scGPS introduction

Quan Nguyen

2018-12-14

Contents

1. Installation instruction

```
# Prior to installing scGPS you need to install the SummarizedExperiment
# bioconductor package as the following
# source('https://bioconductor.org/biocLite.R') biocLite('SummarizedExperiment')

# To install scGPS from github (Depending on the configuration of the local
# computer or HPC, possible custom C++ compilation may be required - see
# installation trouble-shootings below)
devtools::install_github("IMB-Computational-Genomics-Lab/scGPS")

# for C++ compilation trouble-shooting, manual download and installation can be
# done from github

git clone https://github.com/IMB-Computational-Genomics-Lab/scGPS

# then check in scGPS/src if any of the precompiled (e.g.  those with *.so and
# *.o) files exist and delete them before recompiling

# create a Makevars file in the scGPS/src with one line: PKG_LIBS =
# $(LAPACK_LIBS) $(BLAS_LIBS) $(FLIBS)

# then with the scGPS as the R working directory, manually recompile scGPS in R
# using devtools to load and install functions
```

```
devtools::document()
# update the NAMESPACE using the update_NAMESPACE.sh
sh update_NAMESPACE.sh
#for window system, to update the NAMESPACE: copy and paste the content of the file NAMESPACE_toAdd_cpp

#load the package to the workspace
devtools::load_all()
```

2. A simple workflow of the scGPS:

The purpose of this workflow is to solve the following task: given a mixed population with known subpopulations, estimate transition scores between these subpopulation

2.1 Setup scGPS objects

```
# load mixed population 1 (loaded from sample1 dataset, named it as day2)
# setwd('/Users/quan.nguyen/Documents/Powell_group_MacQuan/AllCodes/scGPS/vignettes/')
devtools::load_all()

day2 <- sample1
mixedpop1 <- NewscGPS(ExpressionMatrix = day2$dat2_counts, GeneMetadata = day2$dat2geneInfo,
    CellMetadata = day2$dat2_clusters)

# load mixed population 2 (loaded from sample2 dataset, named it as day5)
day5 <- sample2
mixedpop2 <- NewscGPS(ExpressionMatrix = day5$dat5_counts, GeneMetadata = day5$dat5geneInfo,
    CellMetadata = day5$dat5_clusters)
```

