StromalCell_integration

December 25, 2025

1 load data

[ ]:
```
objList <- list.files('/project/sex_cancer/data/data_zenodo', pattern = 'obj',␣
↪full.names = TRUE)
objList
length(objList)
```

[ ]:
```
seuratList <- lapply(objList, function(x){readRDS(x)})
names(seuratList) <- objList %>% gsub('/project/sex_cancer/data/data_zenodo/obj.
↪', '', .) %>% gsub('.rds', '', .)
```

2 extract intersect genes

[ ]:
```
geneList <- lapply(seuratList, function(x){rownames(x)})
geneList_all <- geneList %>% ext_list() %>% unique()
length(geneList_all) ## 65526 genes
geneList_freq13 <- geneList %>% unlist %>% table() %>% as.data.frame() %>%␣
↪subset(Freq == 13) %>% .[,1] %>% ext_list()
length(geneList_freq13) ## 13412 genes
```

3 filter stromal cells

[ ]:
```
metadata_cell <- readRDS("/project/sex_cancer/data/data_zenodo/metadata_cell.
↪rds")
select_cohort <- metadata_cell %>%
                 group_by(Cohort, gCT, SampleType) %>%
                 subset(SampleType == "tumor" & gCT == "Stromal") %>%
                 summarize(Ncell = n(), .groups = 'drop') %>%
                 .$Cohort
```

[ ]:
```
seuratList_Stromal <- lapply(seuratList[select_cohort], function(obj){
                                    obj %>% subset(SampleType == 'tumor') %>%␣
↪subset(gCT == 'Stromal') %>% subset(feature = geneList_freq13)
                                })
names(seuratList_Stromal) <- names(seuratList_Stromal)
```

```
lapply(seuratList_Stromal, function(x){ncol(x)}) %>% do.call(sum, .)

seurat_Stromal <- merge(seuratList_Stromal[[1]], seuratList_Stromal[-1])
seurat_Stromal
```

4 diet stromal cell

[ ]:
```
obj <- seurat_Stromal
## extract unique group info
meta <- obj@meta.data %>% transform(group = paste(SampleID, mCT, sep = '_'))
groupList <- unique(meta$group)
## down-sampling
metaDiet <- lapply(groupList, function(x){
                    groupMeta <- meta %>% subset(group == x)
                    Ncell <- nrow(groupMeta)
                    if(Ncell > 100){
                        groupMeta <- groupMeta[sample(Ncell, 100), ]
                    }
                    return(groupMeta)
                }) %>% do.call(rbind, .)
## filter
obj.diet <- obj %>% subset(cells = rownames(metaDiet))
obj.diet
```

5 sample integration

[ ]:
```
cohortList = unique(ext_list(obj.diet$Cohort))
cohortList
length(cohortList)
```

[ ]:
```
marker_ref <- read.rds("/project/sex_cancer/data/data_zenodo/marker_annotation.
↪rds")
names(marker_ref) <- names(marker_ref) %>% strsplit2(split = "\\.") %>% .[,1]
marker_stromal <- marker_ref[unique(obj.diet$mCT)] %>% unlist() %>% unique()␣
↪%>% intersect(., rownames(obj.diet))
marker_stromal
```

[ ]:
```
obj.anchor <- lapply(cohortList, function(x){
                        obj <- obj.diet %>%
                               subset(Cohort == x) %>%
                               NormalizeData(normalization.method =␣
↪"LogNormalize", scale.factor = 10000, verbose = F) %>%
                               FindVariableFeatures(selection.method = "vst",␣
↪nfeatures = 1000, verbose = F)
```

```
                        VariableFeatures(obj) <- union(marker_stromal,␣
 ↪VariableFeatures(obj))
                        obj <- obj %>% ScaleData(vars.to.regress =␣
 ↪c("nCount_RNA"),verbose = F)
                        return(obj)
                    })
names(obj.anchor) <- cohortList

## FindIntegrationAnchors
obj.anchor <- FindIntegrationAnchors(obj.anchor, dims = 1:30)
obj.anchor <- IntegrateData(anchorset = obj.anchor, dims = 1:30, verbose = F)
DefaultAssay(obj.anchor)

## scale data+runPCA
obj.anchor <- obj.anchor %>%
              ScaleData(verbose = FALSE) %>%
              RunPCA(npcs = 50, verbose = F)
## Clustering
set.seed(486)
select <- 1:(PC_selection(obj.anchor)$PCselect %>% min())
obj.anchor <- obj.anchor %>%
              RunUMAP(reduction = "pca", dims = select, umap.method = "uwot")␣
 ↪%>%
              RunTSNE(reduction = "pca", dims = select)
obj.anchor
```

