Pancreas example

Jean Fan

11/13/2020

Vignette Template

```
library(veloviz)

library(reticulate)
use_condaenv("r-velocity", required = TRUE)
scv <- import("scvelo")
adata <- scv$datasets$pancreas()

## spliced and unspliced expression matrices
spliced <- as.matrix(t(adata$layers['spliced']))
unspliced <- as.matrix(t(adata$layers['unspliced']))
cells <- adata$obs_names$values
genes <- adata$var_names$values
colnames(spliced) <- colnames(unspliced) <- cells
rownames(spliced) <- rownames(unspliced) <- genes
## extract clusters
clusters <- adata$obs$clusters
names(clusters) <- adata$obs_names$values
## old embedding
emb.original <- adata$obsm['X_umap'] #extract umap embedding
rownames(emb.original) <- names(clusters)
## plot
par(mfrow <- c(1,1))
plotEmbedding(emb.original, groups = clusters,
              xlab = "UMAP X", ylab = "UMAP Y",
              mark.clusters = TRUE)

## subsample to create smaller dataset
## that can be included with package
set.seed(0)
good.cells <- sample(rownames(emb.original), nrow(emb.original)/5)
spliced <- spliced[,good.cells]
unspliced <- unspliced[,good.cells]
clusters <- clusters[good.cells]
emb.original <- emb.original[good.cells,]
## plot
par(mfrow = c(1,1))
plotEmbedding(emb.original, groups = clusters,
              xlab = "UMAP X", ylab = "UMAP Y",
              mark.clusters=TRUE)
```

```r
## filter to well detected genes
vi <- rowSums(spliced) > 10 & rowSums(unspliced) > 10
spliced <- spliced[vi,]
unspliced <- unspliced[vi,]

## analyze
counts <- spliced + unspliced # use combined spliced and unspliced counts
cpm <- normalizeDepth(counts) # cpm normalize
matnorm <- normalizeVariance(cpm)
matnorm <- log10(matnorm + 1)
pcs <- reduceDimensions(matnorm, center = TRUE, scale = TRUE, nPCs = 50)

## velocity model
library(velocyto.R)
cell.dist <- as.dist(1-cor(t(pcs))) # cell distance in PC space
vel <- gene.relative.velocity.estimates(spliced,
                                        unspliced,
                                        kCells = 30,
                                        cell.dist = cell.dist,
                                        fit.quantile = 0.1)

## save
pancreas <- list(
  spliced = spliced,
  unspliced = unspliced,
  clusters = clusters,
  pcs = pcs,
  cell.dist = cell.dist,
  vel = vel
)
usethis::use_data(pancreas, overwrite=TRUE)
```

Compare visualizations

```r
## load data
library(veloviz)
data("pancreas")

par(mfrow=c(2,2), mar=rep(1,4))

## 2D embedding by PCA
emb.pcs = pancreas$pcs[,1:2]
plotEmbedding(emb.pcs, groups=pancreas$clusters, main='PCA')
```

```
## using provided groups as a factor
```

```r
## 2D embedding by tSNE
set.seed(0)
emb.tsne = Rtsne::Rtsne(pancreas$pcs[,1:10], perplexity=30)$Y
rownames(emb.tsne) <- rownames(pancreas$pcs)
plotEmbedding(emb.tsne, groups=pancreas$clusters, main='tSNE')
```

```
## using provided groups as a factor
```

```r
## 2D embedding by UMAP
set.seed(0)
```

```r
emb.umap = uwot::umap(pancreas$pcs[,1:10], min_dist = 0.5)
rownames(emb.umap) <- rownames(pancreas$pcs)
plotEmbedding(emb.umap, groups=pancreas$clusters, main='UMAP')
```