2.2 Run predictions

```
# select a subpopulation
c_selectID <- 1
# load gene list (this can be any lists of user selected genes)
genes <- GeneList
genes <- genes$Merged_unique
# load cluster information
cluster_mixedpop1 <- colData(mixedpop1)[,1]
cluster_mixedpop2 <- colData(mixedpop2)[,1]
#run training
LSOLDA_dat <- bootstrap_scGPS(nboots = 2, mixedpop1 = mixedpop1,
    mixedpop2 = mixedpop2, genes = genes, c_selectID  = c_selectID, listData = list(),
     cluster_mixedpop1 = cluster_mixedpop1,
     cluster_mixedpop2 = cluster_mixedpop2)
#>
#> Call:  glmnet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>        Df       %Dev    Lambda
#>   [1,]  0 -2.563e-15 3.285e-01
```

```
#>   [2,]  2  5.699e-02 2.993e-01
#>   [3,]  2  1.108e-01 2.727e-01
#>   [4,]  2  1.565e-01 2.485e-01
#>   [5,]  2  1.956e-01 2.264e-01
#>   [6,]  2  2.292e-01 2.063e-01
#>   [7,]  2  2.583e-01 1.880e-01
#>   [8,]  2  2.836e-01 1.713e-01
#>   [9,]  2  3.057e-01 1.561e-01
#>  [10,]  2  3.250e-01 1.422e-01
#>  [11,]  2  3.419e-01 1.296e-01
#>  [12,]  2  3.567e-01 1.181e-01
#>  [13,]  3  3.712e-01 1.076e-01
#>  [14,]  4  3.861e-01 9.801e-02
#>  [15,]  4  4.004e-01 8.930e-02
#>  [16,]  5  4.137e-01 8.137e-02
#>  [17,]  5  4.256e-01 7.414e-02
#>  [18,]  6  4.376e-01 6.755e-02
#>  [19,]  7  4.508e-01 6.155e-02
#>  [20,]  7  4.628e-01 5.608e-02
#>  [21,]  8  4.735e-01 5.110e-02
#>  [22,]  9  4.833e-01 4.656e-02
#>  [23,] 10  4.926e-01 4.243e-02
#>  [24,] 12  5.033e-01 3.866e-02
#>  [25,] 14  5.145e-01 3.522e-02
#>  [26,] 16  5.257e-01 3.209e-02
#>  [27,] 18  5.365e-01 2.924e-02
#>  [28,] 20  5.484e-01 2.664e-02
#>  [29,] 21  5.612e-01 2.428e-02
#>  [30,] 24  5.730e-01 2.212e-02
#>  [31,] 28  5.850e-01 2.016e-02
#>  [32,] 28  5.973e-01 1.837e-02
#>  [33,] 28  6.083e-01 1.673e-02
#>  [34,] 29  6.182e-01 1.525e-02
#>  [35,] 30  6.273e-01 1.389e-02
#>  [36,] 35  6.365e-01 1.266e-02
#>  [37,] 38  6.460e-01 1.153e-02
#>  [38,] 41  6.570e-01 1.051e-02
#>  [39,] 43  6.686e-01 9.576e-03
#>  [40,] 46  6.806e-01 8.725e-03
#>  [41,] 48  6.922e-01 7.950e-03
#>  [42,] 48  7.032e-01 7.244e-03
#>  [43,] 49  7.136e-01 6.600e-03
#>  [44,] 51  7.234e-01 6.014e-03
#>  [45,] 54  7.333e-01 5.480e-03
#>  [46,] 57  7.437e-01 4.993e-03
#>  [47,] 58  7.542e-01 4.549e-03
#>  [48,] 63  7.645e-01 4.145e-03
#>  [49,] 65  7.746e-01 3.777e-03
#>  [50,] 66  7.842e-01 3.441e-03
#>  [51,] 69  7.935e-01 3.136e-03
#>  [52,] 68  8.030e-01 2.857e-03
#>  [53,] 69  8.119e-01 2.603e-03
#>  [54,] 69  8.203e-01 2.372e-03
```

```
#>  [55,] 69  8.287e-01 2.161e-03
#>  [56,] 72  8.375e-01 1.969e-03
#>  [57,] 72  8.472e-01 1.794e-03
#>  [58,] 72  8.572e-01 1.635e-03
#>  [59,] 75  8.675e-01 1.490e-03
#>  [60,] 75  8.775e-01 1.357e-03
#>  [61,] 76  8.870e-01 1.237e-03
#>  [62,] 75  8.961e-01 1.127e-03
#>  [63,] 77  9.048e-01 1.027e-03
#>  [64,] 78  9.131e-01 9.355e-04
#>  [65,] 77  9.206e-01 8.524e-04
#>  [66,] 77  9.277e-01 7.767e-04
#>  [67,] 77  9.340e-01 7.077e-04
#>  [68,] 76  9.399e-01 6.448e-04
#>  [69,] 75  9.453e-01 5.875e-04
#>  [70,] 75  9.501e-01 5.354e-04
#>  [71,] 75  9.546e-01 4.878e-04
#>  [72,] 75  9.587e-01 4.445e-04
#>  [73,] 75  9.624e-01 4.050e-04
#>  [74,] 75  9.657e-01 3.690e-04
#>  [75,] 76  9.688e-01 3.362e-04
#>  [76,] 76  9.716e-01 3.063e-04
#>  [77,] 77  9.741e-01 2.791e-04
#>  [78,] 77  9.764e-01 2.543e-04
#>  [79,] 77  9.786e-01 2.317e-04
#>  [80,] 77  9.805e-01 2.112e-04
#>  [81,] 77  9.822e-01 1.924e-04
#>  [82,] 77  9.838e-01 1.753e-04
#>  [83,] 77  9.853e-01 1.597e-04
#>  [84,] 78  9.866e-01 1.455e-04
#>  [85,] 78  9.878e-01 1.326e-04
#>  [86,] 78  9.889e-01 1.208e-04
#>  [87,] 78  9.899e-01 1.101e-04
#>  [88,] 78  9.908e-01 1.003e-04
#>  [89,] 78  9.916e-01 9.140e-05
#>  [90,] 78  9.923e-01 8.328e-05
#>  [91,] 78  9.930e-01 7.588e-05
#>  [92,] 78  9.936e-01 6.914e-05
#>  [93,] 78  9.942e-01 6.300e-05
#>  [94,] 78  9.947e-01 5.740e-05
#>  [95,] 79  9.951e-01 5.230e-05
#>  [96,] 80  9.956e-01 4.766e-05
#>  [97,] 80  9.960e-01 4.342e-05
#>  [98,] 80  9.963e-01 3.957e-05
#>  [99,] 80  9.966e-01 3.605e-05
#> [100,] 79  9.969e-01 3.285e-05
#> [1] "done bootstrap 1"
#>
#> Call:  glmnet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>        Df       %Dev    Lambda
#>   [1,]  0 -2.563e-15 3.209e-01
#>   [2,]  1  5.055e-02 2.924e-01
```

```
#>   [3,]  1  9.304e-02 2.664e-01
#>   [4,]  1  1.291e-01 2.427e-01
#>   [5,]  2  1.650e-01 2.212e-01
#>   [6,]  2  1.958e-01 2.015e-01
#>   [7,]  2  2.223e-01 1.836e-01
#>   [8,]  2  2.452e-01 1.673e-01
#>   [9,]  2  2.650e-01 1.525e-01
#>  [10,]  2  2.821e-01 1.389e-01
#>  [11,]  2  2.970e-01 1.266e-01
#>  [12,]  2  3.100e-01 1.153e-01
#>  [13,]  2  3.212e-01 1.051e-01
#>  [14,]  2  3.310e-01 9.574e-02
#>  [15,]  3  3.406e-01 8.724e-02
#>  [16,]  5  3.544e-01 7.949e-02
#>  [17,]  7  3.688e-01 7.243e-02
#>  [18,]  7  3.824e-01 6.599e-02
#>  [19,]  9  3.963e-01 6.013e-02
#>  [20,]  9  4.091e-01 5.479e-02
#>  [21,] 12  4.262e-01 4.992e-02
#>  [22,] 12  4.424e-01 4.549e-02
#>  [23,] 13  4.570e-01 4.145e-02
#>  [24,] 15  4.713e-01 3.776e-02
#>  [25,] 17  4.847e-01 3.441e-02
#>  [26,] 19  4.992e-01 3.135e-02
#>  [27,] 21  5.134e-01 2.857e-02
#>  [28,] 21  5.270e-01 2.603e-02
#>  [29,] 26  5.409e-01 2.372e-02
#>  [30,] 27  5.550e-01 2.161e-02
#>  [31,] 29  5.683e-01 1.969e-02
#>  [32,] 34  5.823e-01 1.794e-02
#>  [33,] 36  5.962e-01 1.635e-02
#>  [34,] 38  6.093e-01 1.489e-02
#>  [35,] 43  6.241e-01 1.357e-02
#>  [36,] 45  6.385e-01 1.237e-02
#>  [37,] 46  6.520e-01 1.127e-02
#>  [38,] 49  6.648e-01 1.027e-02
#>  [39,] 52  6.771e-01 9.354e-03
#>  [40,] 52  6.887e-01 8.523e-03
#>  [41,] 54  6.997e-01 7.766e-03
#>  [42,] 56  7.106e-01 7.076e-03
#>  [43,] 56  7.211e-01 6.448e-03
#>  [44,] 58  7.310e-01 5.875e-03
#>  [45,] 57  7.404e-01 5.353e-03
#>  [46,] 57  7.492e-01 4.877e-03
#>  [47,] 58  7.574e-01 4.444e-03
#>  [48,] 59  7.651e-01 4.049e-03
#>  [49,] 60  7.724e-01 3.690e-03
#>  [50,] 62  7.792e-01 3.362e-03
#>  [51,] 64  7.863e-01 3.063e-03
#>  [52,] 64  7.947e-01 2.791e-03
#>  [53,] 64  8.026e-01 2.543e-03
#>  [54,] 67  8.106e-01 2.317e-03
#>  [55,] 70  8.189e-01 2.111e-03
```

```
#>  [56,] 69  8.269e-01 1.924e-03
#>  [57,] 70  8.351e-01 1.753e-03
#>  [58,] 71  8.433e-01 1.597e-03
#>  [59,] 72  8.518e-01 1.455e-03
#>  [60,] 73  8.603e-01 1.326e-03
#>  [61,] 73  8.685e-01 1.208e-03
#>  [62,] 75  8.764e-01 1.101e-03
#>  [63,] 76  8.849e-01 1.003e-03
#>  [64,] 76  8.931e-01 9.139e-04
#>  [65,] 76  9.011e-01 8.327e-04
#>  [66,] 75  9.088e-01 7.588e-04
#>  [67,] 76  9.162e-01 6.914e-04
#>  [68,] 76  9.230e-01 6.299e-04
#>  [69,] 75  9.294e-01 5.740e-04
#>  [70,] 77  9.354e-01 5.230e-04
#>  [71,] 78  9.410e-01 4.765e-04
#>  [72,] 79  9.462e-01 4.342e-04
#>  [73,] 79  9.509e-01 3.956e-04
#>  [74,] 80  9.553e-01 3.605e-04
#>  [75,] 80  9.595e-01 3.284e-04
#>  [76,] 80  9.633e-01 2.993e-04
#>  [77,] 80  9.666e-01 2.727e-04
#>  [78,] 80  9.697e-01 2.485e-04
#>  [79,] 79  9.724e-01 2.264e-04
#>  [80,] 78  9.750e-01 2.063e-04
#>  [81,] 78  9.772e-01 1.879e-04
#>  [82,] 79  9.793e-01 1.713e-04
#>  [83,] 79  9.811e-01 1.560e-04
#>  [84,] 78  9.828e-01 1.422e-04
#>  [85,] 78  9.844e-01 1.295e-04
#>  [86,] 79  9.858e-01 1.180e-04
#>  [87,] 79  9.870e-01 1.076e-04
#>  [88,] 79  9.882e-01 9.800e-05
#>  [89,] 79  9.893e-01 8.929e-05
#>  [90,] 79  9.902e-01 8.136e-05
#>  [91,] 78  9.911e-01 7.413e-05
#>  [92,] 78  9.919e-01 6.755e-05
#>  [93,] 78  9.926e-01 6.155e-05
#>  [94,] 78  9.932e-01 5.608e-05
#>  [95,] 77  9.938e-01 5.110e-05
#>  [96,] 77  9.944e-01 4.656e-05
#>  [97,] 77  9.948e-01 4.242e-05
#>  [98,] 78  9.953e-01 3.865e-05
#>  [99,] 78  9.957e-01 3.522e-05
#> [100,] 78  9.961e-01 3.209e-05
#> [1] "done bootstrap 2"
```

