 obj$sample <- sample\_id  
 obj$pathological\_response <- ifelse(grepl("MPR", sample\_id), "MPR", "non-MPR")  
 obj$subgroup <- gsub("MPR\_in\_|non-MPR\_in\_", "", sample\_id)  
   
 # Quality control  
 obj <- PercentageFeatureSet(obj, pattern = "^MT-", col.name = "percent.mt")  
 obj <- subset(obj, subset = nFeature\_RNA > 200 & nFeature\_RNA < 6000 & percent.mt < 20)  
   
 # Normalization and feature scaling  
 obj <- SCTransform(obj, assay = "RNA", variable.features.n = 3000, verbose = FALSE)  
 obj <- RunPCA(obj, verbose = FALSE)  
 obj <- FindNeighbors(obj, reduction = "pca", dims = 1:20, verbose = FALSE)  
 obj <- FindClusters(obj, resolution = 0.5, verbose = FALSE)  
   
 # Calculate CHCHD3 expression  
 obj <- NormalizeData(obj, assay = "RNA", verbose = FALSE)  
 obj$CHCHD3 <- obj@assays$RNA@data["CHCHD3", ]  
   
 return(obj)  
}  
  
# Process all spatial samples  
spatial\_objects <- lapply(names(sample\_paths), process\_spatial\_sample)  
names(spatial\_objects) <- names(sample\_paths)  
  
# 3. Process single-cell RNA-seq data  
process\_sc\_data <- function() {  
 # Load and filter scRNA-seq data  
 sc\_counts <- Read10X(data.dir = sc\_path)  
 sc\_obj <- CreateSeuratObject(  
 counts = sc\_counts,   
 project = "scRNAseq",   
 min.cells = 3,   
 min.features = 200  
 )  
   
 # Quality control  
 sc\_obj <- PercentageFeatureSet(sc\_obj, pattern = "^MT-", col.name = "percent.mt")  
 sc\_obj <- subset(sc\_obj, subset = nFeature\_RNA > 200 & nFeature\_RNA < 6000 & percent.mt < 20)  
   
 # Normalization and clustering  
 sc\_obj <- SCTransform(sc\_obj, verbose = FALSE)  
 sc\_obj <- RunPCA(sc\_obj, verbose = FALSE)  
 sc\_obj <- FindNeighbors(sc\_obj, dims = 1:20, verbose = FALSE)  
 sc\_obj <- FindClusters(sc\_obj, resolution = 0.5, verbose = FALSE)  
   
 # Cell type annotation with marker genes  
 cell\_markers <- list(  
 CD8\_Tcell = c("CD3D", "CD3E", "CD8A", "CD8B"),  
 CD4\_Tcell = c("CD3D", "CD3E", "CD4"),  
 Macrophage = c("CD68", "CD163", "MRC1"),  
 Bcell = c("CD79A", "CD79B", "MS4A1"),  
 Cancer\_cell = c("EPCAM", "KRT19", "KRT18")  
 )  
   
 sc\_obj <- AddModuleScore(  
 sc\_obj,   
 features = cell\_markers,   
 assay = "SCT",   
 name = names(cell\_markers)  
 )  
   
 # Assign cell types based on highest module score  
 sc\_obj$cell\_type <- colnames(sc\_obj@meta.data)[  
 max.col(sc\_obj@meta.data[, names(cell\_markers)])  
 ]  
 sc\_obj$cell\_type <- gsub("\_1", "", sc\_obj$cell\_type)  
   
 return(sc\_obj)  
}  
  
sc\_obj <- process\_sc\_data()  
  
# ------------------------ Deconvolution Analysis ------------------------  
# 4. Run RCTD for cell type deconvolution on spatial data  
if (!requireNamespace("SpaceRanger", quietly = TRUE)) {  
 remotes::install\_github("dmcable/spacexr")  
}  
library(spacexr)  
  
run\_rctd <- function(spatial\_obj, sc\_reference) {  
 # Prepare reference and query  
 reference <- CreateSPOTSReference(  
 sc\_reference,   
 cell\_types = sc\_reference$cell\_type,   
 nUMI = "nCount\_RNA"  
 )  
   
 # Extract spatial coordinates and counts  
 coords <- GetTissueCoordinates(spatial\_obj, cols = c("imagerow", "imagecol"))  
 counts <- GetAssayData(spatial\_obj, assay = "RNA", slot = "counts")  
   
