MERFISH example

Jean Fan

11/17/2020

MERFISH

```
dir <- 'path/to/downloaded/data'
cell_gexp <- as.matrix(read.csv(paste0(dir, 'S12_cell_gexp.csv.gz'),
                                row.names=1))
print(cell_gexp[1:5,1:5])
nuc_gexp <- as.matrix(read.csv(paste0(dir, 'S14_nuc_gexp.csv.gz'),
                               row.names=1))
print(nuc_gexp[1:5,1:5])
cyto_gexp <- cell_gexp - nuc_gexp

gene_info <- read.csv(paste0(dir, 'S1_codebook.csv.gz'),
                      header=FALSE, stringsAsFactors = FALSE)
long.genes <- gene_info[2:9051,1]
short.genes <- gene_info[9280:10279,1]
bad.genes <- gene_info[,1][grepl('Blank', gene_info[,1])]

## use subset of genes
test.genes <- long.genes
cell_gexp <- cell_gexp[test.genes,]
nuc_gexp <- nuc_gexp[test.genes,]
cyto_gexp <- cyto_gexp[test.genes,]
cd <- cell_gexp

## annotate batch
batch <- sapply(colnames(cd), function(x) strsplit(x, '_')[[1]][1])
batch <- factor(batch)
table(batch)

## limit to one batch in example
subcells <- names(batch)[batch=='B1']
spliced = cyto_gexp[, subcells]
unspliced = nuc_gexp[, subcells]
vi <- rowSums(spliced) > 10 & rowSums(unspliced) > 10
spliced = spliced[vi,]
unspliced = unspliced[vi,]

library(veloviz)
## normalize
all.counts = spliced + unspliced # use cell count
all.cpm = normalizeDepth(all.counts) # cpm normalize
pcs = reduceDimensions(all.cpm,
```

```
                        nPCs = 30,
                        center=TRUE, scale=FALSE,
                        use.ods.genes = TRUE,
                        max.ods.genes = 1000,
                        alpha = 0.05,
                        plot=TRUE)

