Single-Cell RNA-seq Seurat Instruction Guide

Detailed Instruction Guide

1. Set Subsampling Size

```
sub_n <- 500
```

- Defines that you'll keep up to 500 cells per sample to speed up downstream analysis and balance sample sizes.

2. Load Each Sample and Subsample

```
seu_list <- lapply(samples_dev, function(dir) {
    id <- basename(dir)
    x <- Read10X(data.dir = dir) %>% CreateSeuratObject(project = id)
    keep <- sample(colnames(x), min(sub_n, ncol(x)))
    subset(x, cells = keep)
})
```

- For each folder in samples_dev:

  1. Read its 10X counts and create a Seurat object (tagged by id).
  2. Randomly pick up to 500 cell barcodes.
  3. Subset to those cells and store the result in seu_list.

3. Merge All Subsampled Objects

```
merged <- merge(
    x = seu_list[[1]],
    y = seu_list[-1],
    add.cell.ids = sapply(seu_list, function(x) x@project.name)
)
merged <- JoinLayers(merged)
```

- Combine sample #1 (base) with samples #2-n into one Seurat object, prefixing barcodes by sample name.

- JoinLayers ensures any extra assays (e.g., ADT, spatial) remain aligned.

4. Label Control vs. Infected

```
samples <- unique(merged$orig.ident)
merged$condition <- ifelse(
    merged$orig.ident == samples[1],
    "control",
    "infected"
)
```

- Marks cells from the first sample as 'control', all others as 'infected'.

5. Compute QC Metrics

```
merged[["percent.mt"]] <- PercentageFeatureSet(merged, pattern = "^MT-")
summary(merged$nFeature_RNA)
summary(merged$percent.mt)
```

- Calculates % mitochondrial reads per cell (genes starting with "MT-").

- Prints summaries of genes per cell (nFeature_RNA) and %mt to decide filtering thresholds.

6. Normalize, Find Variable Genes, Scale, PCA, UMAP

```
dims <- 1:15
merged <- merged %>%
    NormalizeData() %>%
    FindVariableFeatures(nfeatures = 2000) %>%
    ScaleData() %>%
    RunPCA(npcs = 30) %>%
    RunUMAP(dims = dims)
```

- NormalizeData: log-normalizes counts.

- FindVariableFeatures: selects top 2,000 most variable genes.

- ScaleData: centers and scales those genes.

- RunPCA (30 PCs) and RunUMAP (using PCs 1-15) to get a 2D embedding for visualization.

7. Differential Expression (DE) Between Infected vs. Control

```
Idents(merged) <- "condition"
de <- FindMarkers(
    merged,
    ident.1 = "infected",
    ident.2 = "control",
    logfc.threshold = 0.0,
    min.pct = 0.0
)
de$regulation <- ifelse(
    de$avg_log2FC > 0,
    "up_in_infected",
    "down_in_infected"
)
```

- Sets cell identities to the condition column.

- Runs Wilcoxon-based DE test on all genes (no FC or detection-rate filters).

- Adds a 'regulation' label: up_in_infected if avg_log2FC > 0; otherwise down_in_infected.

8. (Optional) Filter DE Results

```
sig_up <- subset(
    de,
    p_val_adj < 0.05 &
    abs(avg_log2FC) >= 0.25 &
    regulation == "up_in_infected"
)
head(sig_up, 10)
```

- Keeps only genes with adjusted p < 0.05, |log2FC| >= 0.25, and up in infected.

Concise Instruction Guide

1. Set Subsampling Size

```
sub_n <- 500
```

- Keep up to 500 cells per sample to balance sizes and speed analysis.

2. Load and Subsample Each Sample

```
seu_list <- lapply(samples_dev, function(dir) {
    id <- basename(dir)
    x <- Read10X(data.dir = dir) %>% CreateSeuratObject(project = id)
    keep <- sample(colnames(x), min(sub_n, ncol(x)))
    subset(x, cells = keep)
})
```

- Read 10X counts into Seurat object, pick up to 500 cells, and store in seu_list.

3. Merge Subsampled Objects

```
merged <- merge(
    x = seu_list[[1]],
    y = seu_list[-1],
    add.cell.ids = sapply(seu_list, function(x) x@project.name)
)
merged <- JoinLayers(merged)
```

- Combine sample #1 (base) with samples #2-n, ensuring unique barcodes and aligned assays.

4. Label Control vs. Infected

```
samples <- unique(merged$orig.ident)
merged$condition <- ifelse(merged$orig.ident == samples[1], "control",
"infected")
```

5. Compute QC Metrics

```
merged[["percent.mt"]] <- PercentageFeatureSet(merged, pattern = "^MT-")
summary(merged$nFeature_RNA)
summary(merged$percent.mt)
```

6. Normalize, Variable Genes, Scale, PCA, UMAP

```
dims <- 1:15
merged <- merged %>%
    NormalizeData() %>%
    FindVariableFeatures(nfeatures = 2000) %>%
    ScaleData() %>%
    RunPCA(npcs = 30) %>%
    RunUMAP(dims = dims)
```

7. Differential Expression (DE)

```
Idents(merged) <- "condition"
de <- FindMarkers(
    merged,
    ident.1 = "infected",
    ident.2 = "control",
    logfc.threshold = 0.0,
    min.pct = 0.0
)
de$regulation <- ifelse(de$avg_log2FC > 0, "up_in_infected", "down_in_infected")
```

8. (Optional) Filter DE

```
sig_up <- subset(
    de,
    p_val_adj < 0.05 &
    abs(avg_log2FC) >= 0.25 &
    regulation == "up_in_infected"
)
head(sig_up, 10)
```