2-RNAvelocity data preparation

BAI Qiang[*]

2021-10-07 14:01:14 +0200

Contents

[*] University Liege, mail qiang.bai@uliege.be

1 Description

Single-cell

2 Data preparation for whole dataset

2.1 Load data and packages

```
library(Seurat)
library(ggplot2)
library(dplyr)
library(loomR)
library(tidyverse)

load(file = "../../Data/Objects/Combined.integrated.rds")
seurat.combined <-Combined.integrated
```

2.2 Prepare data RNAvelocyto

Prepare individual Seurat objects for each sample.

```
list.name.so <- unique(seurat.combined$Sample)
list.name.sample <- list.name.so

for (i in 1:length(list.name.so)) {
  so <- seurat.combined[, seurat.combined$Sample == list.name.so[i]]
  assign(paste(list.name.sample[i], "seuratObject", sep = "."), so)
}

list.name.so <- paste(list.name.sample, "seuratObject", sep = ".")

obj.list <- list()
for (name.so in list.name.so) {
  obj.list <- c(obj.list, get(name.so))
}
list.name.so <- sub("␣", "_", list.name.so)
names(obj.list) <- list.name.so
```

2.2.1 Generate loom files:

> The intermediate loom files were too big to be uploaded to the platform but they can be produced by the following steps.

We counted spliced, unspliced and ambiguous transcripts using velocyto command-line tool (http://velocyto.org)[1].

For each sample, the following code was used to generate the loom file:

```
velocyto run  -b "${sampleID}/outs/filtered_feature_bc_matrix/barcodes.tsv.gz" \
-o "outputDir/${sampleID}.loom" \
"${sampleID}/outs/possorted_genome_bam.bam" \
-m mm10_rmsk.gtf \
/GRCm38/genes/genes.gtf
```

- ${sampleID} is the sample ID.
- ${sampleID}/outs is the output directory of CellRanger.
- ${sampleID}/outs/possorted_genome_bam.bam is the BAM file generated from CellRanger.
- /refdata-cellranger-GRCh38-3.0.0/genes/genes.gtf is the gene reference used for Cellranger counts.

2.2.2 Read loom files and prepare cellnames

Read loom files and create loom objects under sample names

```
list.path.loom <- list.path.loom[-1] # remove the first entry which is parent directory

list.name.loom <- basename(list.path.loom)
list.name.loom <- str_replace(list.name.loom, pattern = "-", replacement = "_")
list.path.loom <- list.files(list.path.loom, pattern = "\\.loom$", full.names = TRUE)

for (i in 1:length(list.name.loom)) {
  assign(make.names(list.name.loom[i]), read.loom.matrices(list.path.loom[i]))
}
```

```
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
```

```
list.name.loom <- make.names(list.name.loom)
```

Make cell names consistent in both loom objects and Seurat objects.

```
# A prefix was added to each cell in Seurat objects during merge step. We should also add prefix to each cell.

sample.names <- str_remove(list.name.loom, pattern = ".loom")

# find the prefix from seurat object.
prefix <- sapply(paste0(sample.names,".seuratObject"),
                 function(x) unique(matrix(unlist(strsplit(colnames(get(x)), split = "_")), nrow = 2)[2, ]))

# Add prefix to cellnames.
source("~/Desktop/velocyto/Script/aggregateLoom.R")
for (i in 1:length(list.name.loom)) {
  loom <- get(list.name.loom[i])
  assign(list.name.loom[i], value = aggregateLoom(loom, Ori.ID = prefix[i], prefix = FALSE))
}
```