2.3 Summarise results

```
# display the list of result information in the LASOLDA_dat object
names(LSOLDA_dat)
```

```
#> [1] "Accuracy"          "ElasticNetGenes"   "Deviance"
#> [4] "ElasticNetFit"     "LDAFit"            "predictor_S1"
#> [7] "ElasticNetPredict" "LDAPredict"
LSOLDA_dat$ElasticNetPredict
#> [[1]]
#> [[1]][[1]]
#> [1] "ElasticNet for subpop1 in target mixedpop2"
#>
#> [[1]][[2]]
#> numeric(0)
#>
#> [[1]][[3]]
#> [1] "ElasticNet for subpop2 in target mixedpop2"
#>
#> [[1]][[4]]
#> [1] 17.85714
#>
#> [[1]][[5]]
#> [1] "ElasticNet for subpop3 in target mixedpop2"
#>
#> [[1]][[6]]
#> numeric(0)
#>
#> [[1]][[7]]
#> [1] "ElasticNet for subpop4 in target mixedpop2"
#>
#> [[1]][[8]]
#> [1] 10
#>
#>
#> [[2]]
#> [[2]][[1]]
#> [1] "ElasticNet for subpop1 in target mixedpop2"
#>
#> [[2]][[2]]
#> [1] 90.90909
#>
#> [[2]][[3]]
#> [1] "ElasticNet for subpop2 in target mixedpop2"
#>
#> [[2]][[4]]
#> [1] 98.57143
#>
#> [[2]][[5]]
#> [1] "ElasticNet for subpop3 in target mixedpop2"
#>
#> [[2]][[6]]
#> [1] 90.22556
#>
#> [[2]][[7]]
#> [1] "ElasticNet for subpop4 in target mixedpop2"
#>
#> [[2]][[8]]
```

```
#> [1] 95
LSOLDA_dat$LDAPredict
#> [[1]]
#> [[1]][[1]]
#> [1] "LDA for subpop 1 in target mixedpop2"
#>
#> [[1]][[2]]
#> [1] NaN
#>
#> [[1]][[3]]
#> [1] "LDA for subpop 2 in target mixedpop2"
#>
#> [[1]][[4]]
#> [1] NaN
#>
#> [[1]][[5]]
#> [1] "LDA for subpop 3 in target mixedpop2"
#>
#> [[1]][[6]]
#> [1] NaN
#>
#> [[1]][[7]]
#> [1] "LDA for subpop 4 in target mixedpop2"
#>
#> [[1]][[8]]
#> [1] NaN
#>
#>
#> [[2]]
#> [[2]][[1]]
#> [1] "LDA for subpop 1 in target mixedpop2"
#>
#> [[2]][[2]]
#> [1] NaN
#>
#> [[2]][[3]]
#> [1] "LDA for subpop 2 in target mixedpop2"
#>
#> [[2]][[4]]
#> [1] NaN
#>
#> [[2]][[5]]
#> [1] "LDA for subpop 3 in target mixedpop2"
#>
#> [[2]][[6]]
#> [1] NaN
#>
#> [[2]][[7]]
#> [1] "LDA for subpop 4 in target mixedpop2"
#>
#> [[2]][[8]]
#> [1] NaN
```

```
# summary results LDA
summary_prediction_lda(LSOLDA_dat = LSOLDA_dat, nPredSubpop = 4)
#>    V1  V2                                names
#> 1 NaN NaN LDA for subpop 1 in target mixedpop2
#> 2 NaN NaN LDA for subpop 2 in target mixedpop2
#> 3 NaN NaN LDA for subpop 3 in target mixedpop2
#> 4 NaN NaN LDA for subpop 4 in target mixedpop2

# summary results Lasso to show the percent of cells classified as cells belonging
summary_prediction_lasso(LSOLDA_dat = LSOLDA_dat, nPredSubpop = 4)
#>                 V1               V2
#> 1               NA 90.9090909090909
#> 2 17.8571428571429 98.5714285714286
#> 3               NA 90.2255639097744
#> 4               10               95
#>                                      names
#> 1 ElasticNet for subpop1 in target mixedpop2
#> 2 ElasticNet for subpop2 in target mixedpop2
#> 3 ElasticNet for subpop3 in target mixedpop2
#> 4 ElasticNet for subpop4 in target mixedpop2

# summary accuracy to check the model accuracy in the leave-out test set
summary_accuracy(object = LSOLDA_dat)
#> [1] 85.26786 84.82143

# summary maximum deviance explained by the model
summary_deviance(object = LSOLDA_dat)
#> $allDeviance
#> [1] "0.3567" "0.3212"
#>
#> $DeviMax
#>         Dfd   Deviance        DEgenes
#> 1         0 -2.563e-15 genes_cluster1
#> 2         2     0.3567 genes_cluster1
#> 3 remaining          1        DEgenes
#>
#> $LassoGenesMax
#> NULL
```

3. A complete workflow of the scGPS:

The purpose of this workflow is to solve the following task: given an unknown mixed population, find clusters and estimate relationship between clusters

3.1 Identify clusters in a dataset using CORE

(skip this step if clusters are known)

```
# find clustering information in an expresion data using CORE
day5 <- sample2
cellnames <- colnames(day5$dat5_counts)
```

```
cluster <-day5$dat5_clusters
cellnames <-data.frame("Cluster"=cluster, "cellBarcodes" = cellnames)
mixedpop2 <-NewscGPS(ExpressionMatrix = day5$dat5_counts, GeneMetadata = day5$dat5geneInfo, CellMetadata
CORE_cluster <- CORE_scGPS(mixedpop2, remove_outlier = c(0), PCA=FALSE)
#> [1] "Performing 1 round of filtering"
#> [1] "Identifying top variable genes"
#> [1] "Calculating distance matrix"
#> [1] "Performing hierarchical clustering"
#> [1] "Finding clustering information"
#> [1] "No more outliers detected in filtering round 1"
#> [1] "Identifying top variable genes"
#> [1] "Calculating distance matrix"
#> [1] "Performing hierarchical clustering"
#> [1] "Finding clustering information"
#> [1] "writing clustering result for run 1"
#> [1] "writing clustering result for run 2"
#> [1] "writing clustering result for run 3"
#> [1] "writing clustering result for run 4"
#> [1] "writing clustering result for run 5"
#> [1] "writing clustering result for run 6"
#> [1] "writing clustering result for run 7"
#> [1] "writing clustering result for run 8"
#> [1] "writing clustering result for run 9"
#> [1] "writing clustering result for run 10"
#> [1] "writing clustering result for run 11"
#> [1] "writing clustering result for run 12"
#> [1] "writing clustering result for run 13"
#> [1] "writing clustering result for run 14"
#> [1] "writing clustering result for run 15"
#> [1] "writing clustering result for run 16"
#> [1] "writing clustering result for run 17"
#> [1] "writing clustering result for run 18"
#> [1] "writing clustering result for run 19"
#> [1] "writing clustering result for run 20"
#> [1] "writing clustering result for run 21"
#> [1] "writing clustering result for run 22"
#> [1] "writing clustering result for run 23"
#> [1] "writing clustering result for run 24"
#> [1] "writing clustering result for run 25"
#> [1] "writing clustering result for run 26"
#> [1] "writing clustering result for run 27"
#> [1] "writing clustering result for run 28"
#> [1] "writing clustering result for run 29"
#> [1] "writing clustering result for run 30"
#> [1] "writing clustering result for run 31"
#> [1] "writing clustering result for run 32"
#> [1] "writing clustering result for run 33"
#> [1] "writing clustering result for run 34"
#> [1] "writing clustering result for run 35"
#> [1] "writing clustering result for run 36"
#> [1] "writing clustering result for run 37"
#> [1] "writing clustering result for run 38"
#> [1] "writing clustering result for run 39"
```

```
#> [1] "writing clustering result for run 40"
#> [1] "Done clustering, moving to stability calculation..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done finding optimal clustering..."
```

