Visualizing MERFISH data using VeloViz

In this example, we will use VeloViz to create a 2D embedding to visualize MERFISH data collected from U-2 OS cells in culture. Since this data comes from a cell line in culture, we expect the main temporal signal to be progression throught the cell-cycle. We will compare the VeloViz embedding to other commonly used embeddings. We will also compare the results we get when we restrict the input genes.

Load preprocessed data

MERFISH data from Xia et. al., *PNAS*, 2019. This data is provided with the VeloViz package.

```
col = MERFISH$col
pcs = MERFISH$pcs
vel = MERFISH$vel

curr = vel$current
proj = vel$projected
```

Load cell cycle genes

We will compare the VeloViz embeddings constructed with all genes to embeddings created using cell-cycle genes in the GO mitotic cell-cycle gene set and to embeddings created using genes that were found to have cell-cycle dependent expression in Xia et. al., *PNAS*, 2019.

```
merfish.genes = rownames(curr)

#GO cell cycle genes (GO:0000278)
# https://www.gsea-msigdb.org/gsea/msigdb/cards/GO_MITOTIC_CELL_CYCLE
cycle.genes.go = read.csv("GO_0000278.csv",header = FALSE)
cycle.genes.go = cycle.genes.go$V1

merfish.cycle.go =  merfish.genes[which(merfish.genes %in% cycle.genes.go)]
curr.go = curr[merfish.cycle.go,]
proj.go = proj[merfish.cycle.go,]

#MERFISH genes exhibiting cell-cycle-dependent expression (Xia et al 2019, Supp Dataset 8)
# https://www.pnas.org/content/116/39/19490
cycle.genes.pnas = read.csv("pnas_sd08.csv",header = TRUE)
cycle.genes.pnas = cycle.genes.pnas$Gene

merfish.cycle.pnas = merfish.genes[which(merfish.genes %in% cycle.genes.pnas)]
curr.pnas = curr[merfish.cycle.pnas,]
proj.pnas = proj[merfish.cycle.pnas,]
```