 # Create RCTD object and run deconvolution  
 rctd <- create.RCTD(coords, counts, reference, max\_cores = 4)  
 rctd <- run.RCTD(rctd, doublet\_mode = "full")  
   
 # Add cell type proportions to spatial object  
 props <- as.data.frame(rctd@results$cell\_type\_proportions)  
 colnames(props) <- paste0("prop\_", colnames(props))  
 spatial\_obj <- AddMetaData(spatial\_obj, metadata = props)  
   
 return(spatial\_obj)  
}  
  
# Run deconvolution on all spatial objects  
spatial\_objects <- lapply(spatial\_objects, function(x) run\_rctd(x, sc\_obj))  
  
# -------------------------- Visualization --------------------------  
# 5. Define visualization parameters  
celltype\_colors <- c(  
 "M2-macrophages" = "#440154FF",  
 "CD3+CD8+ cells" = "#3B528BFF",  
 "Cancer cells" = "#FDE725FF",  
 "CD4\_Tcell" = "#21908CFF",  
 "Bcell" = "#5DC863FF"  
)  
  
chchd3\_colors <- plasma(10)  
cluster\_colors <- c("#440154FF", "#3B528BFF", "#21908CFF", "#5DC863FF", "#FDE725FF",   
 "#FEAD61FF", "#E16462FF", "#B12A90FF", "#6A1B9AFF", "#320E61FF")  
  
# 6. Function to generate spatial panel (CHCHD3 + cell type clusters)  
create\_spatial\_panel <- function(spatial\_obj, sample\_name) {  
 # Plot CHCHD3 expression  
 p\_expr <- SpatialFeaturePlot(  
 spatial\_obj,  
 features = "CHCHD3",  
 pt.size.factor = 1.2,  
 alpha = c(0.7, 1),  
 crop = FALSE  
 ) +  
 scale\_fill\_gradientn(colors = chchd3\_colors, name = "CHCHD3") +  
 ggtitle(sample\_name) +  
 theme(  
 plot.title = element\_text(size = 10, face = "bold", hjust = 0.5),  
 legend.key.size = unit(0.4, "cm"),  
 legend.title = element\_text(size = 7),  
 legend.text = element\_text(size = 6)  
 )  
   
 # Plot spatial clustering with cell type overlay  
 p\_cluster <- SpatialDimPlot(  
 spatial\_obj,  
 group.by = "seurat\_clusters",  
 pt.size.factor = 1.2,  
 alpha = c(0.7, 1),  
 cols = cluster\_colors,  
 crop = FALSE  
 ) +  
 ggtitle("Spatial Clustering Plot") +  
 theme(  
 plot.title = element\_text(size = 10, face = "bold", hjust = 0.5),  
 legend.key.size = unit(0.3, "cm"),  
 legend.text = element\_text(size = 5),  
 legend.title = element\_text(size = 7)  
 )  
   
 return(p\_expr | p\_cluster)  
}  
  
# 7. Generate spatial panels for all samples  
sample\_titles <- c(  
 "MPR\_in\_PR" = "MPR in PR group",  
 "non\_MPR\_in\_PR" = "non-MPR in PR",  
 "non\_MPR\_in\_SD" = "non-MPR in SD",  
 "non\_MPR\_in\_PD" = "non-MPR in PD"  
)  
  
spatial\_panels <- lapply(names(spatial\_objects), function(sample\_id) {  
 create\_spatial\_panel(spatial\_objects[[sample\_id]], sample\_titles[sample\_id])  
})  
names(spatial\_panels) <- names(spatial\_objects)  
  
# 8. Create merged spatial plot (H-K layout)  
combined\_plot <- (spatial\_panels[["MPR\_in\_PR"]] + spatial\_panels[["non\_MPR\_in\_PR"]]) /  
 (spatial\_panels[["non\_MPR\_in\_SD"]] + spatial\_panels[["non\_MPR\_in\_PD"]]) +  
 plot\_layout(heights = c(1, 1)) +  
 plot\_annotation(  
 tag\_levels = "H",  
 tag\_suffix = ")"  
 ) &   
 theme(plot.tag = element\_text(size = 12, face = "bold"))  
  
# 9. Add cell type legend  
celltype\_legend <- get\_legend(  
 ggplot(data.frame(type = names(celltype\_colors), value = 1), aes(x = type, y = value, fill = type)) +  
 geom\_tile() +  
 scale\_fill\_manual(values = celltype\_colors) +  
 theme(  
 legend.position = "bottom",  
 legend.key.size = unit(0.5, "cm"),  
 legend.text = element\_text(size = 8),  
 legend.title = element\_blank()  
 )  
)  
  
