* **­­­­Building STAR index:**

STAR

--runMode genomeGenerate

--runThreadN 10

--genomeDir STAR\_MM99

--genomeFastaFiles Mus\_musculus.GRCm38.dna.primary\_assembly.fa

--limitGenomeGenerateRAM 50000000000

--genomeSAindexNbases 14

--genomeChrBinNbits 18

--genomeSAsparseD 9

* **Yaml file for sample processing with zUMIs:**

project: Sample

sequence\_files:

file1:

name: PATH/Sample\_R1.fastq.gz

base\_definition: cDNA(1-61)

file2:

name: PATH/Sample\_R2.fastq.gz

base\_definition: BC(1-8)

file3:

name: PATH/Sample\_R3.fastq.gz

base\_definition: BC(1-8)

file4:

name: PATH/Sample\_R4.fastq.gz

base\_definition:

- BC(1-8)

- UMI(9-14)

reference:

STAR\_index: PATH/STAR\_MM99

GTF\_file: PATH/Mus\_musculus.GRCm38.99.gtf

additional\_STAR\_params: ''

additional\_files: ~

out\_dir: PATH

num\_threads: 8

mem\_limit: 0

filter\_cutoffs:

BC\_filter:

num\_bases: 1

phred: 20

UMI\_filter:

num\_bases: 1

phred: 20

barcodes:

barcode\_num: ~

barcode\_file: ~

automatic: yes

BarcodeBinning: 0

nReadsperCell: 100

counting\_opts:

introns: yes

downsampling: '0'

strand: 0

Ham\_Dist: 0

velocyto: yes

primaryHit: no

twoPass: yes

make\_stats: yes

which\_Stage: Filtering

Rscript\_exec: Rscript

STAR\_exec: STAR

pigz\_exec: pigz

samtools\_exec: samtools

zUMIs\_directory: PATH

read\_layout: SE

* **Doublet filtering in Scanpy with the Scrublet package:**

adata = adata [(adata.obs["doublet\_scores"] < 0.2) & (adata.obs["n\_genes"] >1200), :]

General filtering in Scanpy:

adata.obs['n\_counts'] = adata.X.sum(axis=1)

adata.var['mt'] = adata.var\_names.str.startswith('mt-')

adata.var["ribo"] = adata.var\_names.str.startswith(("Rps","Rpl"))

adata.var["Rps2"] = adata.var\_names.str.startswith(("Rp2"))

sc.pp.calculate\_qc\_metrics(adata, qc\_vars=['mt', 'ribo',"Rps2"], percent\_top=None, log1p=False, inplace=True)

adata = adata[adata.obs['pct\_counts\_mt'] < 12, :]

adata = adata[adata.obs['pct\_counts\_mt'] > 2, :]

adata = adata[adata.obs['pct\_counts\_ribo'] < 10, :]

adata = adata[adata.obs['pct\_counts\_ribo'] >2.5, :]

sc.pp.filter\_cells(adata, min\_genes=800)

sc.pp.filter\_cells(adata, max\_counts=8000)

* **Data normalization, regression and scaling with Scanpy:**

sc.pp.normalize\_total(adata, target\_sum=1e4)

sc.pp.log1p(adata)

adata.raw = adata

sc.pp.highly\_variable\_genes(adata, min\_mean=0.0125, max\_mean=3, min\_disp=0.5)

sc.pp.regress\_out(adata, ['n\_counts','pct\_counts\_ribo','pct\_counts\_mt', "pct\_counts\_Rps2"])

sc.pp.scale(adata, max\_value=10)

sc.tl.pca(adata, svd\_solver='arpack')

* **UMAP dimensional reductionin Scanpy:**

sc.pp.neighbors(adata, n\_neighbors=20, n\_pcs=30)

sc.tl.umap(adata, min\_dist = 0.45)

* **Leiden cluster computation in Scanpy:**

sc.tl.leiden(adata ,resolution = 2)

* **Differential gene expression identification with Seurat v3:**

adata\_table <- read.table("Matrix.csv", sep = ',’, header = T, row.names=1,as.is = T)

adata <- CreateSeuratObject(counts = adata\_table , min.cells = 3, assay = "RNA",names.field = 2,

names.delim = ".",

project = "iETX ")

adata[["percent.mt"]] <- PercentageFeatureSet(object = adata, pattern = "^mt-")

adata <- NormalizeData(object = adata )

adata <- FindVariableFeatures(object = adata, selection.method = "vst", nfeatures = 2000)

adata<- ScaleData(object = adata, vars.to.regress = c("percent.mt"))

adata <- RunPCA(object = adata, features = VariableFeatures(object = adata))

adata <- RunUMAP(object = adata,reduction = "pca", dims = 1:21, n\_neighbors=100, min\_dist = 1) %>% FindNeighbors(reduction = "pca", dims = 1:21) %>% FindClusters(resolution = 0.5) %>% identity()

DimPlot(adata)

data <- read.csv("Adataluca\_whitelist\_annotation\_new.csv", header=TRUE)

DEGs<- FindMarkers(object = adata, ident.1 = "Ident1", ident.2 = "Ident2", logfc.threshold = 0.001, min.pct = 0.001)