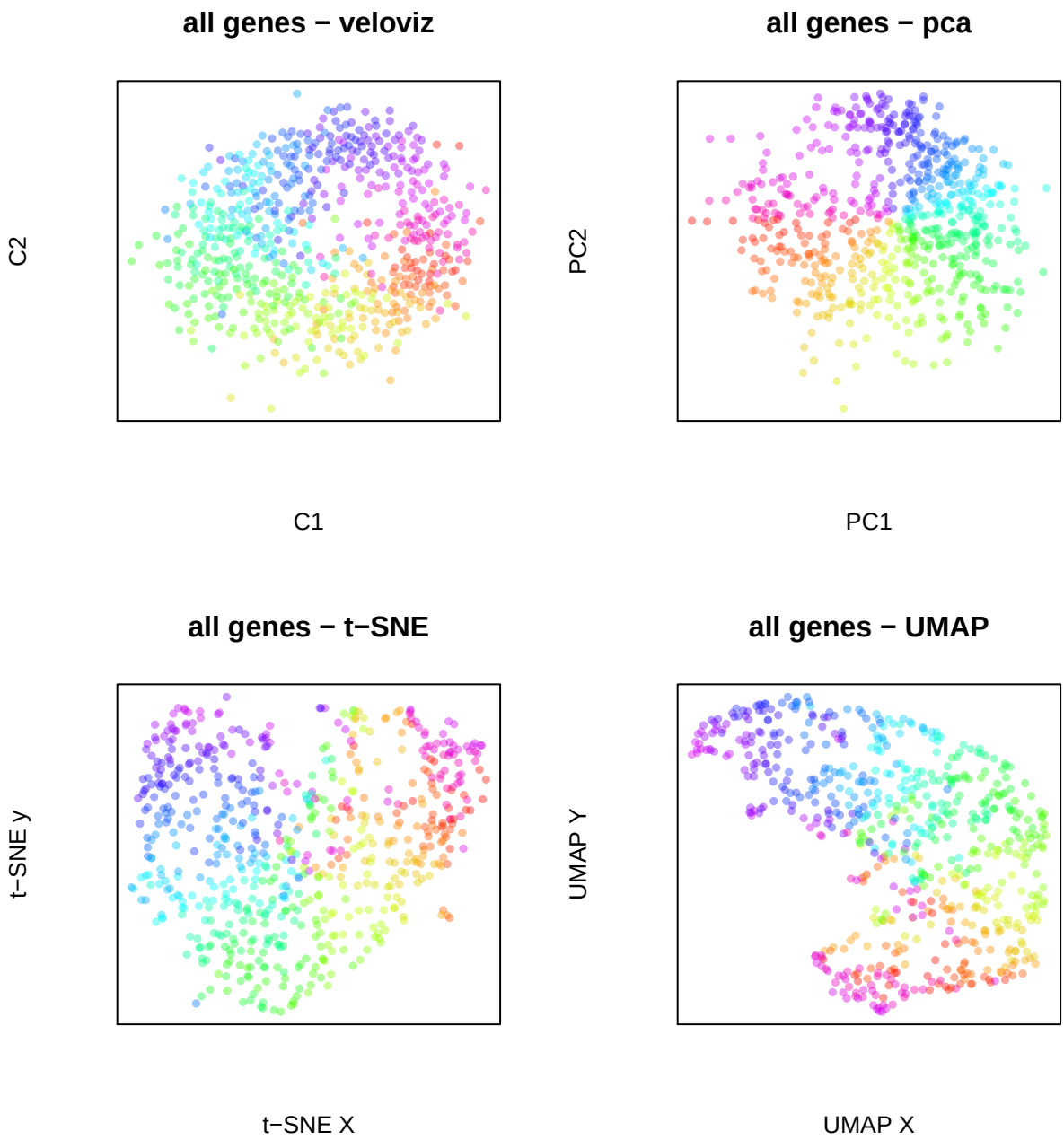
Build VeloViz embedding using all genes

```
par(mfrow = c(2,2))

veloviz.all = buildVeloviz(
  curr = curr,
  proj = proj,
  normalize.depth = TRUE,
  use.ods.genes = FALSE,
  pca = TRUE,
  nPCs = 3,
  center = TRUE,
  scale = TRUE,
  k = 100,
  similarity.threshold = 0,
  distance.weight = 1,
  distance.threshold = 1,
  weighted = TRUE,
  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
```

```
emb.all.vv = veloviz.all$fdg_coords
plotEmbedding(emb.all.vv, colors = col[rownames(emb.all.vv)], main = 'all genes - veloviz')

#PCA
emb.all.pca = pcs[,1:2]
plotEmbedding(emb.all.pca, colors = col, main = 'all genes - pca')

#tSNE
set.seed(0)
emb.all.tsne = Rtsne::Rtsne(pcs[,1:5], perplexity = 100)$Y
rownames(emb.all.tsne) = rownames(pcs)
plotEmbedding(emb.all.tsne, colors = col, main = 'all genes - t-SNE',
              xlab = "t-SNE X", ylab = "t-SNE y")

#UMAP
set.seed(0)
emb.all.umap = umap::umap(pcs[,1:5], min_dist = 0.3)$layout
rownames(emb.all.umap) = rownames(pcs)
plotEmbedding(emb.all.umap, colors = col, main = 'all genes - UMAP',
              xlab = "UMAP X", ylab = "UMAP Y")
```


all genes − veloviz
C2
C1
all genes − pca
PC2
PC1
all genes − t−SNE
t−SNE y
t−SNE X
all genes − UMAP
UMAP Y
UMAP X

Build VeloViz embedding using GO cell cycle genes

First, reduce dimensions.

```
curr.go.norm = normalizeDepth(curr.go)
```

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

```
## Converting to sparse matrix ...
```

```
## Normalizing matrix with 645 cells and 360 genes
```

```
curr.go.norm = normalizeVariance(curr.go.norm, details = TRUE)
```

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

```
## Identifed 78 overdispersed genes using
##     adjusted p-value threshold alpha = 0.05
```

```
curr.go.norm = log10(curr.go.norm$matnorm + 1)
curr.go.pca = RSpectra::svds(A = t(as.matrix(curr.go.norm)), k = 50,
                             opts = list(center = TRUE, scale = FALSE,
                                         maxitr = 2000, tol = 1e-10))
curr.go.pca = curr.go.pca$u
rownames(curr.go.pca) = rownames(pcs)
```

Now, build embeddings.