## use first 2 PCs as embedding
emb <- pcs[,1:2]
rownames(emb) <- colnames(all.cpm)

## use angle as color
angle <- atan2(emb[,2], emb[,1])
obs <- emb[order(angle), ]
col = colorRampPalette(c(rainbow(10)))(nrow(obs))
names(col) = rownames(obs)
## double check
plot(emb, col=col[rownames(emb)], pch=16)

## 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
MERFISH <- list(
  nuc = Matrix::Matrix(unspliced, sparse=TRUE),
  cyto = Matrix::Matrix(spliced, sparse=TRUE),
  col = col,
  pcs = pcs,
  cell.dist = cell.dist,
  vel = vel
)
usethis::use_data(MERFISH, overwrite = TRUE)
```

```
library(veloviz)
data(MERFISH)

par(mfrow=c(2,2), mar=rep(1,4))
## 2D embedding by PCA
emb.pcs = MERFISH$pcs[,1:2]
plotEmbedding(emb.pcs, col=MERFISH$col, main='PCA')
```

```
## using supplied colors as is
```

```
emb.pcs = MERFISH$pcs[,2:3]
plotEmbedding(emb.pcs, col=MERFISH$col, main='PCA')
```

```
## using supplied colors as is
```

```
emb.pcs = MERFISH$pcs[,3:4]
plotEmbedding(emb.pcs, col=MERFISH$col, main='PCA')
```

```
## using supplied colors as is
```

```
emb.pcs = MERFISH$pcs[,4:5]
plotEmbedding(emb.pcs, col=MERFISH$col, main='PCA')
```

```
## using supplied colors as is
```


```
par(mfrow=c(2,2), mar=rep(1,4))
## 2D embedding by tSNE
set.seed(0)
emb.tsne = Rtsne::Rtsne(MERFISH$pcs[,1:5], perplexity = 10)$Y
rownames(emb.tsne) <- rownames(MERFISH$pcs)
plotEmbedding(emb.tsne, col=MERFISH$col, main='tSNE')
```

```
## using supplied colors as is
```

```
set.seed(0)
emb.tsne = Rtsne::Rtsne(MERFISH$pcs[,1:5], perplexity = 30)$Y
rownames(emb.tsne) <- rownames(MERFISH$pcs)