# Combine plot with legend  
final\_figure <- combined\_plot +   
 inset\_element(celltype\_legend, left = 0.3, bottom = -0.1, right = 0.7, top = 0, align\_to = "plot")  
  
# -------------------------- Statistical Output --------------------------  
# 10. Generate quantitative data  
extract\_chchd3\_stats <- function() {  
 data.frame(  
 sample = names(spatial\_objects),  
 response = sapply(spatial\_objects, function(x) unique(x$pathological\_response)),  
 subgroup = sapply(spatial\_objects, function(x) unique(x$subgroup)),  
 mean\_CHCHD3 = sapply(spatial\_objects, function(x) mean(x$CHCHD3)),  
 median\_CHCHD3 = sapply(spatial\_objects, function(x) median(x$CHCHD3)),  
 n\_spots = sapply(spatial\_objects, function(x) ncol(x))  
 )  
}  
  
chchd3\_stats <- extract\_chchd3\_stats()  
  
# 11. Save outputs  
ggsave("CHCHD3\_Spatial\_scRNAseq\_Merged\_Figure.tiff",   
 final\_figure, width = 18, height = 16, dpi = 600, compression = "lzw")  
saveRDS(spatial\_objects, "spatial\_deconvolution\_results.rds")  
saveRDS(sc\_obj, "scRNAseq\_annotated\_object.rds")  
write.csv(chchd3\_stats, "CHCHD3\_statistical\_summary.csv", row.names = FALSE)  
  
# 12. Statistical testing (Mann-Whitney U test between MPR and non-MPR)  
chchd3\_data <- do.call(rbind, lapply(spatial\_objects, function(x) {  
 data.frame(  
 CHCHD3 = x$CHCHD3,  
 response = x$pathological\_response,  
 sample = x$sample  
 )  
}))  
  
stat\_test <- wilcox.test(CHCHD3 ~ response, data = chchd3\_data)  
write.csv(data.frame(  
 statistic = stat\_test$statistic,  
 p\_value = stat\_test$p.value,  
 method = stat\_test$method  
), "CHCHD3\_statistical\_test.csv", row.names = FALSE)

R# Spatial Transcriptomic Analysis of Immune Microenvironment and CHCHD3 Expression in Pathological Response Groups  
# Code for Cancer Research submission  
# Last updated: 2025-06-28  
  
# 1. Load Required Packages  
library(Seurat) # v4.4.0+ (Spatial transcriptomics analysis)  
library(ggplot2) # v3.4.0+ (Visualization)  
library(dplyr) # v1.1.0+ (Data manipulation)  
library(patchwork) # v1.1.2+ (Plot combination)  
library(harmony) # v1.0+ (Batch correction)  
library(spacexr) # v1.3.0+ (Cell type deconvolution)  
library(ggpubr) # v0.5.0+ (Statistical visualization)  
library(viridis) # v0.6.2+ (Color scales)  
library(clustree) # v0.5.0+ (Clustering visualization)  
library(matrixStats) # v0.63.0+ (Matrix operations)  
  
  
# 2. Set Global Parameters  
set.seed(1234) # For reproducibility  
options(stringsAsFactors = FALSE)  
WORKING\_DIR <- "/path/to/your/analysis" # Update to your directory  
OUTPUT\_DIR <- file.path(WORKING\_DIR, "output")  
dir.create(OUTPUT\_DIR, recursive = TRUE, showWarnings = FALSE)  
  
  
# 3. Data Loading & Initial Processing  
## 3.1 Sample metadata (MPR/non-MPR groups)  
sample\_metadata <- data.frame(  
 sample\_id = c("MPR\_PR", "nonMPR\_PR", "nonMPR\_SD", "nonMPR\_PD"),  
 response = factor(c("MPR", "non-MPR", "non-MPR", "non-MPR"),   
 levels = c("MPR", "non-MPR")),  
 clinical\_group = factor(c("PR", "PR", "SD", "PD"),   
 levels = c("PR", "SD", "PD")),  
 spatial\_dir = file.path(WORKING\_DIR, "raw\_data",   
 c("MPR\_PR/spatial", "nonMPR\_PR/spatial",   
 "nonMPR\_SD/spatial", "nonMPR\_PD/spatial")),  
 counts\_dir = file.path(WORKING\_DIR, "raw\_data",   
 c("MPR\_PR/filtered\_feature\_bc\_matrix",   
 "nonMPR\_PR/filtered\_feature\_bc\_matrix",   
 "nonMPR\_SD/filtered\_feature\_bc\_matrix",   
 "nonMPR\_PD/filtered\_feature\_bc\_matrix"))  
)  
  