3.1 Identify clusters in a dataset using SCORE (Stable Clustering at Optimal REsolution)

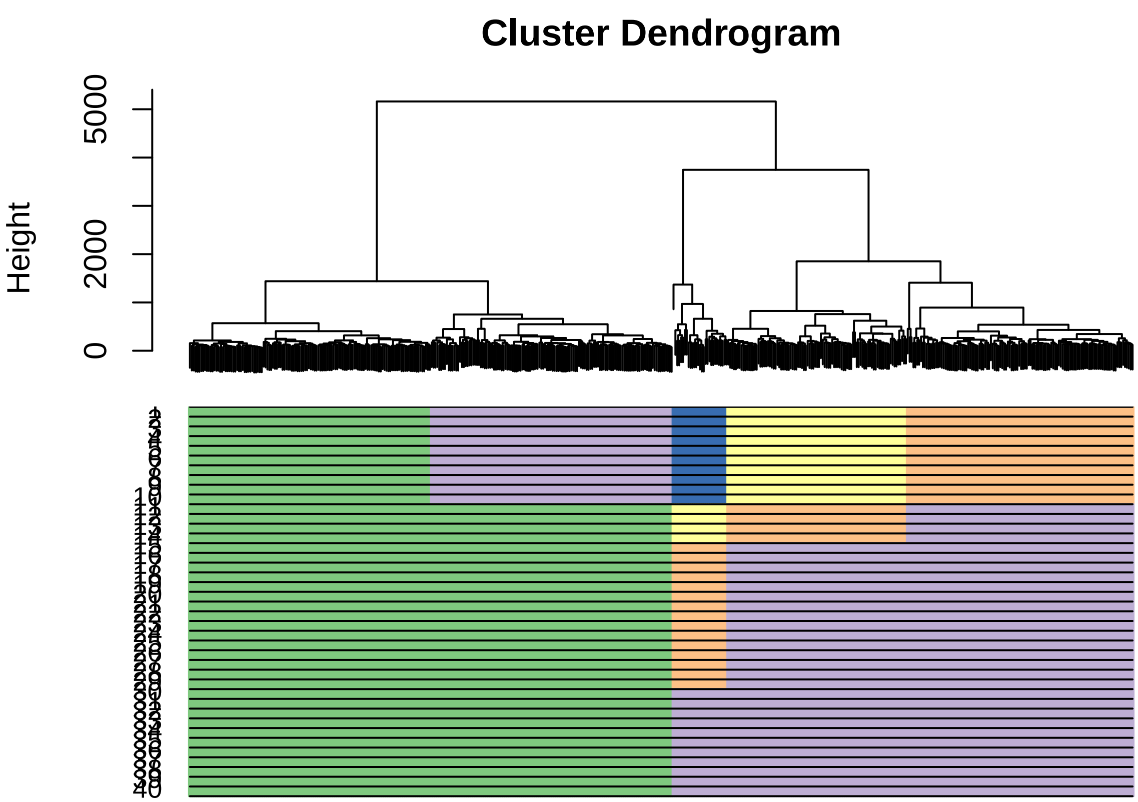
(skip this step if clusters are known) (SCORE aims to get stable subpopulation results, by introducing bagging aggregation and bootstrapping to the CORE algorithm)

```
# find clustering information in an expresion data using SCORE
day5 <- sample2
cellnames <- colnames(day5$dat5_counts)
cluster <-day5$dat5_clusters
cellnames <-data.frame("Cluster"=cluster, "cellBarcodes" = cellnames)
mixedpop2 <-NewscGPS(ExpressionMatrix = day5$dat5_counts, GeneMetadata = day5$dat5geneInfo, CellMetadata

SCORE_test <- CORE_scGPS_bagging(mixedpop2, remove_outlier = c(0), PCA=FALSE,
                          bagging_run = 20, subsample_proportion = .8)
#> [1] "Performing 1 round of filtering"
#> [1] "Identifying top variable genes"
#> [1] "Calculating distance matrix"
#> [1] "Performing hierarchical clustering"
#> [1] "Finding clustering information"
#> [1] "No more outliers detected in filtering round 1"
#> [1] "Identifying top variable genes"
#> [1] "Calculating distance matrix"
#> [1] "Performing hierarchical clustering"
#> [1] "Finding clustering information"
#> [1] "500 cells left after filtering"
#> [1] "Running 20 bagging runs, with 0.8 subsampling..."
#> [1] "Done clustering, moving to stability calculation..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
```

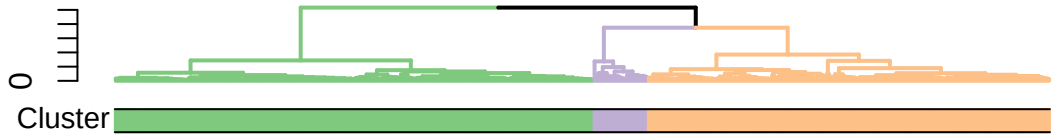
```
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
#> [1] "Done calculating stability..."
#> [1] "Start finding optimal clustering..."
```

3.2 Visualise all cluster results in all iterations

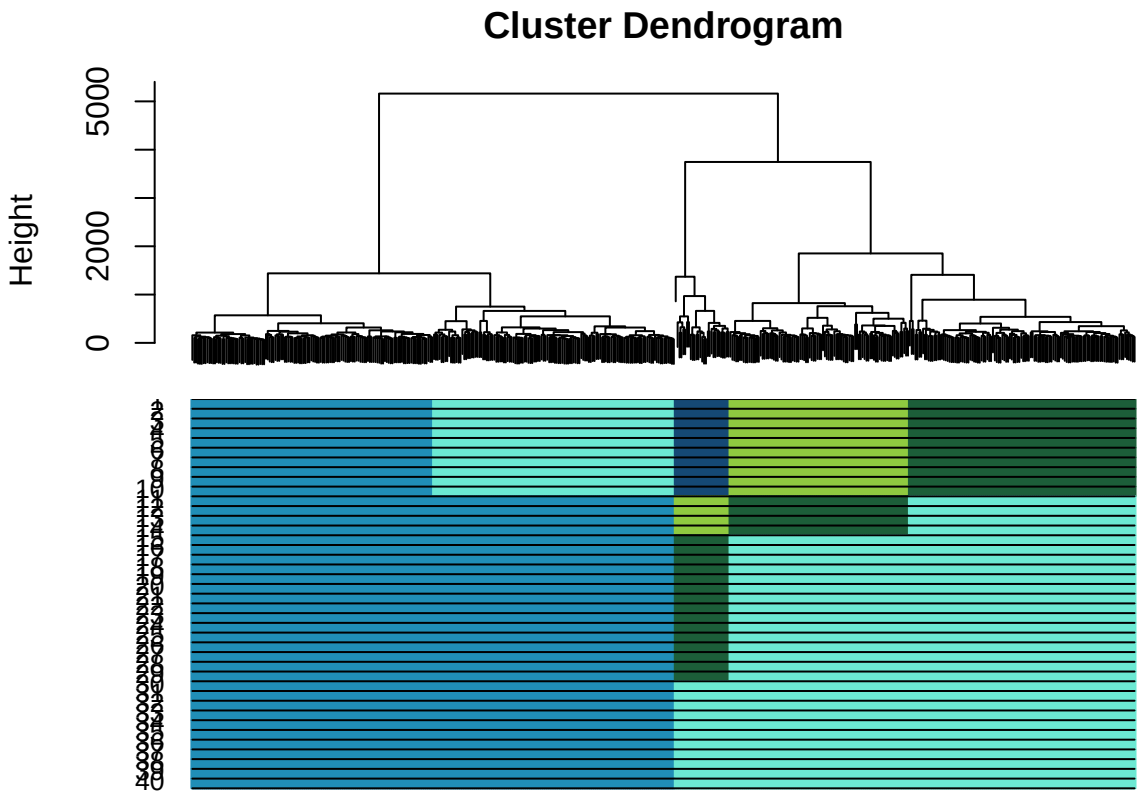
```
##3.2.1 plot CORE clustering
plot_CORE(CORE_cluster$tree, CORE_cluster$Cluster) #plot all clustering bars
```



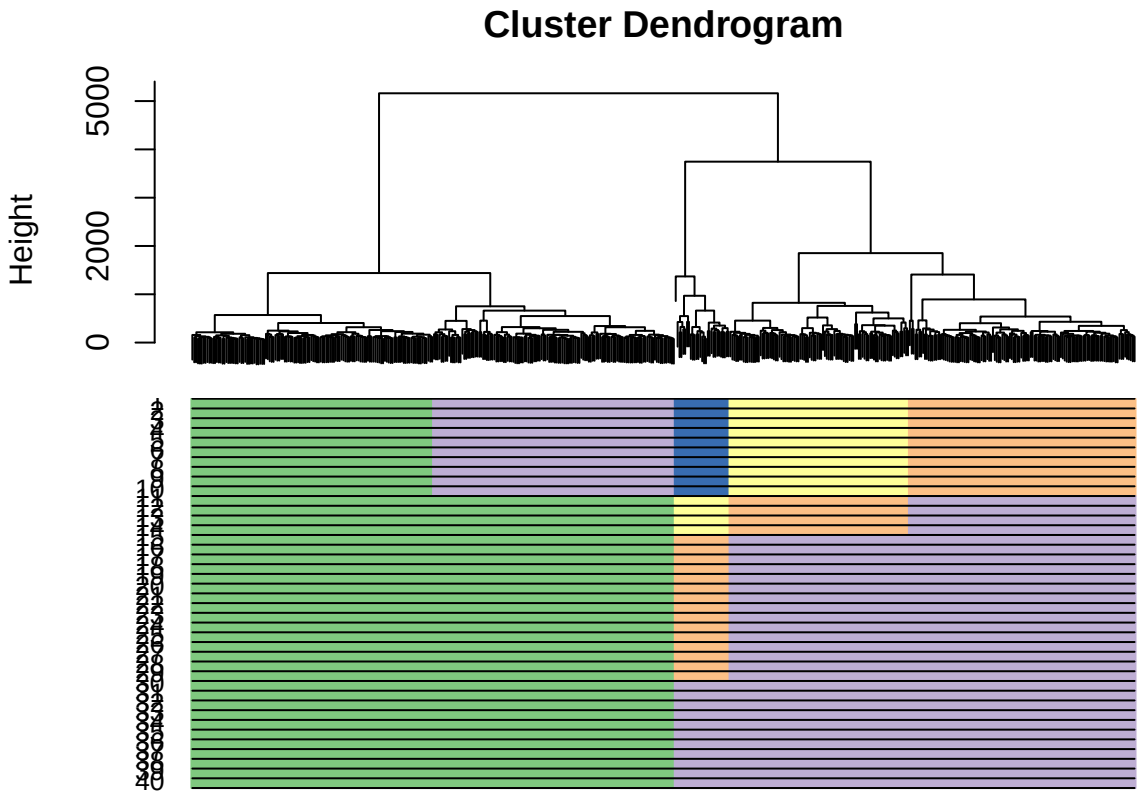
```
#extract optimal index identified by CORE_scGPS
key_height <- CORE_cluster$optimalClust$KeyStats$Height
optimal_res <- CORE_cluster$optimalClust$OptimalRes
optimal_index = which(key_height == optimal_res)
#plot one optimal clustering bar
plot_optimal_CORE(original_tree= CORE_cluster$tree,
                  optimal_cluster = unlist(CORE_cluster$Cluster[optimal_index]), shift = -2000)
#> [1] "Ordering and assigning labels..."
#> [1] 2
#> [1] 128 270  NA
#> [1] 3
#> [1] 128 270 393
#> [1] "Plotting the colored dendrogram now...."
```