plotEmbedding(emb.tsne, col=MERFISH$col, main='tSNE')
```

```
## using supplied colors as is
```

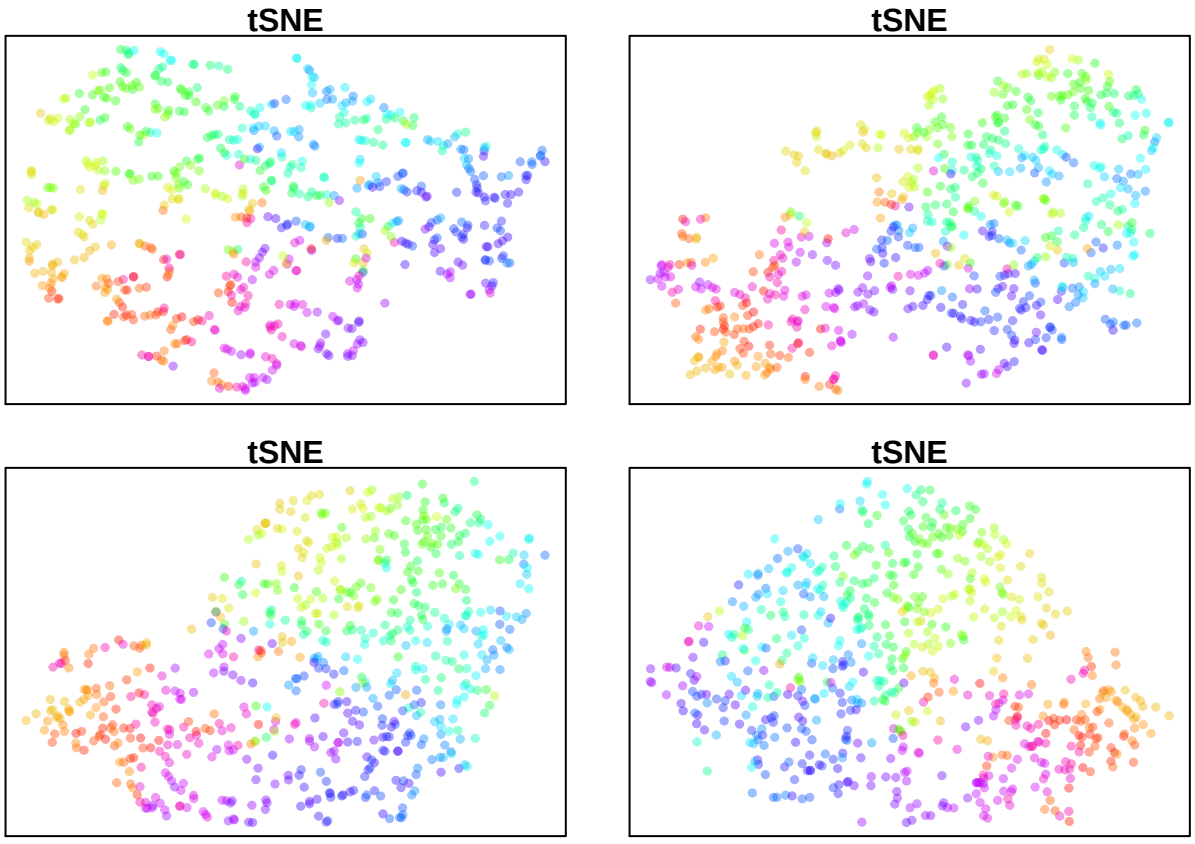
```
set.seed(0)
emb.tsne = Rtsne::Rtsne(MERFISH$pcs[,1:5], perplexity = 50)$Y
rownames(emb.tsne) <- rownames(MERFISH$pcs)
```

```
plotEmbedding(emb.tsne, col=MERFISH$col, main='tSNE')
```

```
## using supplied colors as is
```

```
set.seed(0)
emb.tsne = Rtsne::Rtsne(MERFISH$pcs[,1:5], perplexity = 100)$Y
rownames(emb.tsne) <- rownames(MERFISH$pcs)
plotEmbedding(emb.tsne, col=MERFISH$col, main='tSNE')
```

```
## using supplied colors as is
```




```
par(mfrow=c(2,2), mar=rep(1,4))
## 2D embedding by UMAP
set.seed(0)
emb.umap = uwot::umap(MERFISH$pcs[,1:5], min_dist = 0.1)
rownames(emb.umap) <- rownames(MERFISH$pcs)
plotEmbedding(emb.umap, col=MERFISH$col, main='UMAP')
```

```
## using supplied colors as is
```

```
set.seed(0)
emb.umap = uwot::umap(MERFISH$pcs[,1:5], min_dist = 0.3)
rownames(emb.umap) <- rownames(MERFISH$pcs)
plotEmbedding(emb.umap, col=MERFISH$col, main='UMAP')
```

```
## using supplied colors as is
```

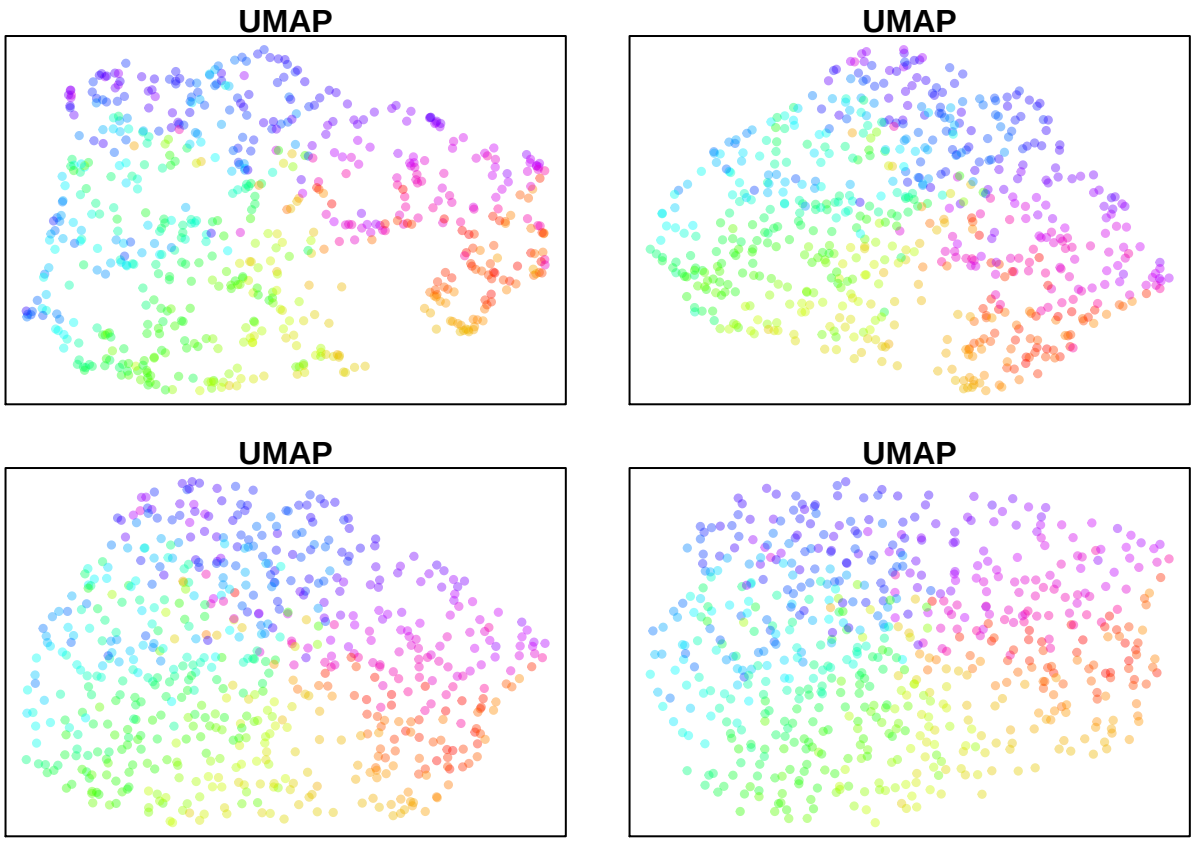
```
set.seed(0)
emb.umap = uwot::umap(MERFISH$pcs[,1:5], min_dist = 0.5)
```

```
rownames(emb.umap) <- rownames(MERFISH$pcs)
plotEmbedding(emb.umap, col=MERFISH$col, main='UMAP')
```

```
## using supplied colors as is
```

```
set.seed(0)
emb.umap = uwot::umap(MERFISH$pcs[,1:5], min_dist = 1)
rownames(emb.umap) <- rownames(MERFISH$pcs)
plotEmbedding(emb.umap, col=MERFISH$col, main='UMAP')
```

```
## using supplied colors as is
```




```
par(mfrow=c(2,2), mar=rep(1,4))