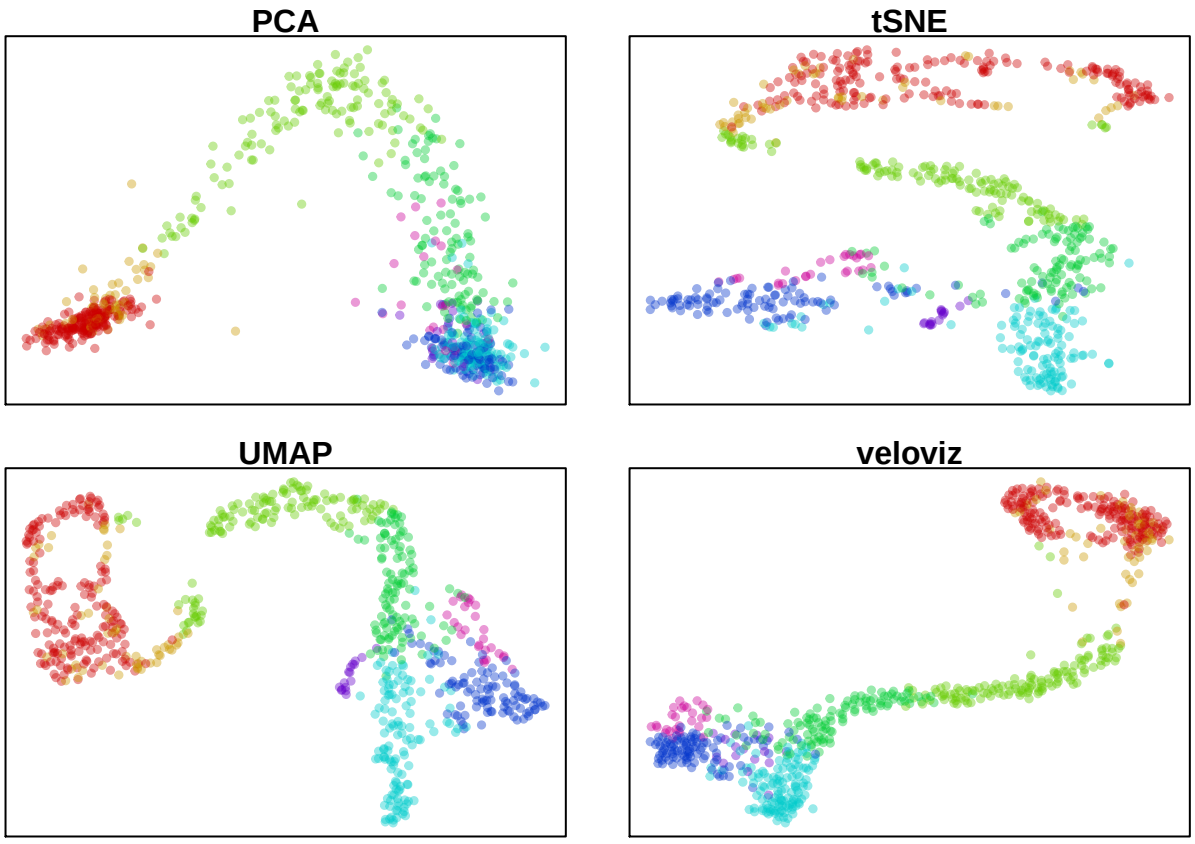
```
## using provided groups as a factor
```

```r
## 2D embedding by veloviz
vig = buildVeloviz(
  curr = pancreas$vel$curr,
  proj = pancreas$vel$proj,
  normalize.depth = FALSE,
  use.ods.genes = TRUE,
  alpha = 0.05,
  pca = TRUE,
  nPCs = 10,
  center = TRUE,
  scale = TRUE,
  k = 5,
  seed = 0,
  verbose = TRUE
)
```

```
## Warning in if (!class(curr) %in% c("dgCMatrix", "dgTMatrix")) {: the condition
## has length > 1 and only the first element will be used

## Converting to sparse matrix ...

## Warning in if (!class(proj) %in% c("dgCMatrix", "dgTMatrix")) {: the condition
## has length > 1 and only the first element will be used

## Converting to sparse matrix ...

## Identifying overdispersed genes...

## Using general additive modeling with k = 5...

## Identifed 1151 overdispersed genes using
##     adjusted p-value threshold alpha = 0.05

## Performing dimensionality reduction by PCA...

## Centering...

## Using unit variance...

## Projecting current cells onto PCs...

## Projecting future cells onto PCs...

## Generating velocity informed embedding...

## [1] "Done finding neighbors"
## [1] "Done making graph"
```

```r
emb.veloviz = vig$fdg_coords
plotEmbedding(emb.veloviz, groups=pancreas$clusters, main='veloviz')
```