## 3.2 Load and merge spatial datasets  
seurat\_objects <- list()  
for (i in 1:nrow(sample\_metadata)) {  
 seurat\_objects[[i]] <- Load10X\_Spatial(  
 data.dir = sample\_metadata$counts\_dir[i],  
 filename = "filtered\_feature\_bc\_matrix.h5",  
 assay = "Spatial",  
 slice = sample\_metadata$sample\_id[i],  
 filter.matrix = TRUE,  
 to.upper = FALSE  
 )  
 seurat\_objects[[i]] <- AddMetaData(  
 seurat\_objects[[i]],  
 metadata = sample\_metadata[i, c("response", "clinical\_group")]  
 )  
}  
spatial\_combined <- merge(  
 x = seurat\_objects,  
 y = seurat\_objects[2:length(seurat\_objects)],  
 add.cell.ids = sample\_metadata$sample\_id  
)  
  
  
# 4. Quality Control & Normalization  
## 4.1 Calculate QC metrics  
spatial\_combined <- PercentageFeatureSet(spatial\_combined,   
 pattern = "^MT-",   
 col.name = "percent.mt")  
  
## 4.2 Filter low-quality spots  
spatial\_combined <- subset(spatial\_combined,   
 subset = nFeature\_Spatial > 500 &   
 nCount\_Spatial < 30000 &   
 percent.mt < 20)  
  
## 4.3 Normalization and feature selection  
spatial\_combined <- SCTransform(spatial\_combined,   
 assay = "Spatial",   
 verbose = FALSE,  
 return.only.var.genes = FALSE)  
spatial\_combined <- RunPCA(spatial\_combined,   
 assay = "SCT",   
 verbose = FALSE)  
  
## 4.4 Batch correction with Harmony  
spatial\_combined <- RunHarmony(  
 object = spatial\_combined,  
 group.by.vars = "orig.ident",  
 assay.use = "SCT",  
 reduction.save = "harmony"  
)  
  
  
# 5. Spatial Clustering & Dimensionality Reduction  
## 5.1 Determine optimal resolution  
spatial\_combined <- FindNeighbors(spatial\_combined,   
 reduction = "harmony",   
 dims = 1:30)  
resolutions <- seq(0.1, 1.2, by = 0.1)  
spatial\_combined <- FindClusters(spatial\_combined,   
 resolutions = resolutions,   
 verbose = FALSE)  
clustree\_plot <- clustree(spatial\_combined, prefix = "SCT\_snn\_res.")  
ggsave(file.path(OUTPUT\_DIR, "clustree\_resolution.pdf"),   
 clustree\_plot, width = 10, height = 8)  
  
## 5.2 Final clustering (selected resolution = 0.6 based on clustree)  
spatial\_combined <- FindClusters(spatial\_combined,   
 resolution = 0.6,   
 verbose = FALSE)  
  
## 5.3 Run UMAP for visualization  
spatial\_combined <- RunUMAP(spatial\_combined,   
 reduction = "harmony",   
 dims = 1:30,  
 reduction.name = "umap\_harmony")

# 6. Immune Microenvironment Deconvolution  
## 6.1 Prepare single-cell reference (immune cell signatures)  
# Load preprocessed scRNA-seq reference (immune cells from TCGA-BRCA)  
# Reference: https://singlecell.broadinstitute.org/single\_cell/study/SCP1039  
ref\_sc <- readRDS(file.path(WORKING\_DIR, "reference\_data/immune\_ref\_brca.rds"))  
ref\_sc <- subset(ref\_sc, celltype %in% c("CD3+CD8+ T cells", "M2-macrophages", "Cancer cells"))  
  
## 6.2 Run RCTD (Robust Cell Type Deconvolution)  
# Create RCTD object  
myRCTD <- create.RCTD(  
 spatial\_combined@assays$Spatial@counts,  
 spatial\_combined@meta.data[, c("row", "col")],  
 ref\_sc,  
 max\_cores = 8  
)  
  
