AML_Galen2019_process

December 25, 2025

1 load data

[ ]:
```
meta_path <- data.frame(meta_path = list.files('/project/sex_cancer/data/
↪AML_Galen2019') %>% .[grepl('anno.txt', .)]) %>%
            mutate(sampleID = strsplit2(meta_path, split = '_')[,2] %>% gsub('.
↪anno.txt', '', .)) %>% .[order(.$sampleID),]
exp_path <- data.frame(exp_path = list.files('project/sex_cancer/data/
↪AML_Galen2019') %>% .[grepl('dem.txt', .)]) %>%
           mutate(sampleID = strsplit2(exp_path, split = '_')[,2] %>% gsub('.
↪dem.txt', '', .)) %>% .[order(.$sampleID),]
file_path <- merge(meta_path, exp_path, by = 'sampleID')
```

[ ]:
```
obj_list <- lapply(1:nrow(file_path), function(x){
                    meta <- fread(paste('/project/sex_cancer/data/AML_Galen2019/
↪', file_path[x, 2], sep = '')) %>% transform(barcode = Cell) %>%␣
↪column_to_rownames('Cell')
                    exp <- fread(paste('/project/sex_cancer/data/AML_Galen2019/
↪', file_path[x, 3], sep = '')) %>% column_to_rownames('Gene') %>% .[,␣
↪rownames(meta)] %>% as.sparse()
                    CreateSeuratObject(counts = exp, assay = 'RNA', meta.data =␣
↪meta, min.cells = 0, min.features = 0, project = 'AML_Galen2019')
                })

## data merge
obj.AML <- merge(obj_list[[1]], obj_list[-1])
obj.AML
```

2 filter sample

[ ]:
```
obj.AML <- obj.AML %>%
           subset(Tissue == 'Bone marrow') %>%  ## remove cellline sample
           subset(time == 'D0' | sample %in% c('BM1', 'BM2', 'BM3', 'BM4')) %>%␣
↪## remove treated samples
           subset(sample != 'BM5') ## remove sorted samples
table(obj.AML$sample, obj.AML@meta.data$Sex)
unique(obj.AML$CellType)
```

3 modify meta.data

[ ]:
```
table(colnames(obj.AML) == rownames(obj.AML@meta.data))
table(obj.AML$`Days from diagnosis`)
```

[ ]:
```
obj.AML@meta.data <- obj.AML@meta.data %>%
                     dplyr::select(-(names(obj.AML@meta.data) %>% .
↪[grepl('Score', .)])) %>%
                     dplyr::rename(c('DonorID' = 'sample', 'SampleID' = 'orig.
↪ident')) %>%
                     transform(Sex = ifelse(Sex == 'Female', 'F', 'M')) %>%
                     transform(SampleType = ifelse(grepl('AML', .$DonorID),␣
↪'tumor', 'normal')) %>%
                     transform(Cohort = 'AML_Galen2019') %>%
                     transform(Chemistry = 'Seq-Well')
```

4 cell type annotation

4.1 assign oCT

[ ]:
```
obj.AML@meta.data <- obj.AML@meta.data %>%
                     transform(oCT = CellType)
```

4.2 assign mCT

[ ]:
```
obj.AML@meta.data <- obj.AML@meta.data %>%
                     mutate(mCT = case_when(CellType %in% c('B', 'Plasma',␣
↪'ProB', 'NK', 'CTL', 'T') ~ 'Lymphoid',
                                            CellType %in% c('earlyEry',␣
↪'lateEry') ~ 'Erythroid',
                                            CellType %in% c('cDC', 'cDC-like',␣
↪'pDC', 'GMP', 'GMP-like', 'Mono', 'Mono-like', 'ProMono', 'ProMono-like') ~␣
↪'Myeloid',
                                            CellType %in% c('HSC', 'HSC-like',␣
↪'Prog', 'Prog-like') ~ 'Undiff',
                                            TRUE ~ 'Others'))
```

4.3 assign gCT

[ ]:
```
obj.AML@meta.data <- obj.AML@meta.data %>%
                     mutate(gCT = case_when(mCT %in% c('Myeloid', 'Undiff') ~␣
↪'Tumor',
                                            mCT %in% c('Lymphoid', 'Erythroid')␣
↪~ 'Immune',
                                            TRUE ~ 'Others'))
```

```r
table(obj.AML$dCT, obj.AML$gCT)
table(obj.AML$gCT, obj.AML$PredictionRefined)
```

4.4 check annotation

check cell type annotation provided in the original research

[ ]:
```r
# pacman::p_load(Seurat, SeuratWrappers, ggpubr, ggsci, RColorBrewer, ggrepel,
 ↪paletteer, scico,extrafont, sysfonts,paletteer,
#                circlize, scico, ComplexHeatmap, ggplot2, dplyr, tidydr,
 ↪extrafont, scCustomize, ggridges, tidyverse,viridis,limma,
#                SingleCellExperiment, hdf5r, magrittr, data.table, COSG)
obj.AML <- readRDS("/home3/sunruya/1_work/12_panTME_sex/data_zenodo_R1/obj.
 ↪AML_Galen2019.rds")
obj.AML
```