2.2.3 Filter cellnames and feature names in loom with Seurat gene/cell list

As the Seurat object contains only filtered cells and genes, the genes and cells in loom files should be also filtered.

```
source("~/Desktop/velocyto/Script/filterLoom.R")

ldat.list <- list()

for (name.sample in list.name.sample) {
  obj.name <- paste(name.sample, "seuratObject", sep = ".")
  loom.name <- paste(name.sample, "loom", sep = ".")
  ldat.name <- paste(name.sample, "ldat", "filtered", sep = ".")

  assign(ldat.name,
         value = filterLoom(loomObj = get(loom.name),
                            geneList = rownames(obj.list[[obj.name]]),
                            cellList = colnames(obj.list[[obj.name]])))
}
```

```
## [1] "Following genes are not in the loom gene list:"
##  [1] "Ptp4a1.1"     "Arhgef4.1"    "Sept2.1"      "Gm28040.1"    "
   Zc3h11a.1"
##  [6] "Ndor1.1"      "Jakmip1.1"    "Fam220a.1"    "Olfr290.1"    "Aldoa
   .1"
## [11] "Ddit3.1"      "Dpep2.1"      "Chtf8.1"      "St6galnac2.1" "
   Vmn1r216.1"
## [16] "Nnt.1"        "Ighv5-8.1"    "Ighv1-13.1"   "Gcat.1"       "Atp5o
   .1"
## [21] "Ggnbp1.1"     "Pcdha11.1"    "Arhgap26.1"   "Hdhd2.1"      "
   Sssca1.1"
## [26] "Fam205a2.1"   "Ccl21b.1"     "Il11ra2.1"    "Ccl21b.2"     "
   Gm3286.1"
## [31] "Ccl27.1"      "Il11ra2.2"    "Ccl19.1"      "Ccl21a.1"
## [1] "Following genes are not in the loom gene list:"
##  [1] "Ptp4a1.1"     "Arhgef4.1"    "Sept2.1"      "Gm28040.1"    "
   Zc3h11a.1"
##  [6] "Ndor1.1"      "Jakmip1.1"    "Fam220a.1"    "Olfr290.1"    "Aldoa
   .1"
## [11] "Ddit3.1"      "Dpep2.1"      "Chtf8.1"      "St6galnac2.1" "
   Vmn1r216.1"
## [16] "Nnt.1"        "Ighv5-8.1"    "Ighv1-13.1"   "Gcat.1"       "Atp5o
   .1"
## [21] "Ggnbp1.1"     "Pcdha11.1"    "Arhgap26.1"   "Hdhd2.1"      "
   Sssca1.1"
## [26] "Fam205a2.1"   "Ccl21b.1"     "Il11ra2.1"    "Ccl21b.2"     "
   Gm3286.1"
## [31] "Ccl27.1"      "Il11ra2.2"    "Ccl19.1"      "Ccl21a.1"
## [1] "Following genes are not in the loom gene list:"
##  [1] "Ptp4a1.1"     "Arhgef4.1"    "Sept2.1"      "Gm28040.1"    "
   Zc3h11a.1"
##  [6] "Ndor1.1"      "Jakmip1.1"    "Fam220a.1"    "Olfr290.1"    "Aldoa
   .1"
## [11] "Ddit3.1"      "Dpep2.1"      "Chtf8.1"      "St6galnac2.1" "
   Vmn1r216.1"
```