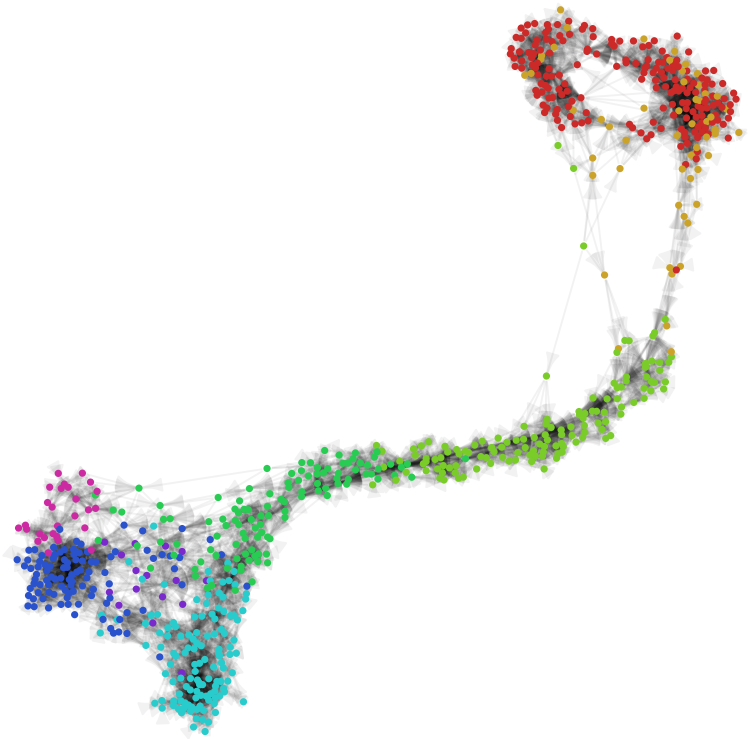
```
## using provided groups as a factor
```




```
par(mfrow=c(1,1), mar=rep(1,4))
g = plotVeloviz(vig, clusters=pancreas$clusters, seed=0, verbose=TRUE)
```

```
## Warning in if (!is.na(clusters) & is.na(col)) {: the condition has length > 1
## and only the first element will be used
```

```
## Using provided clusters...
```



Now we remove cells

```
## remove EP cells along original trajectory
x = emb.original[,1]
vi = x > -5 & x < 0
good.cells = rownames(emb.original)[!vi]
plotEmbedding(emb.original[good.cells,], groups=clusters,
              xlab = "UMAP X", ylab = "UMAP Y", mark.clusters=TRUE)
spliced = spliced[,good.cells]
unspliced = unspliced[,good.cells]
clusters = clusters[good.cells]

## analyze
counts <- spliced + unspliced # use combined spliced and unspliced counts
cpm <- normalizeDepth(counts) # cpm normalize
matnorm <- normalizeVariance(cpm)
matnorm <- log10(matnorm + 1)
pcs <- reduceDimensions(matnorm, center = TRUE, scale = TRUE, nPCs = 50)

## velocity model
library(velocyto.R)
cell.dist <- as.dist(1-cor(t(pcs))) # cell distance in PC space
vel <- gene.relative.velocity.estimates(spliced,
                                        unspliced,
                                        kCells = 30,
                                        cell.dist = cell.dist,
                                        fit.quantile = 0.1)

pancreasWithGap <- list(
  spliced = spliced,
```

```
  unspliced = unspliced,
  clusters = clusters,
  pcs = pcs,
  cell.dist = cell.dist,
  vel = vel
)
usethis::use_data(pancreasWithGap, overwrite=TRUE)
```

Compare

```
## load data
library(veloviz)
data("pancreasWithGap")

par(mfrow=c(2,2), mar=rep(1,4))

## 2D embedding by PCA
emb.pcs = pancreasWithGap$pcs[,1:2]
plotEmbedding(emb.pcs, groups=pancreasWithGap$clusters, main='PCA')
```

```
## using provided groups as a factor
```

```
## 2D embedding by tSNE
set.seed(0)
emb.tsne = Rtsne::Rtsne(pancreasWithGap$pcs[,1:10], perplexity=30)$Y
rownames(emb.tsne) <- rownames(pancreasWithGap$pcs)
plotEmbedding(emb.tsne, groups=pancreasWithGap$clusters, main='tSNE')
```

```
## using provided groups as a factor
```

```
## 2D embedding by UMAP
set.seed(0)
emb.umap = uwot::umap(pancreasWithGap$pcs[,1:10], min_dist = 0.5)
rownames(emb.umap) <- rownames(pancreasWithGap$pcs)
plotEmbedding(emb.umap, groups=pancreasWithGap$clusters, main='UMAP')
```

```
## using provided groups as a factor
```

```
## 2D embedding by veloviz
vig = buildVeloviz(
  curr = pancreasWithGap$vel$curr,
  proj = pancreasWithGap$vel$proj,
  normalize.depth = FALSE,
  use.ods.genes = TRUE,
  alpha = 0.05,
  pca = TRUE,
  nPCs = 10,
  center = TRUE,
  scale = TRUE,
  k = 5,
  seed = 0,
  verbose = FALSE
)
```

```
## Warning in if (!class(curr) %in% c("dgCMatrix", "dgTMatrix")) {: the condition
## has length > 1 and only the first element will be used
```