[ ]:
```r
## data normalization
obj <- obj.AML %>% NormalizeData(normalization.method = "LogNormalize", scale.
 ↪factor = 10000, verbose = F)
Idents(obj) <- obj$mCT

## load marker list utilized in this study
marker_ref <- readRDS("marker_ref.rds")
unique(obj$mCT)[unique(obj$mCT) %in% names(marker_ref) == F]
```

[ ]:
```r
## check mCT marker expression
marker_mCT <- obj %>%
              cosg(groups = "all", assay = "RNA", slot = "data",
                   mu = 10, ## The penalty factor to penalize gene expression
 ↪in cells not belonging to the cluster of interest
                   n_genes_user = 50, # Number of top ranked genes returned in
 ↪the result
                   remove_lowly_expressed=T, # If TRUE, genes that express a
 ↪percentage of target cells smaller than a specific value (expressed_pct) are
 ↪not considered as marker genes for the target cells. The default value is
 ↪TRUE.
                   expressed_pct=0.1) # If TRUE, genes that express a
 ↪percentage of target cells smaller than a specific value (expressed_pct) are
 ↪not considered as marker genes for the target cells.
marker_mCT
```

[ ]:
```r
## check marker expression
obj.AML <- obj.AML %>% NormalizeData(normalization.method = "LogNormalize",
 ↪scale.factor = 10000, verbose = F)
Idents(obj.AML) <- obj.AML$mCT
```

```
deg_mCT <- obj.AML %>%
           FindAllMarkers(assay = "RNA", slot = "data", test.use = "wilcox",␣
 ↪group.by = "mCT",
                          min.pct = 0.1, logfc.threshold = 0.25, return.thresh␣
 ↪= 0.25, verbose = T, only.pos = FALSE) %>%
           subset(p_val_adj<0.25 & p_val<0.05 & avg_log2FC>0) %>%
           .[order(.$cluster, -.$avg_log2FC),]
```

[ ]:
```
# load marker for annotation (list in supp Table)
marker_mCT <- readRDS("marker_mCT.rds")
unique(obj.AML$mCT)[unique(obj.AML$mCT) %in% names(marker_mCT) == F]
marker_mCT[unique(obj.AML$mCT)]
```

[ ]:
```
deg_mCT <- lapply(unique(obj.AML$mCT), function(x){
                deg_mCT %>%
                subset(gene %in% unlist(marker_CT[[x]]) & cluster == x) %>%
                remove_rownames()
           }) %>%
           do.call(rbind, .) %>%
           dplyr::rename(c("mCT" = "cluster", "marker" = "gene")) %>%
           mutate(Cohort = "ccRCC_Hu2024")
deg_mCT %>% head(n = 1)

write.csv(deg_mCT, "anno_mCT/MarkerExpression_AML_Galen2019.csv", row.names =␣
 ↪FALSE, quote = FALSE)
```

5 UMAP visualization

[ ]:
```
obj.AML <- obj.AML %>%
           NormalizeData(normalization.method = "LogNormalize", scale.factor =␣
 ↪10000, verbose = F) %>%
           FindVariableFeatures(selection.method = "vst", nfeatures = 3000,␣
 ↪verbose = F) %>%
           ScaleData(vars.to.regress = c("nCount_RNA"), verbose = F) %>%
           RunPCA(verbose = F)

options(repr.plot.height = 3, repr.plot.width = 6)
DimPlot(object = obj.AML, reduction = "pca", group.by = "SampleID", cols =␣
 ↪paletteer_d("ggsci::default_igv"))
PC_selection(obj.AML)
```

[ ]:
```
obj.AML <- obj.AML %>% RunHarmony(group.by.vars = "SampleID", plot_convergence␣
 ↪= TRUE)
## cluster
nPC <- min(PC_selection_harmony(obj.AML)$PCselect)
nPC
```

```
obj.AML <- obj.AML %>%
           RunUMAP(reduction = "harmony", dims = 1:nPC, umap.method = "uwot")␣
 ↪%>%
           RunTSNE(reduction = "harmony", dims = 1:nPC)
```

[ ]:
```
options(repr.plot.height = 5, repr.plot.width = 25)
select <- 'umap'
DimPlot_scCustom(obj.AML, pt.size = .1, group.by = "gCT", reduction = select,␣
 ↪label = TRUE, label.size = 4, colors_use = pal_igv("default")(51))|
DimPlot_scCustom(obj.AML, pt.size = .1, group.by = "mCT", reduction = select,␣
 ↪label = TRUE, label.size = 4, colors_use = pal_igv("default")(51))|
DimPlot_scCustom(obj.AML, pt.size = .1, group.by = "oCT", reduction = select,␣
 ↪label = TRUE, label.size = 4, colors_use = pal_igv("default")(51))
```

[ ]:
```
obj.AML@meta.data[,c('SampleID', 'SampleType', 'Sex')] %>% .[!duplicated(.
 ↪$SampleID),] %$% table(.$SampleType, .$Sex)
obj.AML@meta.data %$% table(.$SampleType)
```

6 save

[ ]:
```
saveRDS(obj.AML, 'obj.AML.use.rds')
```