```
## [16] "Nnt.1"         "Ighv5-8.1"     "Ighv1-13.1"    "Gcat.1"        "Atp5o
   .1"
## [21] "Ggnbp1.1"      "Pcdha11.1"     "Arhgap26.1"    "Hdhd2.1"       "
   Sssca1.1"
## [26] "Fam205a2.1"    "Ccl21b.1"      "Il11ra2.1"     "Ccl21b.2"      "
   Gm3286.1"
## [31] "Ccl27.1"       "Il11ra2.2"     "Ccl19.1"       "Ccl21a.1"
## [1] "Following genes are not in the loom gene list:"
##  [1] "Ptp4a1.1"      "Arhgef4.1"     "Sept2.1"       "Gm28040.1"     "
   Zc3h11a.1"
##  [6] "Ndor1.1"       "Jakmip1.1"     "Fam220a.1"     "Olfr290.1"     "Aldoa
   .1"
## [11] "Ddit3.1"       "Dpep2.1"       "Chtf8.1"       "St6galnac2.1" "
   Vmn1r216.1"
## [16] "Nnt.1"         "Ighv5-8.1"     "Ighv1-13.1"    "Gcat.1"        "Atp5o
   .1"
## [21] "Ggnbp1.1"      "Pcdha11.1"     "Arhgap26.1"    "Hdhd2.1"       "
   Sssca1.1"
## [26] "Fam205a2.1"    "Ccl21b.1"      "Il11ra2.1"     "Ccl21b.2"      "
   Gm3286.1"
## [31] "Ccl27.1"       "Il11ra2.2"     "Ccl19.1"       "Ccl21a.1"
## [1] "Following genes are not in the loom gene list:"
##  [1] "Ptp4a1.1"      "Arhgef4.1"     "Sept2.1"       "Gm28040.1"     "
   Zc3h11a.1"
##  [6] "Ndor1.1"       "Jakmip1.1"     "Fam220a.1"     "Olfr290.1"     "Aldoa
   .1"
## [11] "Ddit3.1"       "Dpep2.1"       "Chtf8.1"       "St6galnac2.1" "
   Vmn1r216.1"
## [16] "Nnt.1"         "Ighv5-8.1"     "Ighv1-13.1"    "Gcat.1"        "Atp5o
   .1"
## [21] "Ggnbp1.1"      "Pcdha11.1"     "Arhgap26.1"    "Hdhd2.1"       "
   Sssca1.1"
## [26] "Fam205a2.1"    "Ccl21b.1"      "Il11ra2.1"     "Ccl21b.2"      "
   Gm3286.1"
## [31] "Ccl27.1"       "Il11ra2.2"     "Ccl19.1"       "Ccl21a.1"
## [1] "Following genes are not in the loom gene list:"
##  [1] "Ptp4a1.1"      "Arhgef4.1"     "Sept2.1"       "Gm28040.1"     "
   Zc3h11a.1"
##  [6] "Ndor1.1"       "Jakmip1.1"     "Fam220a.1"     "Olfr290.1"     "Aldoa
   .1"
## [11] "Ddit3.1"       "Dpep2.1"       "Chtf8.1"       "St6galnac2.1" "
   Vmn1r216.1"
## [16] "Nnt.1"         "Ighv5-8.1"     "Ighv1-13.1"    "Gcat.1"        "Atp5o
   .1"
## [21] "Ggnbp1.1"      "Pcdha11.1"     "Arhgap26.1"    "Hdhd2.1"       "
   Sssca1.1"
## [26] "Fam205a2.1"    "Ccl21b.1"      "Il11ra2.1"     "Ccl21b.2"      "
   Gm3286.1"
## [31] "Ccl27.1"       "Il11ra2.2"     "Ccl19.1"       "Ccl21a.1"
```