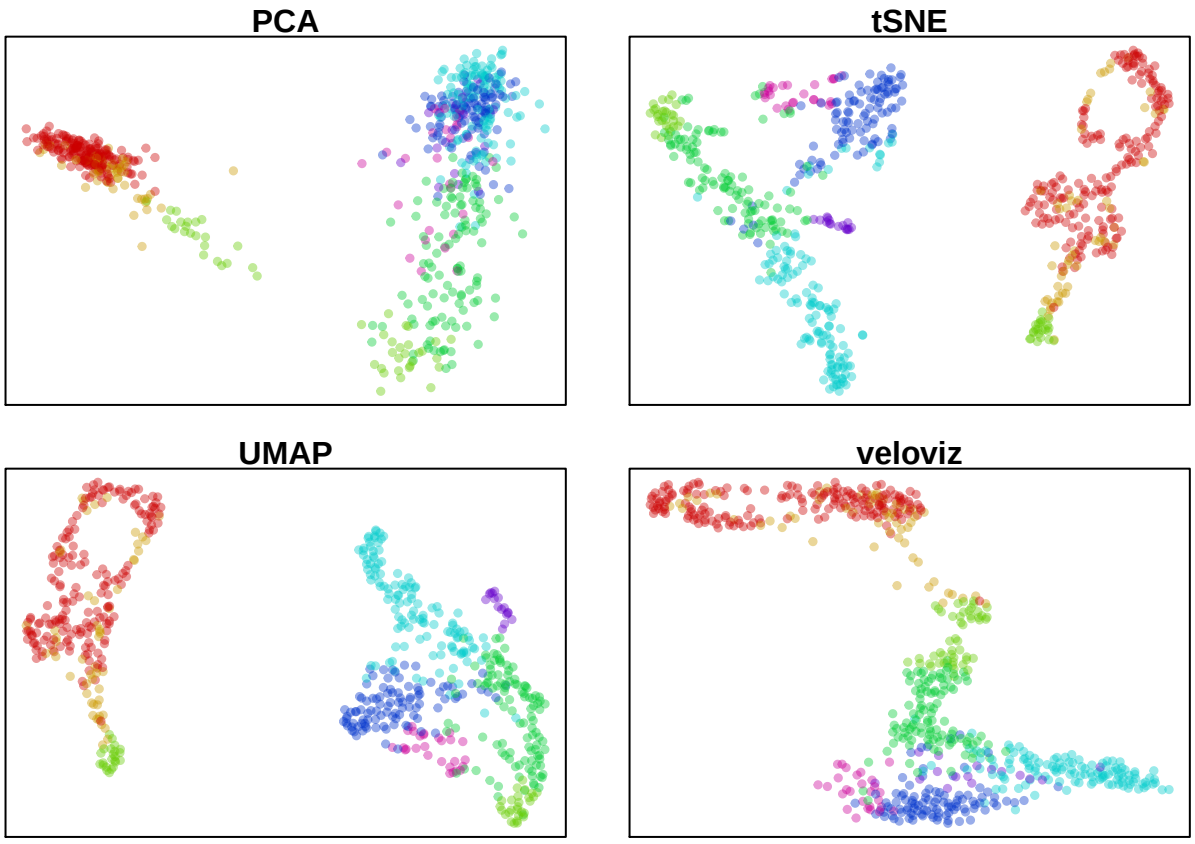
```
## Warning in if (!class(proj) %in% c("dgCMatrix", "dgTMatrix")) {: the condition
```

```
## has length > 1 and only the first element will be used
```

```
## [1] "Done finding neighbors"
## [1] "Done making graph"
```

```
emb.veloviz = vig$fdg_coords
plotEmbedding(emb.veloviz, groups=pancreasWithGap$clusters, main='veloviz')
```

```
## using provided groups as a factor
```




```
par(mfrow=c(1,1), mar=rep(1,4))
g = plotVeloviz(vig, clusters = pancreasWithGap$clusters, seed = 0)
```

```
## Warning in if (!is.na(clusters) & is.na(col)) {: the condition has length > 1
## and only the first element will be used
```

```
## Using provided clusters...
```