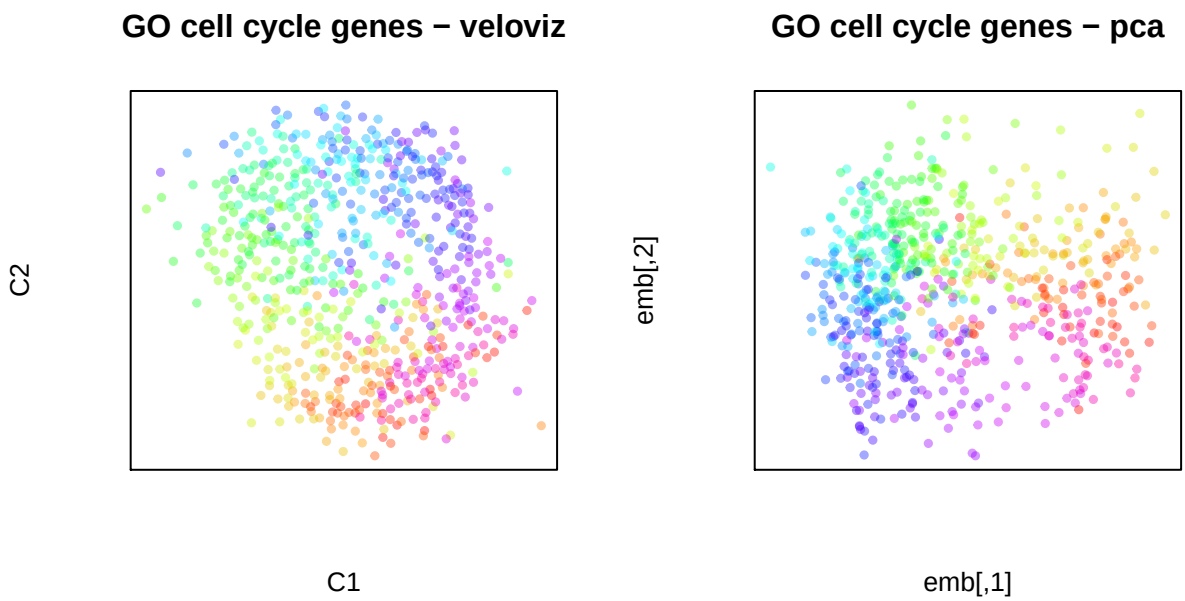
```
par(mfrow = c(2,2))

veloviz.go = buildVeloviz(
  curr = curr.go,
  proj = proj.go,
  normalize.depth = TRUE,
  use.ods.genes = FALSE,
  pca = TRUE,
  nPCs = 3,
  center = TRUE,
  scale = TRUE,
  k = 20,
  similarity.threshold = 0,
  distance.weight = 0.1,
  distance.threshold = 1,
  weighted = TRUE,
  seed = 0,
  verbose = FALSE
)
```

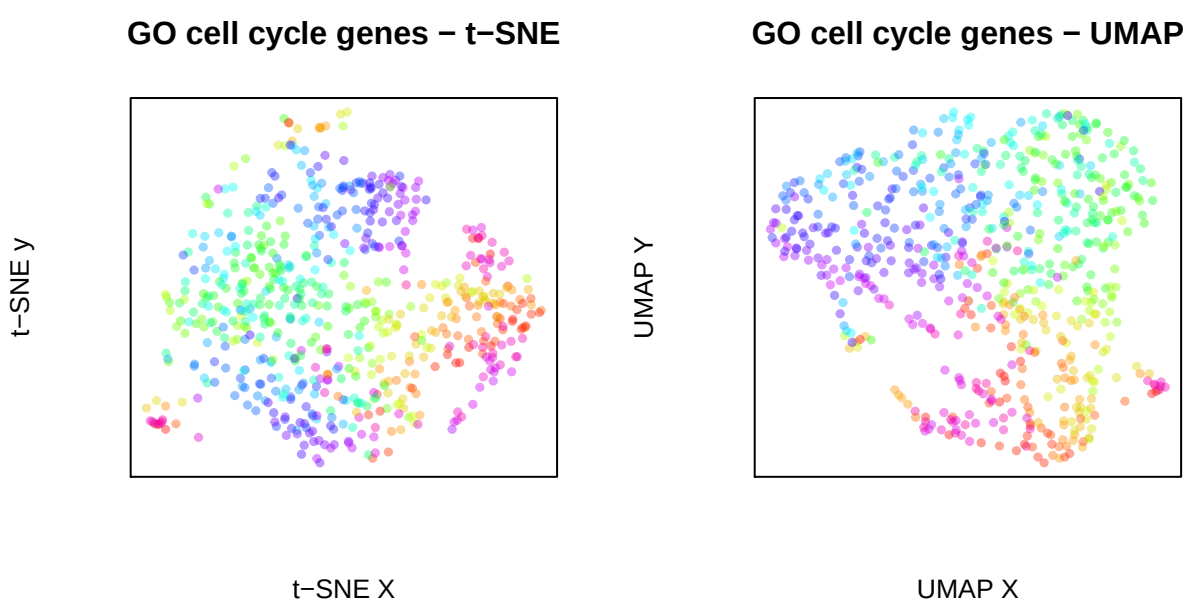
```
## Warning in if (!class(curr) %in% c("dgCMatrix", "dgTMatrix")) {: the condition
## has length > 1 and only the first element will be used
```