All the samples have the same unfound genes (34 genes with redundant symbols). Why? They were mapped in different transcriptome build?

Remove the redundant genes:

```
for (name.sample in list.name.sample) {
obj.name <- paste(name.sample, "seuratObject", sep = ".")
loom.name <- paste(name.sample, "loom", sep = ".")
ldat.name <- paste(name.sample, "ldat", "filtered", sep = ".")
genes.toRemove <- get(paste(name.sample, "ldat.filtered", sep = "."))

genes <- rownames(obj.list[[obj.name]])
genes.new <- genes[-which(genes %in% genes.toRemove)]

assign(ldat.name,
       value = filterLoom(loomObj = get(loom.name),
                          geneList = genes.new,
                          cellList = colnames(obj.list[[obj.name]]) ))

}
```

ALL FINE!

Make list of Seurat objects and ldat objects, each under ther sample same.

```
obj <- list()

for (name.sample in list.name.sample) {
  obj.name <- paste(name.sample, "seuratObject", sep = ".")
  ldat.name <- paste(name.sample, "ldat", "filtered", sep = ".")

  tmp <- list(ldat = get(ldat.name), seurat = get(obj.name) )

  obj[[name.sample]] <- tmp
}
```

Save for other analyses.

```
saveRDS(obj, file = "./obj.list.loom_surat.Rds")
```

2.2.4 Group loom/Seurat objects by treatment

Merge all Seurat objects to one, with only filtered cells. Merge all ldat objects to one, with only filtered cells.

```
obj.all <- obj
# now create merged seurat object and loom data.

# 1. merged seurat object.
list.name.sample <- names(obj.all)

seurat.all <- list()
ldat.all <- list()

for (sample.name in list.name.sample) {
  obj <- obj.all[[sample.name]]

  seurat.all[[sample.name]] <- obj[["seurat"]]
  ldat.all[[sample.name]] <- obj[["ldat"]]
}
```

```

cellnames <- character()
for (sample.name in list.name.sample) {
  obj <- seurat.all[[sample.name]]

  cellnames <- append(cellnames, colnames(obj))
}

seurat.merge <- seurat.combined[ , cellnames]
seurat.merge
```

```
## An object of class Seurat
## 37418 features across 29687 samples within 4 assays
## Active assay: RNA (22597 features, 0 variable features)
##  3 other assays present: HTO, SCT, integrated
##  2 dimensional reductions calculated: pca, umap
```

```
# 2. merged loom data;
# source("~/Desktop/velocyto/Script/aggregateLoom.R")

i=1
for (sample.name in list.name.sample) {

  obj <- ldat.all[[sample.name]]
  if (i==1) {
    spliced <- obj$spliced
    unspliced <- obj$unspliced
    ambiguous <- obj$ambiguous
  } else {
    spliced <- cbind(spliced, obj$spliced)
    unspliced <- cbind(unspliced, obj$unspliced) # note: previous code here was wrong.
    ambiguous <- cbind(ambiguous, obj$ambiguous) # note: previous code here was wrong.
  }

  i=1+1
}

ldat.merge <- list(spliced=spliced,
                   unspliced=unspliced,
                   ambiguous=ambiguous)
```

Now separate them by group.

```
groupBy <- "Sample"
sample.groupBy <- unique(seurat.merge@meta.data[[groupBy]])

obj <- list()
for (sample.name in sample.groupBy) {
  seurat <- seurat.merge[ , seurat.merge@meta.data[[groupBy]] == sample.name]
  cellnames <- colnames(seurat)
  ldat <- list(spliced = ldat.merge$spliced[, cellnames],
               unspliced = ldat.merge$unspliced[, cellnames],
               ambiguous = ldat.merge$ambiguous[, cellnames])
  obj[[sample.name]] <- list(ldat=ldat,
                             seurat=seurat)
}
```

OPTIONAL: save data for other presentations

```
saveRDS(obj, file = "./obj_group_by_group.list.loom_surat.Rds")
```

2.3 Prepare for scVelo analysis

2.3.1 Correct NA in Seurat Metadata

```
obj <- lapply(obj, function(x)
{
  obj <- x[["seurat"]]
  ldat <- x[["ldat"]]
  for(j in 1:ncol(obj@meta.data)){
    if(is.factor(obj@meta.data[,j]) == T){
      obj@meta.data[,j][is.na(obj@meta.data[,j])] <- "N.A"
    }
    if(is.character(obj@meta.data[,j]) == T){
      obj@meta.data[,j][is.na(obj@meta.data[,j])] <- "N.A"
    }
  }
  x[["seurat"]] <- obj
  x[["ldat"]] <- ldat

  return(x)
}
)
```

