DropDAE

Contents

```
library(DropDAE)
library(Seurat)
library(splatter)
library(scater)
library(ggpubr)
library(torch)
set.seed(1234)
```

```
# Parameters for data simulation
ngroups <- 6                # Number of groups
batchCells <- 6000          # Number of cells in each batch
nGenes <- 500               # Number of genes
de.prob <- 0.1              # Proportion of differentially expressed (DE) genes
de.facLoc <- 0.2            # Location factor for DE effect
de.downProb <- 0.5          # Proportion of downregulated DE genes
de.facScale <- 0.4          # Scale factor for DE effect
dropout.type <- "experiment" # Type of dropout
dropout.mid <- 3            # Midpoint for dropout function
dropout.shape <- -1         # Shape parameter for dropout
seed <- 1
```

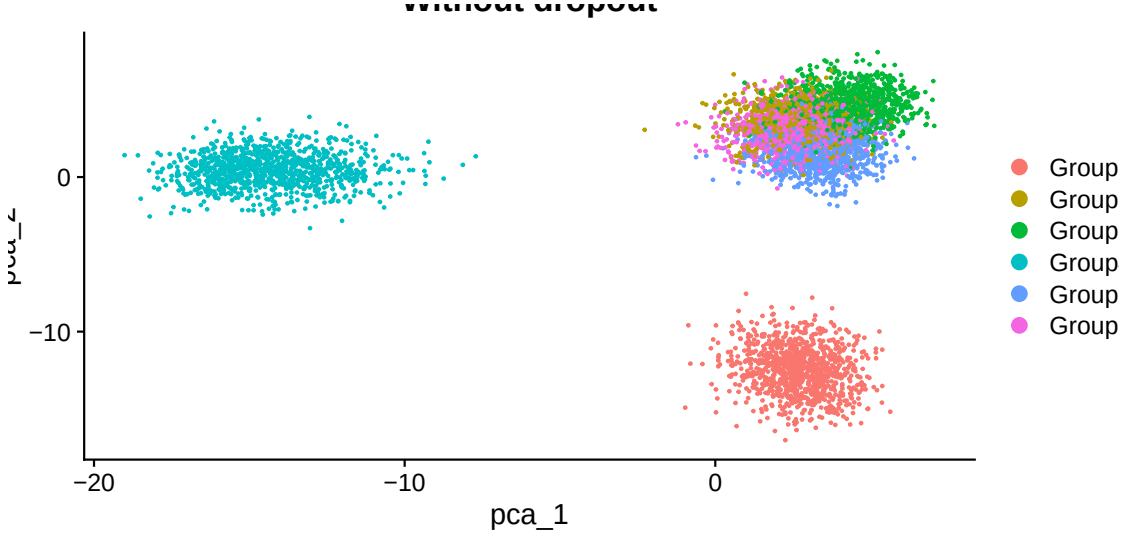
```
# Simulated data without dropouts (sim1)
# --------------------------------------
params.groups <- newSplatParams(batchCells = batchCells, nGenes = nGenes, seed = 1)

# Simulate groups with Splatter
sim1 <- splatSimulateGroups(params.groups,
                            group.prob = rep(1, ngroups) / ngroups, # Equal group proportions
                            de.prob = de.prob,
                            de.facLoc = de.facLoc,
                            de.facScale = de.facScale,
                            de.downProb = de.downProb,
                            verbose = FALSE,
                            seed = 1)

# Normalize counts and run PCA
sim1 <- logNormCounts(sim1)
sim1 <- runPCA(sim1)

# Add placeholder dropout to sim1
params <- newSplatParams(batchCells = batchCells, nGenes = nGenes, seed = 1)
params <- setParams(params, update = list(dropout.type = "experiment", dropout.mid = -99999, seed = 1))
sim1 <- splatter:::splatSimDropout(sim1, params)
```

```r
# Convert SingleCellExperiment object to Seurat object
sim1.s <- as.Seurat(sim1)
sim1.s$seurat_clusters <- as.numeric(as.factor(sim1$Group))
Idents(sim1.s) <- as.numeric(as.factor(sim1$Group))

# Process the Seurat object (Normalization, PCA, and UMAP)
sim1.s <- SCTransform(sim1.s, assay = "originalexp")
sim1.s <- RunPCA(sim1.s, features = VariableFeatures(sim1.s), verbose = FALSE)
sim1.s <- RunUMAP(sim1.s, dims = 1:10, verbose = FALSE)

# Simulated data with dropouts (sim2)
# -----------------------------------
params <- newSplatParams(seed = 1)
params <- setParams(params, nGenes = nGenes, update = list(dropout.type = "experiment",
                                                           dropout.mid = dropout.mid,
                                                           dropout.shape = dropout.shape,
                                                           seed = 1))

# Add experimental dropouts to sim1
sim2 <- splatter:::splatSimDropout(sim1, params)
sim2 <- logNormCounts(sim2)
sim2 <- runPCA(sim2)

# Convert SingleCellExperiment object to Seurat object
sim2.s <- as.Seurat(sim2)
sim2.s$seurat_clusters <- as.numeric(as.factor(sim2$Group))
Idents(sim2.s) <- as.numeric(as.factor(sim2$Group))

# Process the Seurat object (Normalization, PCA, and UMAP)
sim2.s <- SCTransform(sim2.s, assay = "originalexp")
sim2.s <- RunPCA(sim2.s, features = VariableFeatures(sim2.s), verbose = FALSE)
sim2.s <- RunUMAP(sim2.s, dims = 1:10, verbose = FALSE)

# Identify top 100 highly variable genes
hv_genes <- VariableFeatures(sim2.s)[1:100]

# Retrieve raw data matrix
count <- GetAssayData(sim2.s, slot = "count")
```