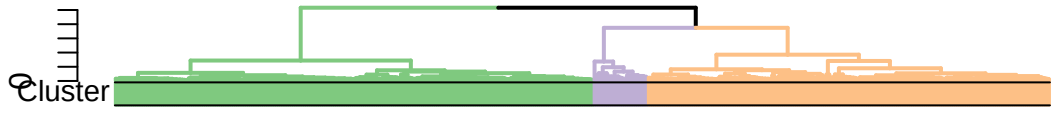

```
#> [1] "Plotting the bar underneath now...."
# you can customise the cluster color bars (provide color_branch values)
plot_CORE(CORE_cluster$tree, CORE_cluster$Cluster, color_branch = c("#208eb7", "#6ce9d3", "#1c5e39", "#8
```




```
##3.2.2 plot SCORE clustering
plot_CORE(SCORE_test$tree, list_clusters = SCORE_test$Cluster)#plot all clustering bars
```



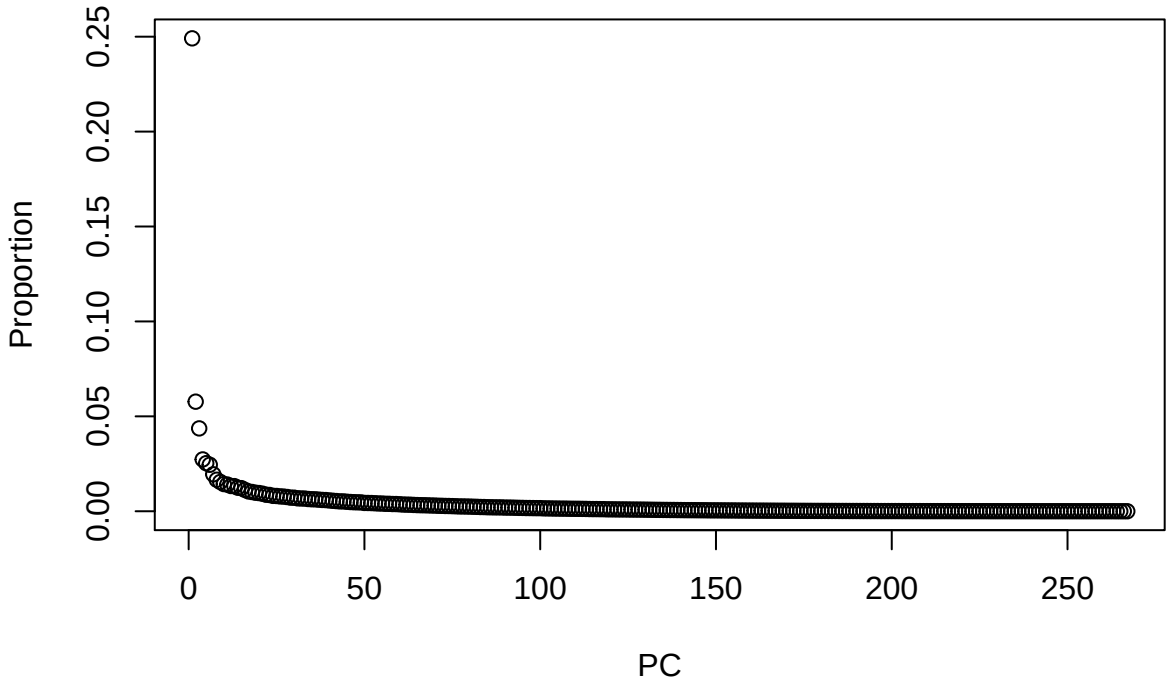

```
#plot one stable optimal clustering bar
plot_optimal_CORE(original_tree= SCORE_test$tree,
                  optimal_cluster = unlist(SCORE_test$Cluster[SCORE_test$optimal_index]), shift = -100)
#> [1] "Ordering and assigning labels..."
#> [1] 2
#> [1] 128 270  NA
#> [1] 3
#> [1] 128 270 393
#> [1] "Plotting the colored dendrogram now...."
```



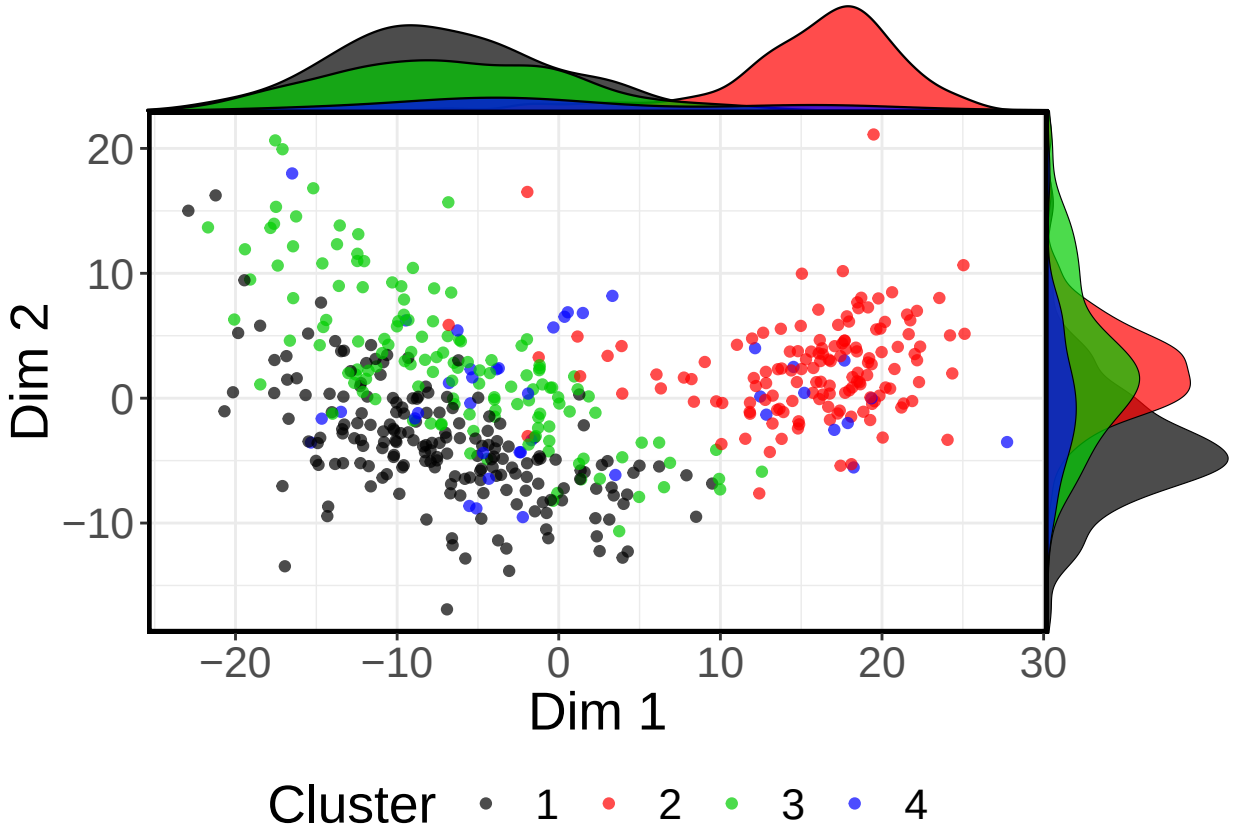

```
#> [1] "Plotting the bar underneath now...."
```