2.3.2 Make loom file from Seurat/loom object

To facilitate the work, we optimized Seurat `Convert` function to merge a Seurat/Loom list to one Loom file, containing the var matrix with spliced, unspliced layers and obs with all embedding, tsne, umap, pca, clustering, etc. A Loom file issue from the function above `Convert.seurat_loom` will be saved in the current working folder.

```
library(loomR)
source("~/Desktop/velocyto/Script/Convert_Seurat_loom.R")

for (sample.name in sample.groupBy) {
  obj.sl <- obj[[sample.name]]
  pfile <- Convert.seurat_loom(from = obj.sl, to = "loom", filename =
     paste0(sample.name, ".loom") )
  pfile$close_all()
}
```

3 Data preparation for zoom dataset (memory subset with 15 clusters)

3.1 Load data and packages

```r
require(Seurat)
require(ggplot2)
require(dplyr)
require(loomR)
require(tidyverse)

rm(list = ls())
load(file = "../../Data/Objects/Zoom.integrated.rds")
seurat.combined <-Zoom.integrated
```

3.2 Prepare data RNAvelocyto

Prepare individual Seurat objects for each sample.

```r
list.name.so <- unique(seurat.combined$Sample)
list.name.sample <- list.name.so

for (i in 1:length(list.name.so)) {
  so <- seurat.combined[, seurat.combined$Sample == list.name.so[i]]
  assign(paste(list.name.sample[i], "seuratObject", sep = "."), so)
}

list.name.so <- paste(list.name.sample, "seuratObject", sep = ".")

obj.list <- list()
for (name.so in list.name.so) {
  obj.list <- c(obj.list, get(name.so))
}
list.name.so <- sub("␣", "_", list.name.so)
names(obj.list) <- list.name.so
```

3.2.1 Generate loom files:

> The intermediate loom files were too big to be uploaded to the platform but they can be produced by the following steps.

We counted spliced, unspliced and ambiguous transcripts using velocyto command-line tool (http://velocyto.org)[1].

For each sample, the following code was used to generate the loom file:

```
velocyto run  -b "${sampleID}/outs/filtered_feature_bc_matrix/barcodes.tsv.gz" \
-o "outputDir/${sampleID}.loom" \
"${sampleID}/outs/possorted_genome_bam.bam" \
-m mm10_rmsk.gtf \
/GRCm38/genes/genes.gtf
```

- `${sampleID}` is the sample ID.

- `${sampleID}/outs` is the output directory of CellRanger.
- `${sampleID}/outs/possorted_genome_bam.bam` is the BAM file generated from CellRanger.
- `/refdata-cellranger-GRCh38-3.0.0/genes/genes.gtf` is the gene reference used for Cellranger counts.

3.2.2 Read loom files and prepare cellnames

Read loom files and create loom objects under sample names

```r
list.path.loom <- list.path.loom[-1] # remove the first entry which is parent directory

list.name.loom <- basename(list.path.loom)
list.name.loom <- str_replace(list.name.loom, pattern = "-", replacement = "_")
list.path.loom <- list.files(list.path.loom, pattern = "\\.loom$", full.names = TRUE)

for (i in 1:length(list.name.loom)) {
  assign(make.names(list.name.loom[i]), read.loom.matrices(list.path.loom[i]))
}
```

```
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
## reading loom file via hdf5r...
```

```r
list.name.loom <- make.names(list.name.loom)
```

Make cell names consistent in both loom objects and Seurat objects.

```r
# A prefix was added to each cell in Seurat objects during merge step. We should also add prefix to each cell.

sample.names <- str_remove(list.name.loom, pattern = ".loom")

# find the prefix from seurat object.
prefix <- sapply(paste0(sample.names,".seuratObject"),
                 function(x) unique(matrix(unlist(strsplit(colnames(get(x)), split = "_")), nrow = 2)[2, ]))

# Add prefix to cellnames.
source("~/Desktop/velocyto/Script/aggregateLoom.R")
for (i in 1:length(list.name.loom)) {
  loom <- get(list.name.loom[i])
  assign(list.name.loom[i], value = aggregateLoom(loom, Ori.ID = prefix[i], prefix = FALSE))
}
```