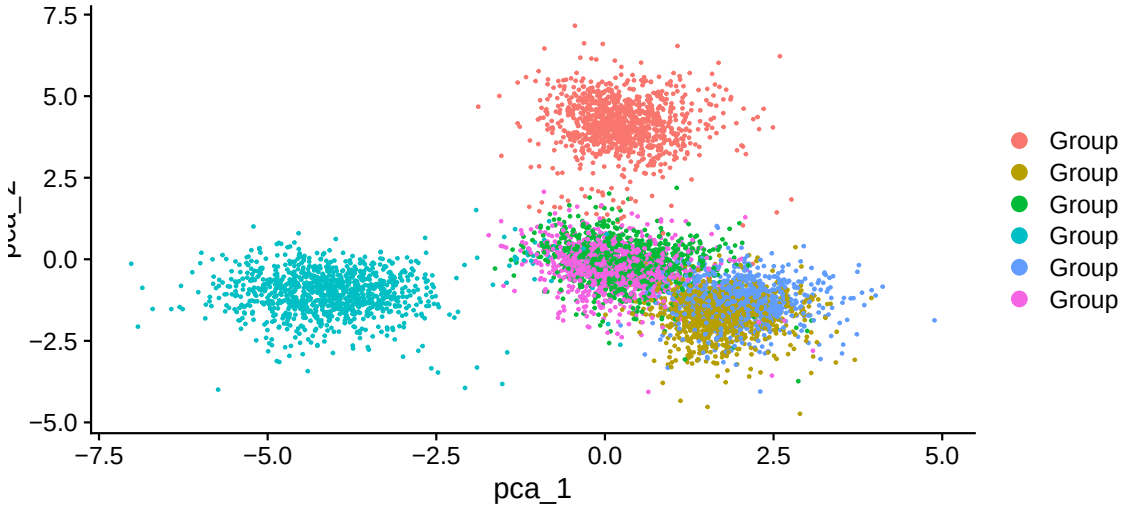
```r
# Impute the top 100 highly variable genes
output <- DropDAE(training_data = count,
                  test_data = count,
                  select_genes = hv_genes,
                  normalization = "sct",
                  num_initializations = 5,
                  seed = seed)

newdata_imputed = t(output$best_x_hat)

# Add imputed data as a new assay in the Seurat object
imputed <- sim2.s
imputed[["imputed"]] <- CreateAssayObject(data = newdata_imputed)
DefaultAssay(imputed) <- "imputed"
```

```r
# Process the imputed Seurat object (Normalization, PCA, and UMAP)
# imputed <- FindVariableFeatures(imputed, verbose = FALSE)
imputed <- ScaleData(imputed, assay = "imputed")
imputed <- RunPCA(imputed, features = VariableFeatures(sim2.s), verbose = FALSE,
                  reduction.name = "pca", reduction.key = "pca_")
imputed <- RunUMAP(imputed, dims = 1:10, verbose = FALSE)
```