3.4 Compare clustering results with other dimensional reduction methods (e.g., CIDR)

```
library(cidr)
t <- CIDR_scGPS(expression.matrix=assay(mixedpop2))
#> [1] "building cidr object..."
#> [1] "determine dropout candidates..."
#> [1] "determine the imputation weighting threshold..."
#> [1] "computes the _CIDR_ dissimilarity matrix..."
#> [1] "PCA plot with proportion of variance explained..."
```



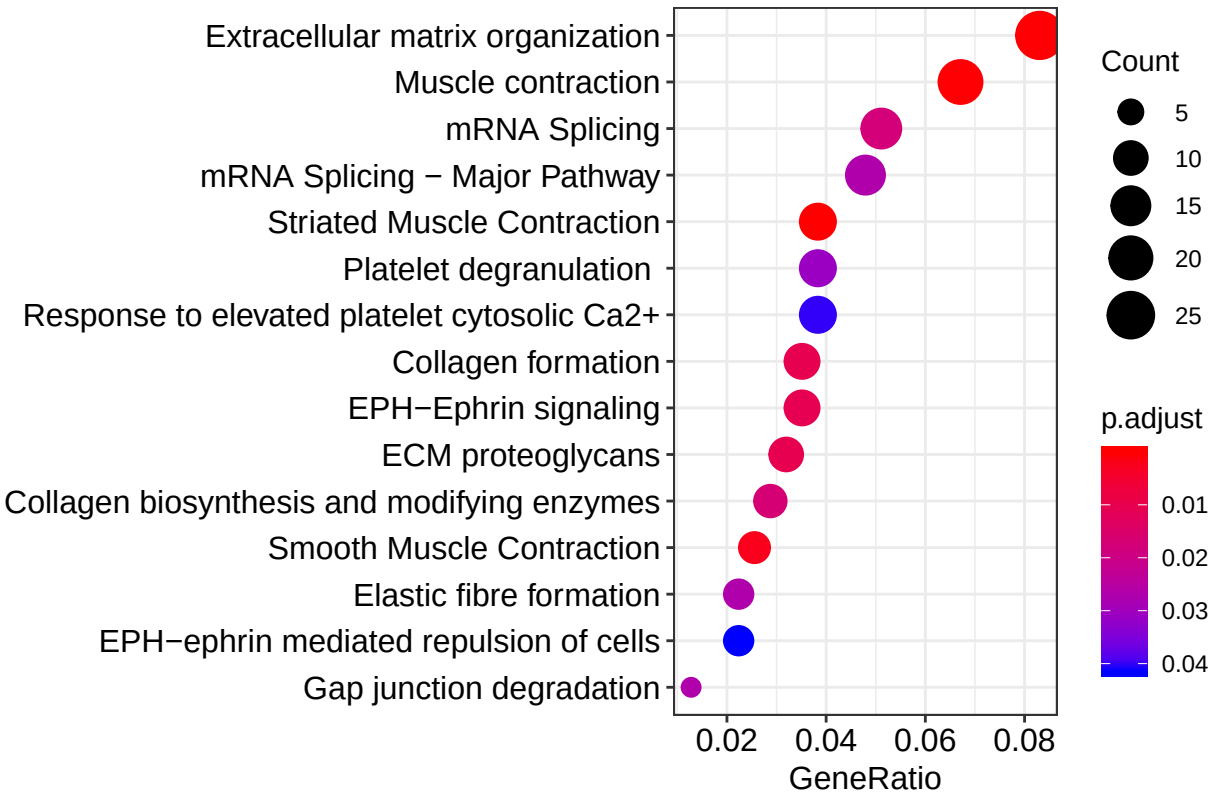

```
#> [1] "find the number of PC..."
#> [1] "perform clustering..."
p2 <-plotReduced_scGPS(t, color_fac = factor(colData(mixedpop2)[,1]),palletes =1:length(unique(colData(m
p2
```



3.5 Find gene markers and annotate clusters

```
#load gene list (this can be any lists of user-selected genes)
genes <-GeneList
genes <-genes$Merged_unique

#the gene list can also be objectively identified by differential expression analysis
#cluster information is requied for findMarkers_scGPS. Here, we use CORE results.

colData(mixedpop2)[,1] <- unlist(SCORE_test$Cluster[SCORE_test$optimal_index])

suppressMessages(library(locfit))
suppressMessages(library(DESeq))

DEgenes <- findMarkers_scGPS(expression_matrix=assay(mixedpop2), cluster = colData(mixedpop2)[,1],
                             selected_cluster=unique(colData(mixedpop2)[,1]))
#> [1] "Start estimate dispersions for cluster 1..."
#> [1] "Done estimate dispersions. Start nbinom test for cluster 1..."
#> [1] "Done nbinom test for cluster 1 ..."
#> [1] "Adjust foldchange by subtracting basemean to 1..."
#> [1] "Start estimate dispersions for cluster 2..."
#> [1] "Done estimate dispersions. Start nbinom test for cluster 2..."
#> [1] "Done nbinom test for cluster 2 ..."
#> [1] "Adjust foldchange by subtracting basemean to 1..."
#> [1] "Start estimate dispersions for cluster 3..."
```

```
#> [1] "Done estimate dispersions. Start nbinom test for cluster 3..."
#> [1] "Done nbinom test for cluster 3 ..."
#> [1] "Adjust foldchange by subtracting basemean to 1..."

#the output contains dataframes for each cluster.
#the data frame contains all genes, sorted by p-values
names(DEgenes)
#> [1] "DE_Subpop1vsRemaining" "DE_Subpop2vsRemaining" "DE_Subpop3vsRemaining"

#you can annotate the identified clusters
DEgeneList_3vsOthers <- DEgenes$DE_Subpop3vsRemaining$id

#users need to check the format of the gene input to make sure they are consistent to
#the gene names in the expression matrix
DEgeneList_3vsOthers <-gsub("_.*", "", DEgeneList_3vsOthers )

#the following command saves the file "PathwayEnrichment.xlsx" to the working dir
#use 500 top DE genes
suppressMessages(library(DOSE))
suppressMessages(library(ReactomePA))
suppressMessages(library(clusterProfiler))
enrichment_test <- annotate_scGPS(DEgeneList_3vsOthers[1:500], pvalueCutoff=0.05, gene_symbol=TRUE)
#> [1] "Original gene number in geneList"
#> [1] 500
#> [1] "Number of genes successfully converted"
#> [1] 488

#the enrichment outputs can be displayed by running
dotplot(enrichment_test, showCategory=15)
```



4. Relationship between clusters within one sample or between two samples

The purpose of this workflow is to solve the following task: given one or two unknown mixed population(s) and clusters in each mixed population, estimate and visualise relationship between clusters