3.2.3 Filter cellnames and feature names in loom with Seurat gene/cell list

As the Seurat object contains only filtered cells and genes, the genes and cells in loom files should be also filtered.

```r
source("~/Desktop/velocyto/Script/filterLoom.R")

ldat.list <- list()

for (name.sample in list.name.sample) {
  obj.name <- paste(name.sample, "seuratObject", sep = ".")
  loom.name <- paste(name.sample, "loom", sep = ".")
  ldat.name <- paste(name.sample, "ldat", "filtered", sep = ".")

  assign(ldat.name,
         value = filterLoom(loomObj = get(loom.name),
                            geneList = rownames(obj.list[[obj.name]]),
                            cellList = colnames(obj.list[[obj.name]])))
}
```

```
## [1] "Following genes are not in the loom gene list:"
## [1] "Fam220a.1"
## [1] "Following genes are not in the loom gene list:"
## [1] "Fam220a.1"
## [1] "Following genes are not in the loom gene list:"
## [1] "Fam220a.1"
## [1] "Following genes are not in the loom gene list:"
## [1] "Fam220a.1"
## [1] "Following genes are not in the loom gene list:"
## [1] "Fam220a.1"
## [1] "Following genes are not in the loom gene list:"
## [1] "Fam220a.1"
```

All the samples have the same unfound genes (1 gene with redundant symbols). Fam220a.1

Remove the redundant genes:

```r
for (name.sample in list.name.sample) {
obj.name <- paste(name.sample, "seuratObject", sep = ".")
loom.name <- paste(name.sample, "loom", sep = ".")
ldat.name <- paste(name.sample, "ldat", "filtered", sep = ".")
genes.toRemove <- get(paste(name.sample, "ldat.filtered", sep = "."))

genes <- rownames(obj.list[[obj.name]])
genes.new <- genes[-which(genes %in% genes.toRemove)]

assign(ldat.name,
       value = filterLoom(loomObj = get(loom.name),
                          geneList = genes.new,
                          cellList = colnames(obj.list[[obj.name]]) ))

}
```

ALL FINE!

Make list of Seurat objects and ldat objects, each under ther sample same.

```
obj <- list()

for (name.sample in list.name.sample) {
  obj.name <- paste(name.sample, "seuratObject", sep = ".")
  ldat.name <- paste(name.sample, "ldat", "filtered", sep = ".")

  tmp <- list(ldat = get(ldat.name), seurat = get(obj.name) )

  obj[[name.sample]] <- tmp
}
```

Save for other analyses.

```
saveRDS(obj, file = "./obj_zoom.list.loom_surat.Rds")
```

3.2.4 Group loom/Seurat objects by treatment

Merge all Seurat objects to one, with only filtered cells. Merge all ldat objects to one, with only filtered cells.

```
obj.all <- obj
# now create merged seurat object and loom data.

# 1. merged seurat object.
list.name.sample <- names(obj.all)

seurat.all <- list()
ldat.all <- list()

for (sample.name in list.name.sample) {
  obj <- obj.all[[sample.name]]

  seurat.all[[sample.name]] <- obj[["seurat"]]
  ldat.all[[sample.name]] <- obj[["ldat"]]
}

cellnames <- character()
for (sample.name in list.name.sample) {
  obj <- seurat.all[[sample.name]]

  cellnames <- append(cellnames, colnames(obj))
}

seurat.merge <- seurat.combined[ , cellnames]
seurat.merge
```

```
## An object of class Seurat
## 35551 features across 8198 samples within 4 assays
## Active assay: integrated (2000 features, 2000 variable features)
##  3 other assays present: RNA, HTO, SCT
##  2 dimensional reductions calculated: pca, umap
```

```
# 2. merged loom data;
# source("~/Desktop/velocyto/Script/aggregateLoom.R")
```

```
i=1
for (sample.name in list.name.sample) {

  obj <- ldat.all[[sample.name]]
  if (i==1) {
    spliced <- obj$spliced
    unspliced <- obj$unspliced
    ambiguous <- obj$ambiguous
  } else {
    spliced <- cbind(spliced, obj$spliced)
    unspliced <- cbind(unspliced, obj$unspliced) # note: previous code here was wrong.
    ambiguous <- cbind(ambiguous, obj$ambiguous) # note: previous code here was wrong.
  }

  i=1+1
}

ldat.merge <- list(spliced=spliced,
                   unspliced=unspliced,
                   ambiguous=ambiguous)
```