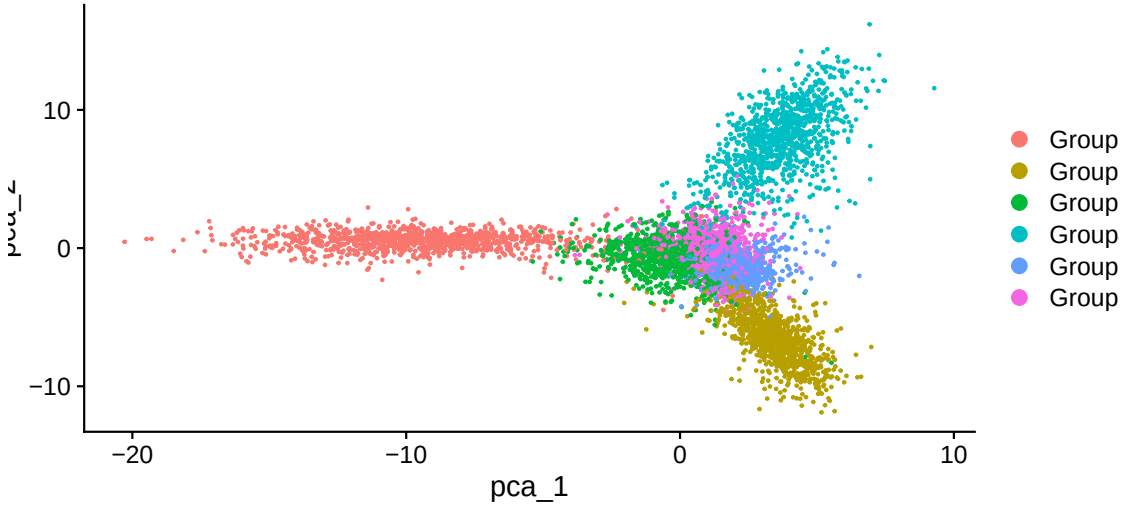
```r
# Assess performance with PCA plots
p1 <- DimPlot(sim1.s, reduction = "pca", group.by = "Group") + ggtitle("Without dropout")
p2 <- DimPlot(sim2.s, reduction = "pca", group.by = "Group") + ggtitle("With dropout")
p3 <- DimPlot(imputed, reduction = "pca", group.by = "Group") + ggtitle("Imputed by CCI")
summary_pca <- ggarrange(p1, p2, p3, nrow = 3, ncol = 1)
print(summary_pca)
```