4.1 Start the scGPS prediction to find relationship between clusters

```
#select a subpopulation, and input gene list
c_selectID <- 1
genes = DEgenes$DE_Subpop1vsRemaining$id[1:500]
#format gene names
genes <- gsub("_.*", "", genes)

#run the test bootstrap with nboots = 2 runs

cluster_mixedpop1 <- colData(mixedpop1)[,1]
cluster_mixedpop2 <- colData(mixedpop2)[,1]

sink("temp")
LSOLDA_dat <- bootstrap_scGPS(nboots = 2, mixedpop1 = mixedpop1,
    mixedpop2 = mixedpop2, genes = genes, c_selectID  = c_selectID, listData = list(),
     cluster_mixedpop1 = cluster_mixedpop1,
     cluster_mixedpop2 = cluster_mixedpop2)
```

```
#>
#> Call:  glmnet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>        Df       %Dev   Lambda
#>   [1,]  0 -2.563e-15 0.284000
#>   [2,]  1  2.069e-02 0.271100
#>   [3,]  2  4.588e-02 0.258800
#>   [4,]  3  7.166e-02 0.247000
#>   [5,]  4  9.689e-02 0.235800
#>   [6,]  4  1.202e-01 0.225100
#>   [7,]  4  1.418e-01 0.214900
#>   [8,]  5  1.622e-01 0.205100
#>   [9,]  5  1.834e-01 0.195800
#>  [10,]  5  2.031e-01 0.186900
#>  [11,]  6  2.222e-01 0.178400
#>  [12,]  7  2.411e-01 0.170300
#>  [13,]  7  2.614e-01 0.162500
#>  [14,]  8  2.809e-01 0.155100
#>  [15,]  8  2.994e-01 0.148100
#>  [16,]  9  3.170e-01 0.141400
#>  [17,]  9  3.338e-01 0.134900
#>  [18,]  9  3.497e-01 0.128800
#>  [19,] 10  3.651e-01 0.122900
#>  [20,] 11  3.801e-01 0.117400
#>  [21,] 11  3.956e-01 0.112000
#>  [22,] 12  4.104e-01 0.106900
#>  [23,] 12  4.249e-01 0.102100
#>  [24,] 13  4.391e-01 0.097430
#>  [25,] 13  4.527e-01 0.093000
#>  [26,] 14  4.657e-01 0.088780
#>  [27,] 14  4.785e-01 0.084740
#>  [28,] 14  4.908e-01 0.080890
#>  [29,] 17  5.036e-01 0.077210
#>  [30,] 19  5.175e-01 0.073700
#>  [31,] 19  5.311e-01 0.070350
#>  [32,] 21  5.441e-01 0.067160
#>  [33,] 21  5.568e-01 0.064100
#>  [34,] 21  5.689e-01 0.061190
#>  [35,] 21  5.805e-01 0.058410
#>  [36,] 22  5.917e-01 0.055750
#>  [37,] 22  6.026e-01 0.053220
#>  [38,] 22  6.131e-01 0.050800
#>  [39,] 23  6.235e-01 0.048490
#>  [40,] 24  6.337e-01 0.046290
#>  [41,] 28  6.439e-01 0.044180
#>  [42,] 30  6.539e-01 0.042180
#>  [43,] 31  6.638e-01 0.040260
#>  [44,] 32  6.734e-01 0.038430
#>  [45,] 36  6.831e-01 0.036680
#>  [46,] 40  6.940e-01 0.035020
#>  [47,] 43  7.049e-01 0.033420
#>  [48,] 46  7.156e-01 0.031900
#>  [49,] 47  7.264e-01 0.030450
```

```
#>  [50,] 50  7.369e-01 0.029070
#>  [51,] 51  7.472e-01 0.027750
#>  [52,] 53  7.570e-01 0.026490
#>  [53,] 54  7.666e-01 0.025280
#>  [54,] 54  7.758e-01 0.024130
#>  [55,] 53  7.846e-01 0.023040
#>  [56,] 53  7.930e-01 0.021990
#>  [57,] 55  8.012e-01 0.020990
#>  [58,] 55  8.093e-01 0.020040
#>  [59,] 55  8.170e-01 0.019130
#>  [60,] 55  8.243e-01 0.018260
#>  [61,] 54  8.314e-01 0.017430
#>  [62,] 55  8.381e-01 0.016640
#>  [63,] 57  8.446e-01 0.015880
#>  [64,] 60  8.510e-01 0.015160
#>  [65,] 61  8.574e-01 0.014470
#>  [66,] 60  8.636e-01 0.013810
#>  [67,] 61  8.694e-01 0.013180
#>  [68,] 61  8.750e-01 0.012580
#>  [69,] 61  8.803e-01 0.012010
#>  [70,] 62  8.855e-01 0.011470
#>  [71,] 62  8.904e-01 0.010940
#>  [72,] 62  8.951e-01 0.010450
#>  [73,] 64  8.996e-01 0.009972
#>  [74,] 65  9.041e-01 0.009519
#>  [75,] 65  9.083e-01 0.009087
#>  [76,] 65  9.124e-01 0.008674
#>  [77,] 67  9.163e-01 0.008279
#>  [78,] 67  9.201e-01 0.007903
#>  [79,] 67  9.236e-01 0.007544
#>  [80,] 68  9.271e-01 0.007201
#>  [81,] 69  9.303e-01 0.006874
#>  [82,] 69  9.335e-01 0.006561
#>  [83,] 69  9.365e-01 0.006263
#>  [84,] 69  9.393e-01 0.005978
#>  [85,] 69  9.420e-01 0.005707
#>  [86,] 69  9.446e-01 0.005447
#>  [87,] 70  9.471e-01 0.005200
#>  [88,] 70  9.495e-01 0.004963
#>  [89,] 70  9.517e-01 0.004738
#>  [90,] 69  9.539e-01 0.004522
#>  [91,] 70  9.560e-01 0.004317
#>  [92,] 72  9.579e-01 0.004121
#>  [93,] 72  9.598e-01 0.003933
#>  [94,] 72  9.616e-01 0.003755
#>  [95,] 72  9.634e-01 0.003584
#>  [96,] 72  9.650e-01 0.003421
#>  [97,] 72  9.666e-01 0.003266
#>  [98,] 72  9.681e-01 0.003117
#>  [99,] 72  9.695e-01 0.002975
#> [100,] 73  9.709e-01 0.002840
#> [1] "done bootstrap 1"
#>
```

```
#> Call:  glmnet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>        Df       %Dev   Lambda
#>   [1,]  0 -2.563e-15 0.293000
#>   [2,]  1  2.202e-02 0.279600
#>   [3,]  2  4.793e-02 0.266900
#>   [4,]  2  7.307e-02 0.254800
#>   [5,]  3  9.930e-02 0.243200
#>   [6,]  3  1.249e-01 0.232200
#>   [7,]  4  1.492e-01 0.221600
#>   [8,]  4  1.734e-01 0.211500
#>   [9,]  5  1.960e-01 0.201900
#>  [10,]  6  2.177e-01 0.192800
#>  [11,]  7  2.387e-01 0.184000
#>  [12,]  7  2.587e-01 0.175600
#>  [13,]  7  2.774e-01 0.167600
#>  [14,]  8  2.957e-01 0.160000
#>  [15,]  8  3.139e-01 0.152800
#>  [16,]  8  3.310e-01 0.145800
#>  [17,]  8  3.471e-01 0.139200
#>  [18,]  8  3.623e-01 0.132900
#>  [19,]  8  3.766e-01 0.126800
#>  [20,]  9  3.911e-01 0.121100
#>  [21,] 10  4.053e-01 0.115600
#>  [22,] 10  4.188e-01 0.110300
#>  [23,] 10  4.317e-01 0.105300
#>  [24,] 11  4.442e-01 0.100500
#>  [25,] 11  4.570e-01 0.095930
#>  [26,] 16  4.718e-01 0.091570
#>  [27,] 16  4.863e-01 0.087410
#>  [28,] 17  5.002e-01 0.083440
#>  [29,] 17  5.134e-01 0.079640
#>  [30,] 17  5.260e-01 0.076020
#>  [31,] 17  5.379e-01 0.072570
#>  [32,] 17  5.493e-01 0.069270
#>  [33,] 19  5.606e-01 0.066120
#>  [34,] 19  5.719e-01 0.063120
#>  [35,] 19  5.826e-01 0.060250
#>  [36,] 20  5.930e-01 0.057510
#>  [37,] 21  6.032e-01 0.054900
#>  [38,] 21  6.131e-01 0.052400
#>  [39,] 21  6.227e-01 0.050020
#>  [40,] 22  6.324e-01 0.047750
#>  [41,] 22  6.421e-01 0.045580
#>  [42,] 22  6.515e-01 0.043500
#>  [43,] 24  6.605e-01 0.041530
#>  [44,] 24  6.695e-01 0.039640
#>  [45,] 26  6.782e-01 0.037840
#>  [46,] 28  6.868e-01 0.036120
#>  [47,] 29  6.958e-01 0.034480
#>  [48,] 30  7.047e-01 0.032910
#>  [49,] 31  7.134e-01 0.031410
#>  [50,] 32  7.220e-01 0.029990
```

```
#>  [51,] 35  7.308e-01 0.028620
#>  [52,] 35  7.396e-01 0.027320
#>  [53,] 37  7.483e-01 0.026080
#>  [54,] 37  7.569e-01 0.024890
#>  [55,] 38  7.652e-01 0.023760
#>  [56,] 39  7.733e-01 0.022680
#>  [57,] 39  7.812e-01 0.021650
#>  [58,] 41  7.888e-01 0.020670
#>  [59,] 42  7.962e-01 0.019730
#>  [60,] 43  8.035e-01 0.018830
#>  [61,] 45  8.107e-01 0.017980
#>  [62,] 47  8.177e-01 0.017160
#>  [63,] 50  8.246e-01 0.016380
#>  [64,] 51  8.315e-01 0.015630
#>  [65,] 52  8.382e-01 0.014920
#>  [66,] 53  8.447e-01 0.014250
#>  [67,] 52  8.510e-01 0.013600
#>  [68,] 54  8.570e-01 0.012980
#>  [69,] 55  8.629e-01 0.012390
#>  [70,] 57  8.686e-01 0.011830
#>  [71,] 60  8.741e-01 0.011290
#>  [72,] 62  8.796e-01 0.010780
#>  [73,] 62  8.848e-01 0.010290
#>  [74,] 62  8.899e-01 0.009819
#>  [75,] 63  8.947e-01 0.009373
#>  [76,] 63  8.993e-01 0.008947
#>  [77,] 66  9.037e-01 0.008540
#>  [78,] 67  9.081e-01 0.008152
#>  [79,] 69  9.123e-01 0.007781
#>  [80,] 70  9.164e-01 0.007428
#>  [81,] 69  9.202e-01 0.007090
#>  [82,] 70  9.238e-01 0.006768
#>  [83,] 71  9.273e-01 0.006460
#>  [84,] 72  9.306e-01 0.006167
#>  [85,] 74  9.337e-01 0.005886
#>  [86,] 76  9.368e-01 0.005619
#>  [87,] 76  9.397e-01 0.005363
#>  [88,] 77  9.424e-01 0.005120
#>  [89,] 78  9.450e-01 0.004887
#>  [90,] 79  9.476e-01 0.004665
#>  [91,] 78  9.500e-01 0.004453
#>  [92,] 79  9.523e-01 0.004250
#>  [93,] 78  9.544e-01 0.004057
#>  [94,] 78  9.565e-01 0.003873
#>  [95,] 78  9.585e-01 0.003697
#>  [96,] 78  9.604e-01 0.003529
#>  [97,] 80  9.622e-01 0.003368
#>  [98,] 81  9.639e-01 0.003215
#>  [99,] 80  9.656e-01 0.003069
#> [100,] 80  9.671e-01 0.002930
#> [1] "done bootstrap 2"

sink()
```