## 2D embedding by velovizr
g <- buildVeloviz(MERFISH$vel$current,
                  MERFISH$vel$projected,
                  k = 10,
                  nPCs = 5,
                  center=TRUE, scale=FALSE,
                  use.ods.genes = TRUE,
                  max.ods.genes = 1000,
                  alpha = 0.05,
                  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"
```

```
plotEmbedding(g$fdg_coords, col = MERFISH$col[rownames(g$fdg_coords)], main='veloviz')
```

```
## using supplied colors as is
```

```
g <- buildVeloviz(MERFISH$vel$current,
                  MERFISH$vel$projected,
                  k = 30,
                  nPCs = 5,
                  center=TRUE, scale=FALSE,
                  use.ods.genes = TRUE,
                  max.ods.genes = 1000,
                  alpha = 0.05,
                  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(curr) %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"
```

```
plotEmbedding(g$fdg_coords, col = MERFISH$col[rownames(g$fdg_coords)], main='veloviz')
```

```
## using supplied colors as is
```

```
g <- buildVeloviz(MERFISH$vel$current,
                  MERFISH$vel$projected,
                  k = 50,
                  nPCs = 5,
                  center=TRUE, scale=FALSE,
                  use.ods.genes = TRUE,
                  max.ods.genes = 1000,
                  alpha = 0.05,
                  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(curr) %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"
```

```
plotEmbedding(g$fdg_coords, col = MERFISH$col[rownames(g$fdg_coords)], main='veloviz')
```

```