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

```
emb.go.vv = veloviz.go$fdg_coords
plotEmbedding(emb.go.vv, colors = col[rownames(emb.go.vv)],
              main = 'GO cell cycle genes - veloviz')

#PCA
```

```r
emb.go.pca = curr.go.pca[,1:2]
plotEmbedding(emb.go.pca, colors = col, main = 'GO cell cycle genes - pca')

#tSNE
set.seed(0)
emb.go.tsne = Rtsne::Rtsne(curr.go.pca[,1:5], perplexity = 100)$Y
rownames(emb.go.tsne) = rownames(curr.go.pca)
plotEmbedding(emb.go.tsne, colors = col, main = 'GO cell cycle genes - t-SNE',
              xlab = "t-SNE X", ylab = "t-SNE y")

#UMAP
set.seed(0)
emb.go.umap = umap::umap(curr.go.pca[,1:5], min_dist = 0.3)$layout
rownames(emb.go.umap) = rownames(curr.go.pca)
plotEmbedding(emb.go.umap, colors = col, main = 'GO cell cycle genes - UMAP',
              xlab = "UMAP X", ylab = "UMAP Y")
```


GO cell cycle genes − veloviz
C2
C1
GO cell cycle genes − pca
emb[,2]
emb[,1]


GO cell cycle genes − t−SNE
t−SNE y
t−SNE X
GO cell cycle genes − UMAP
UMAP Y
UMAP X

Build VeloViz embedding with cell-cycle dependent genes

First, reduce dimensions.

```
curr.pnas.norm = normalizeDepth(curr.pnas)
```

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

```
## Converting to sparse matrix ...
```

```
## Normalizing matrix with 645 cells and 1471 genes
```

```
curr.pnas.norm = normalizeVariance(curr.pnas.norm, details = TRUE)
```

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

```
## Identifed 389 overdispersed genes using
##     adjusted p-value threshold alpha = 0.05
```

```
curr.pnas.norm = log10(curr.pnas.norm$matnorm + 1)
curr.pnas.pca = RSpectra::svds(A = t(as.matrix(curr.pnas.norm)), k = 50,
                               opts = list(center = TRUE, scale = FALSE,
                                           maxitr = 2000, tol = 1e-10))
curr.pnas.pca = curr.pnas.pca$u
rownames(curr.pnas.pca) = rownames(pcs)
```

Now, build embeddings.