Now separate them by group.

```
groupBy <- "Sample"
sample.groupBy <- unique(seurat.merge@meta.data[[groupBy]])

obj <- list()
for (sample.name in sample.groupBy) {
  seurat <- seurat.merge[ , seurat.merge@meta.data[[groupBy]] == sample.name]
  cellnames <- colnames(seurat)
  ldat <- list(spliced = ldat.merge$spliced[, cellnames],
               unspliced = ldat.merge$unspliced[, cellnames],
               ambiguous = ldat.merge$ambiguous[, cellnames])
  obj[[sample.name]] <- list(ldat=ldat,
                             seurat=seurat)
}
```

OPTIONAL: save data for other presentations

```
saveRDS(obj, file = "./obj.zoom_group_by_group.list.loom_surat.Rds")
```

3.3 Prepare for scVelo analysis

3.3.1 Correct NA in Seurat Metadata

```
obj <- lapply(obj, function(x)
{
  obj <- x[["seurat"]]
  ldat <- x[["ldat"]]
  for(j in 1:ncol(obj@meta.data)){
    if(is.factor(obj@meta.data[,j]) == T){
      obj@meta.data[,j][is.na(obj@meta.data[,j])] <- "N.A"
    }
    if(is.character(obj@meta.data[,j]) == T){
      obj@meta.data[,j][is.na(obj@meta.data[,j])] <- "N.A"
    }
  }
  x[["seurat"]] <- obj
  x[["ldat"]] <- ldat

  return(x)
}
)
```

3.3.2 Make loom file from Seurat/loom object

To facilitate the work, we optimized Seurat `Convert` function to merge a Seurat/Loom list to one Loom file, containing the var matrix with spliced, unspliced layers and obs with all embedding, tsne, umap, pca, clustering, etc. A Loom file issue from the function above `Convert.seurat_loom` will be saved in the current working folder.

```
library(loomR)
source("~/Desktop/velocyto/Script/Convert_Seurat_loom.R")

for (sample.name in sample.groupBy) {
  obj.sl <- obj[[sample.name]]
  pfile <- Convert.seurat_loom(from = obj.sl, to = "loom", filename =
     paste0(sample.name, "_zoom.loom") )
  pfile$close_all()
}
```

4 Session information

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.3 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_GB.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_GB.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
```