## using supplied colors as is
```

```
g <- buildVeloviz(MERFISH$vel$current,
                  MERFISH$vel$projected,
                  k = 100,
                  nPCs = 5,
                  center=TRUE, scale=FALSE,
                  use.ods.genes = TRUE,
```

```
                  max.ods.genes = 1000,
                  alpha = 0.05,
                  verbose = FALSE)
```

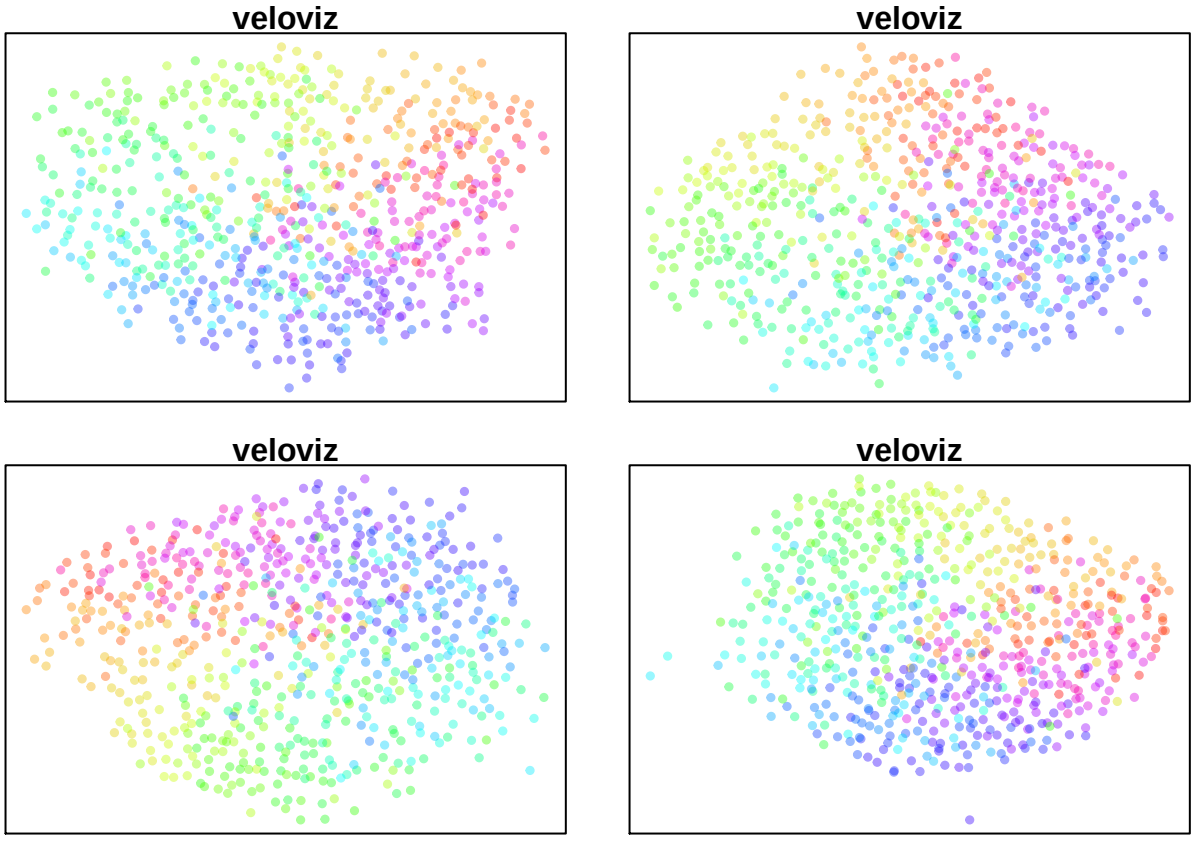
```
## Warning in if (!class(curr) %in% c("dgCMatrix", "dgTMatrix")) {: the condition
## has length > 1 and only the first element will be used

## Warning in if (!class(curr) %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"
```

```
plotEmbedding(g$fdg_coords, col = MERFISH$col[rownames(g$fdg_coords)], main='veloviz')
```

```
## using supplied colors as is
```




```
library(velocyto.R)
show.velocity.on.embedding.cor(scale(emb.pcs), vel,
                               n = 100, show.grid.flow = TRUE, grid.n = 20,
                               cell.colors = col, arrow.scale = 1)
show.velocity.on.embedding.cor(scale(emb.tsne), vel,
                               n = 100, show.grid.flow = TRUE, grid.n = 20,
                               cell.colors = col, arrow.scale = 1)
show.velocity.on.embedding.cor(scale(emb.umap), vel,
                               n = 100, show.grid.flow = TRUE, grid.n = 20,
                               cell.colors = col, arrow.scale = 1)
show.velocity.on.embedding.cor(scale(g$fdg_coords), vel,
                               n = 100, show.grid.flow = TRUE, grid.n = 20,
```

```
                                cell.colors = col, arrow.scale = 1)
```