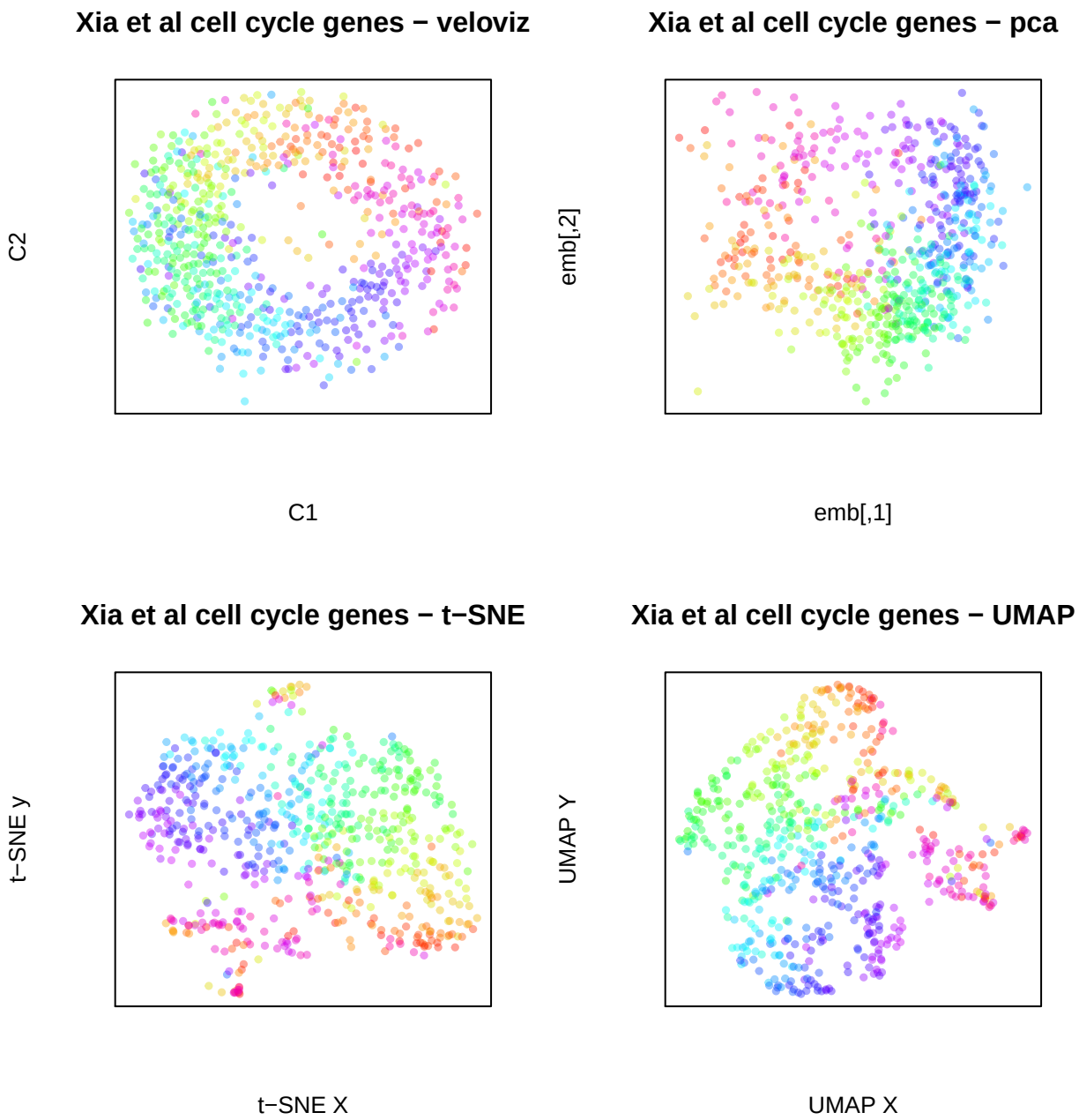
```
par(mfrow = c(2,2))

veloviz.pnas = buildVeloviz(
  curr = curr.pnas,
  proj = proj.pnas,
  normalize.depth = TRUE,
  use.ods.genes = FALSE,
  pca = TRUE,
  nPCs = 3,
  center = TRUE,
  scale = TRUE,
  k = 50,
  similarity.threshold = 0.5,
  distance.weight = 0.01,
  distance.threshold = 1,
  weighted = TRUE,
  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
```

```
emb.pnas.vv = veloviz.pnas$fdg_coords
plotEmbedding(emb.pnas.vv, colors = col[rownames(emb.pnas.vv)],
              main = 'Xia et al cell cycle genes - veloviz')

#PCA
```

```r
emb.pnas.pca = curr.pnas.pca[,1:2]
plotEmbedding(emb.pnas.pca, colors = col, main = 'Xia et al cell cycle genes - pca')

#tSNE
set.seed(0)
emb.pnas.tsne = Rtsne::Rtsne(curr.pnas.pca[,1:5], perplexity = 100)$Y
rownames(emb.pnas.tsne) = rownames(curr.pnas.pca)
plotEmbedding(emb.pnas.tsne, colors = col, main = 'Xia et al cell cycle genes - t-SNE',
              xlab = "t-SNE X", ylab = "t-SNE y")

#UMAP
set.seed(0)
emb.pnas.umap = umap::umap(curr.pnas.pca[,1:5], min_dist = 0.3)$layout
rownames(emb.pnas.umap) = rownames(curr.pnas.pca)
plotEmbedding(emb.pnas.umap, colors = col, main = 'Xia et al cell cycle genes - UMAP',
              xlab = "UMAP X", ylab = "UMAP Y")
```


Xia et al cell cycle genes − veloviz
C2
C1
Xia et al cell cycle genes − pca
emb[,2]
emb[,1]
Xia et al cell cycle genes − t−SNE
t−SNE y
t−SNE X
Xia et al cell cycle genes − UMAP
UMAP Y
UMAP X

Comparing VeloViz embeddings

```
par(mfrow = c(1,3))

plotEmbedding(emb.all.vv, colors = col[rownames(emb.pnas.vv)],
              main = 'all genes - veloviz')

plotEmbedding(emb.go.vv, colors = col[rownames(emb.pnas.vv)],
              main = 'GO cell cycle genes - veloviz')

plotEmbedding(emb.pnas.vv, colors = col[rownames(emb.pnas.vv)],
              main = 'Xia et al cell cycle genes - veloviz')
```