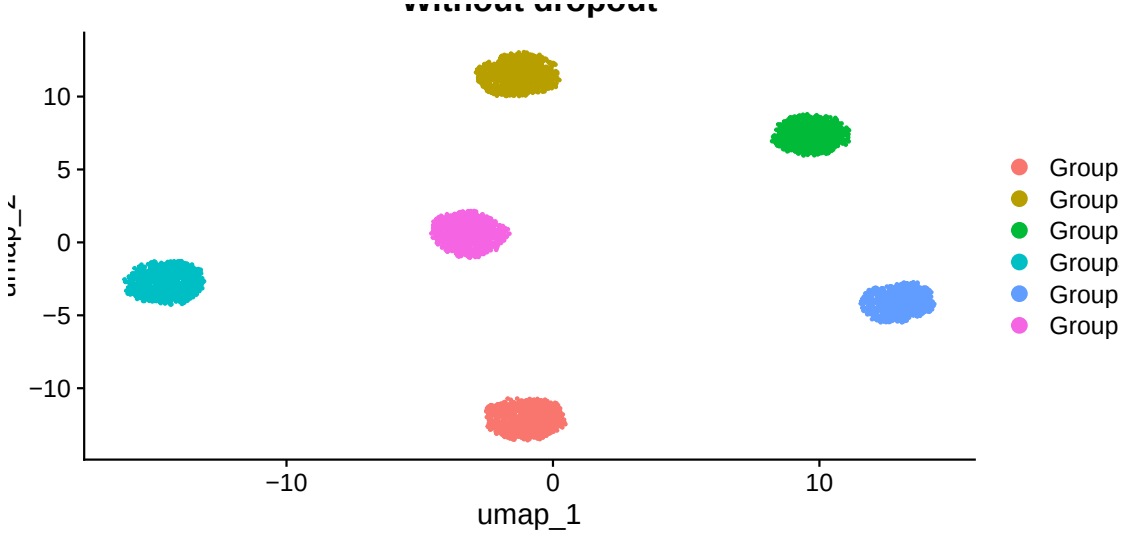
With dropout



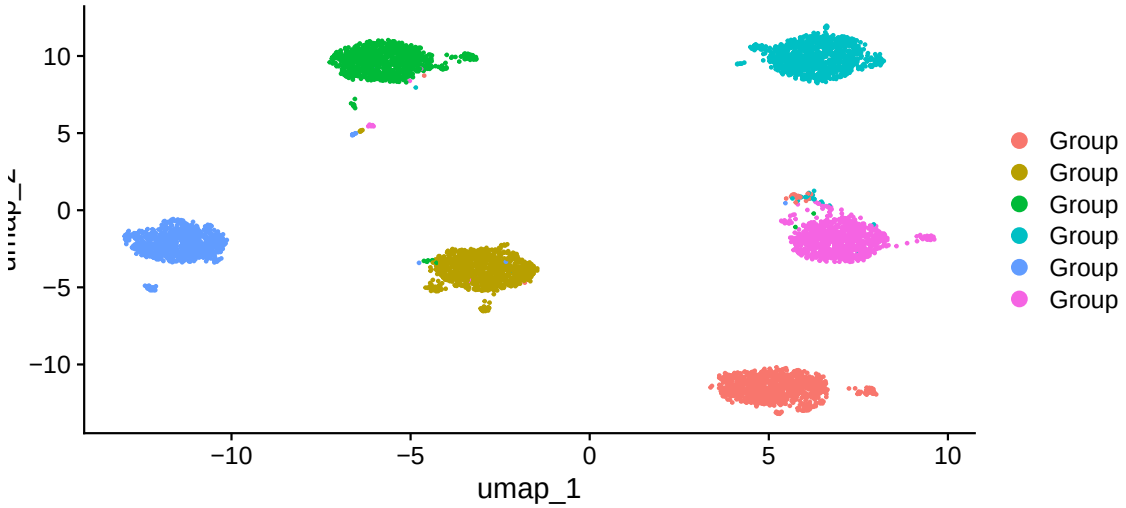
Imputed by CCI



```r
# Assess performance with UMAP plots
p1 <- DimPlot(sim1.s, reduction = "umap", group.by = "Group") + ggtitle("Without dropout")
p2 <- DimPlot(sim2.s, reduction = "umap", group.by = "Group") + ggtitle("With dropout")
p3 <- DimPlot(imputed, reduction = "umap", group.by = "Group") + ggtitle("Imputed by CCI")
summary_umap <- ggarrange(p1, p2, p3, nrow = 3, ncol = 1)
print(summary_umap)
```



With dropout



Imputed by CCI