[ ]:
```
options(repr.plot.height = 5, repr.plot.width = 30)
select <- 'umap'
DimPlot_scCustom(obj.anchor, pt.size = .1, group.by = "gCT", reduction =␣
 ↪select, label = F, label.size = 4, colors_use = pal_igv("default")(51))|
DimPlot_scCustom(obj.anchor, pt.size = .1, group.by = "mCT", reduction =␣
 ↪select, label = TRUE, label.size = 4, colors_use = pal_igv("default")(51))|
DimPlot_scCustom(obj.anchor, pt.size = .1, group.by = "Cohort", reduction =␣
 ↪select, label = F, label.size = 4, colors_use = pal_igv("default")(51))|
DimPlot_scCustom(obj.anchor, pt.size = 1, group.by = "Sex", label = TRUE, label.
 ↪size = 4, colors_use = pal_igv("default")(51))
```

5.1 integration quality evaluation

LISI-based

[ ]:
```
embed <- obj.anchor@reductions$umap@cell.embeddings
meta <- obj.anchor@meta.data
res <- compute_lisi(embed, meta, c('Cohort'), perplexity = 100)
str(res)
data.frame(LISI_mean = mean(res$Cohort), LISI_median = median(res$Cohort))
```

[ ]:
```
options(repr.plot.height = 2, repr.plot.width = 4)
ggplot(res, aes(x = Cohort, y = 1))+
geom_density_ridges(scale = 2, alpha = 0.8, rel_min_height= 0, fill =␣
 ↪"#99806c", color = "#696969",
                    quantile_lines= TRUE, quantiles= 0.5, vline_size=0.3,␣
 ↪vline_linetype= "dashed")+
scale_x_continuous(breaks = c(1, 5, 9), labels = c(1, 5, 9))+
labs(x = "LISI", y = "", title = "Stromal component | integration quality")+
coord_cartesian(expand=TRUE, clip = "off")+
ridge_theme
```

6 save

[ ]:
```
DefaultAssay(obj.anchor) <- "RNA"
obj <- DietSeurat(obj.anchor, counts = TRUE, data = TRUE, scale.data = FALSE,␣
 ↪features = rownames(obj), assays = "RNA", dimreducs = c("pca", "umap"), misc␣
 ↪= FALSE)
saveRDS(obj.anchor, "obj.StromalCell.diet.rds")
```

7 function for use

[ ]:
```
ridge_theme <- theme(panel.background = element_rect(fill = NA),
                     panel.grid.major.y = element_blank(),
                     panel.grid.major.x = element_blank(),
                     plot.margin = margin(t=10,r=10,b=5,l=5,unit = "mm"),
                     legend.position = "none",
                     plot.title = element_text(size = 8, color = "#696969",␣
 ↪family = "Arial", face = "bold", vjust = 2, hjust = 0.5),
                     axis.ticks.length.x = unit(0.3, "mm"),
                     axis.ticks.y = element_blank(),
                     axis.line.x = element_line(colour = "grey40",size = 0.5),
                     axis.line.y = element_blank(),
                     axis.text.x = element_text(size = 6, family = "Arial",␣
 ↪color = "black"),
                     axis.text.y = element_blank(),
                     axis.title = element_text(size = 7, family = "Arial", face␣
 ↪= "bold", color = "black", hjust = 0.5))
```