```
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] velocyto.R_0.6     Matrix_1.3-4       forcats_0.5.1      stringr_1
   .4.0
##  [5] purrr_0.3.4        readr_2.0.0        tidyr_1.1.3        tibble_3
   .1.3
##  [9] tidyverse_1.3.1    loomR_0.2.1.9000   hdf5r_1.3.3        R6_2.5.0
## [13] dplyr_1.0.7        ggplot2_3.3.5      SeuratObject_4.0.2 Seurat_4
   .0.3
##
## loaded via a namespace (and not attached):
##   [1] Rtsne_0.15            colorspace_2.0-2      deldir_0.2-10
##   [4] ellipsis_0.3.2        ggridges_0.5.3        fs_1.5.0
##   [7] rstudioapi_0.13       spatstat.data_2.1-0   leiden_0.3.9
##  [10] listenv_0.8.0         ggrepel_0.9.1         bit64_4.0.5
##  [13] lubridate_1.7.10      fansi_0.5.0           xml2_1.3.2
##  [16] codetools_0.2-18      splines_4.0.3         knitr_1.33
##  [19] polyclip_1.10-0       jsonlite_1.7.2        broom_0.7.9
##  [22] ica_1.0-2             dbplyr_2.1.1          cluster_2.1.0
##  [25] png_0.1-7             uwot_0.1.10.9000      shiny_1.6.0
##  [28] sctransform_0.3.2     spatstat.sparse_2.0-0 compiler_4.0.3
##  [31] httr_1.4.2            backports_1.2.1       assertthat_0.2.1
##  [34] fastmap_1.1.0         lazyeval_0.2.2        cli_3.0.1
##  [37] later_1.2.0           htmltools_0.5.1.1     tools_4.0.3
##  [40] igraph_1.2.6          gtable_0.3.0          glue_1.4.2
##  [43] RANN_2.6.1            reshape2_1.4.4        Rcpp_1.0.7
##  [46] Biobase_2.50.0        scattermore_0.7       cellranger_1.1.0
##  [49] vctrs_0.3.8           nlme_3.1-152          lmtest_0.9-38
##  [52] xfun_0.24             globals_0.14.0        rvest_1.0.1
##  [55] mime_0.11             miniUI_0.1.1.1        lifecycle_1.0.0
##  [58] irlba_2.3.3           goftest_1.2-2         future_1.21.0
##  [61] MASS_7.3-53           zoo_1.8-9             scales_1.1.1
##  [64] spatstat.core_2.3-0   pcaMethods_1.82.0     hms_1.1.0
##  [67] promises_1.2.0.1      spatstat.utils_2.2-0  parallel_4.0.3
##  [70] RColorBrewer_1.1-2    yaml_2.2.1            reticulate_1.20
##  [73] pbapply_1.4-3         gridExtra_2.3         rpart_4.1-15
##  [76] stringi_1.7.3         BiocGenerics_0.36.1   rlang_0.4.11
##  [79] pkgconfig_2.0.3       matrixStats_0.60.0    evaluate_0.14
##  [82] lattice_0.20-41       ROCR_1.0-11           tensor_1.5
##  [85] patchwork_1.1.1       htmlwidgets_1.5.3     cowplot_1.1.1
##  [88] bit_4.0.4             tidyselect_1.1.1      parallelly_1.27.0
##  [91] RcppAnnoy_0.0.19      plyr_1.8.6            magrittr_2.0.1
##  [94] generics_0.1.0        DBI_1.1.1             haven_2.4.3
##  [97] pillar_1.6.2          withr_2.4.2           mgcv_1.8-33
## [100] fitdistrplus_1.1-5    survival_3.2-7        abind_1.4-5
## [103] future.apply_1.7.0    modelr_0.1.8          crayon_1.4.1
## [106] KernSmooth_2.23-20    utf8_1.2.2            spatstat.geom_2.2-2
## [109] plotly_4.9.4.1        tzdb_0.1.2            rmarkdown_2.9
## [112] readxl_1.3.1          grid_4.0.3            data.table_1.14.0
## [115] reprex_2.0.0          digest_0.6.27         xtable_1.8-4
## [118] httpuv_1.6.1          munsell_0.5.0         viridisLite_0.4.0
```

References

1. La Manno G, Soldatov R, Zeisel A, Braun E, Hochgerner H, Petukhov V, Lidschreiber K, Kastriti ME, Lönnerberg P, Furlan A, Fan J, Borm LE, Liu Z, Bruggen D van, Guo J, He X, Barker R, Sundström E, Castelo-Branco G, Cramer P, Adameyko I, Linnarsson S, Kharchenko PV. RNA velocity of single cells. *Nature* 2018;