```
############################
# Seurat pre-processing

############################
# Load necessary libraries
library(Seurat)
library(tidyverse)


############################
# Read in mapped data, store 10X data as a list

my_filepaths <- c("~/data/sample_1/filtered_feature_bc_matrix/",
                  "~/data/sample_2/filtered_feature_bc_matrix")    # provide filepaths to cellranger mapping output
my_sample_names <- c("sample1", "sample2")                         # names of samples
my_list <- vector(mode = "list", length = length(my_filepaths))    # list for 10X data storage

for(i in 1:length(my_filepaths)){
  my_initial_data <- Read10X(data.dir = my_filepaths[i])
  my_initial_object <- CreateSeuratObject(counts = my_initial_data, project = my_sample_names[i], assay = "RNA")
  my_initial_object[["mito_genes"]] <- PercentageFeatureSet(my_initial_object, pattern = "^MT-")

  my_list[[i]] <- my_initial_object
}

############################
# Subset 10X data based on predefined cutoffs
# Manually adjust the following cutoffs based on experimental QC
higher_RNA_cutoff = 10000
lower_RNA_cutoff = 100
mito_cutoff = 25

for(i in 1:length(my_filepaths)){
  my_initial_object <- my_list[[i]]
  my_intial_object <- subset(my_initial_object, subset = nFeature_RNA < higher_RNA_cutoff &
                               nFeature_RNA > lower_RNA_cutoff &
                               mito_genes < mito_cutoff)
  my_intial_object <- NormalizeData(my_intial_object, normalization.method = "LogNormalize", scale.factor = 10000)
  my_intial_object <- FindVariableFeatures(my_intial_object, selection.method = "mvp", nfeatures = 2000)
  my_list[[i]] <- my_intial_object
}

############################
# Perform the cannonical correlation analysis
my_object_anchors <- FindIntegrationAnchors(object.list = my_list, dims=1:25)
my_object_combined <- IntegrateData(anchorset = my_object_anchors, dims = 1:25)

DefaultAssay(my_object_combined) <- "integrated"
my_object_combined <- ScaleData(my_object_combined, verbose = FALSE)
my_object_combined <- RunPCA(my_object_combined, npcs = 30, verbose = FALSE)

############################
# Perform the dimensionality reduction and clustering
ElbowPlot(my_object_combined, ndims = 25)
final_dim = 20 # manually select the final dimension to include in dimensionality reduction

my_object_combined <- RunUMAP(my_object_combined, reduction = "pca", dims = 1:final_dim)
my_object_combined <- FindNeighbors(my_object_combined, reduction = "pca", dims = 1:final_dim)
my_object_combined <- FindClusters(my_object_combined, resolution = 0.5)
```