# Run deconvolution  
myRCTD <- run.RCTD(myRCTD, doublet\_mode = "full")  
  
# Extract cell type proportions  
results <- myRCTD@results  
cell\_type\_props <- as.data.frame(results$cell\_type\_proportions[,   
 c("CD3+CD8+ T cells", "M2-macrophages", "Cancer cells")])  
spatial\_combined <- AddMetaData(spatial\_combined, cell\_type\_props)

# 7. Immune Microenvironment Differences Between Groups  
## 7.1 Statistical testing (MPR vs non-MPR)  
immune\_markers <- c("CD3+CD8+ T cells", "M2-macrophages", "Cancer cells")  
stats\_list <- list()  
for (marker in immune\_markers) {  
 stats <- compare\_means(  
 formula = as.formula(paste(marker, "~ response")),  
 data = spatial\_combined@meta.data,  
 method = "wilcox.test",  
 group.by = "orig.ident" # Account for spatial dependence  
 )  
 stats\_list[[marker]] <- stats  
}  
saveRDS(stats\_list, file.path(OUTPUT\_DIR, "immune\_microenvironment\_stats.rds"))  
  
## 7.2 Spatial visualization of immune cell types  
immune\_plots <- lapply(immune\_markers, function(ct) {  
 SpatialFeaturePlot(spatial\_combined,   
 features = ct,   
 pt.size.factor = 1.2,  
 ncol = 2,  
 cols = viridis(100),  
 facet.by = "response") +  
 theme(legend.position = "right") +  
 labs(title = ct)  
})  
wrap\_plots(immune\_plots, ncol = 1)  
ggsave(file.path(OUTPUT\_DIR, "immune\_cell\_spatial\_distribution.pdf"),   
 width = 14, height = 18)  
  
  
# 8. Target Gene Analysis (CHCHD3)  
## 8.1 Spatial expression of CHCHD3  
target\_gene <- "CHCHD3"  
  
# Normalize expression (log2(CPM + 1))  
spatial\_combined <- NormalizeData(spatial\_combined, assay = "Spatial", verbose = FALSE)  
spatial\_combined <- ScaleData(spatial\_combined, assay = "Spatial", features = target\_gene)  
  
# Spatial plots by pathological response group  
chchd3\_plots <- list()  
for (i in 1:nrow(sample\_metadata)) {  
 sample\_id <- sample\_metadata$sample\_id[i]  
 group\_label <- paste0(sample\_metadata$response[i], " in ", sample\_metadata$clinical\_group[i], " group")  
   
 p <- SpatialFeaturePlot(  
 subset(spatial\_combined, orig.ident == sample\_id),  
 features = target\_gene,  
 pt.size.factor = 1.5,  
 cols = c("lightgray", "blue", "darkblue"),  
 image.alpha = 0.6  
 ) +  
 ggtitle(group\_label) +  
 theme(plot.title = element\_text(hjust = 0.5, size = 12)) +  
 NoLegend()  
   
 # Add clustering overlay  
 cluster\_p <- SpatialDimPlot(  
 subset(spatial\_combined, orig.ident == sample\_id),  
 group.by = "seurat\_clusters",  
 pt.size.factor = 1.5,  
 cols = brewer.pal(8, "Set1"),  
 image.alpha = 0.6  
 ) +  
 ggtitle("Spatial Clustering Plot") +  
 theme(plot.title = element\_text(hjust = 0.5, size = 12)) +  
 NoLegend()  
   
 chchd3\_plots[[i]] <- p | cluster\_p  
}  
  
# Combine plots (matches H-K panels in the manuscript figure)  
combined\_plot <- wrap\_plots(chchd3\_plots, ncol = 2)  
ggsave(file.path(OUTPUT\_DIR, "CHCHD3\_spatial\_expression.pdf"),   
 combined\_plot, width = 16, height = 14)  
  
## 8.2 Statistical analysis of CHCHD3 expression  
chchd3\_stats <- compare\_means(  
 as.formula(paste0(target\_gene, " ~ response")),  
 data = spatial\_combined@meta.data,  
 method = "wilcox.test",  
 group.by = "orig.ident"  
)  
write.csv(chchd3\_stats, file.path(OUTPUT\_DIR, "CHCHD3\_expression\_stats.csv"), row.names = FALSE)

# 9. Save Final Object  
saveRDS(spatial\_combined, file.path(OUTPUT\_DIR, "spatial\_combined\_analysis.rds"))  
  
sessionInfo() # For reproducibility documentation