4.2 Display summary results for the prediction

```
#get the number of rows for the summary matrix
row_cluster <-length(unique(colData(mixedpop2)[,1]))

#summary results LDA to to show the percent of cells classified as cells belonging by LDA classifier
summary_prediction_lda(LSOLDA_dat=LSOLDA_dat, nPredSubpop = row_cluster )
#>                  V1               V2                                    names
#> 1        66.796875        54.296875 LDA for subpop 1 in target mixedpop2
#> 2 38.6046511627907               40 LDA for subpop 2 in target mixedpop2
#> 3 37.9310344827586 37.9310344827586 LDA for subpop 3 in target mixedpop2

#summary results Lasso to show the percent of cells classified as cells belonging by Lasso classifier
summary_prediction_lasso(LSOLDA_dat=LSOLDA_dat, nPredSubpop = row_cluster)
#>                  V1               V2
#> 1        30.859375        62.109375
#> 2 46.9767441860465 27.4418604651163
#> 3 55.1724137931034 20.6896551724138
#>                                       names
#> 1 ElasticNet for subpop1 in target mixedpop2
#> 2 ElasticNet for subpop2 in target mixedpop2
#> 3 ElasticNet for subpop3 in target mixedpop2

# summary maximum deviance explained by the model during the model training
summary_deviance(object = LSOLDA_dat)
#> $allDeviance
#> [1] "0.6337" "0.7733"
#>
#> $DeviMax
#>          Dfd   Deviance        DEgenes
#> 1          0 -2.563e-15 genes_cluster1
#> 2          1    0.02202 genes_cluster1
#> 3          2    0.07307 genes_cluster1
#> 4          3     0.1249 genes_cluster1
#> 5          4     0.1734 genes_cluster1
#> 6          5      0.196 genes_cluster1
#> 7          6     0.2177 genes_cluster1
#> 8          7     0.2774 genes_cluster1
#> 9          8     0.3766 genes_cluster1
#> 10         9     0.3911 genes_cluster1
#> 11        10     0.4317 genes_cluster1
#> 12        11      0.457 genes_cluster1
#> 13        16     0.4863 genes_cluster1
#> 14        17     0.5493 genes_cluster1
#> 15        19     0.5826 genes_cluster1
#> 16        20      0.593 genes_cluster1
#> 17        21     0.6227 genes_cluster1
#> 18        22     0.6515 genes_cluster1
#> 19        24     0.6695 genes_cluster1
#> 20        26     0.6782 genes_cluster1
#> 21        28     0.6868 genes_cluster1
#> 22        29     0.6958 genes_cluster1
#> 23        30     0.7047 genes_cluster1
```

```
#> 24         31     0.7134 genes_cluster1
#> 25         32      0.722 genes_cluster1
#> 26         35     0.7396 genes_cluster1
#> 27         37     0.7569 genes_cluster1
#> 28         38     0.7652 genes_cluster1
#> 29         39     0.7733 genes_cluster1
#> 30 remaining          1        DEgenes
#>
#> $LassoGenesMax
#> NULL

# summary accuracy to check the model accuracy in the leave-out test set
summary_accuracy(object = LSOLDA_dat)
#> [1] 90.17857 89.28571
```

4.3 Plot the relationship between clusters

Here we look at one example use case to find relationship between clusters within one sample or between two sample

```
#run prediction for 3 clusters
cluster_mixedpop1 <- colData(mixedpop1)[,1]
cluster_mixedpop2 <- as.numeric(as.vector(colData(mixedpop2)[,1]))

c_selectID <- 1
genes = DEgenes$DE_Subpop1vsRemaining$id[1:200] #top 200 gene markers distinguishing cluster 1
genes <- gsub("_.*", "", genes)

LSOLDA_dat1 <- bootstrap_scGPS(nboots = 1, mixedpop1 = mixedpop1, mixedpop2 = mixedpop2, genes=genes, c
#>
#> Call:  glmnet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>       Df       %Dev    Lambda
#>  [1,]  0 -2.563e-15 2.897e-01
#>  [2,]  3  5.306e-02 2.640e-01
#>  [3,]  3  1.073e-01 2.405e-01
#>  [4,]  3  1.537e-01 2.191e-01
#>  [5,]  3  1.939e-01 1.997e-01
#>  [6,]  3  2.289e-01 1.819e-01
#>  [7,]  4  2.615e-01 1.658e-01
#>  [8,]  4  2.915e-01 1.510e-01
#>  [9,]  4  3.182e-01 1.376e-01
#> [10,]  4  3.420e-01 1.254e-01
#> [11,]  5  3.634e-01 1.143e-01
#> [12,]  5  3.842e-01 1.041e-01
#> [13,]  5  4.027e-01 9.486e-02
#> [14,]  7  4.212e-01 8.643e-02
#> [15,]  8  4.391e-01 7.875e-02
#> [16,] 10  4.571e-01 7.176e-02
#> [17,] 13  4.778e-01 6.538e-02
#> [18,] 14  5.008e-01 5.958e-02
#> [19,] 15  5.220e-01 5.428e-02
#> [20,] 15  5.414e-01 4.946e-02
```

```
#> [21,] 19  5.609e-01 4.507e-02
#> [22,] 22  5.803e-01 4.106e-02
#> [23,] 25  6.013e-01 3.741e-02
#> [24,] 26  6.229e-01 3.409e-02
#> [25,] 27  6.431e-01 3.106e-02
#> [26,] 30  6.630e-01 2.830e-02
#> [27,] 31  6.819e-01 2.579e-02
#> [28,] 34  6.995e-01 2.350e-02
#> [29,] 36  7.161e-01 2.141e-02
#> [30,] 36  7.313e-01 1.951e-02
#> [31,] 39  7.460e-01 1.778e-02
#> [32,] 40  7.601e-01 1.620e-02
#> [33,] 41  7.738e-01 1.476e-02
#> [34,] 43  7.871e-01 1.345e-02
#> [35,] 46  8.002e-01 1.225e-02
#> [36,] 51  8.135e-01 1.116e-02
#> [37,] 54  8.273e-01 1.017e-02
#> [38,] 55  8.412e-01 9.268e-03
#> [39,] 55  8.540e-01 8.445e-03
#> [40,] 55  8.658e-01 7.694e-03
#> [41,] 56  8.768e-01 7.011e-03
#> [42,] 57  8.871e-01 6.388e-03
#> [43,] 59  8.967e-01 5.821e-03
#> [44,] 60  9.056e-01 5.303e-03
#> [45,] 59  9.137e-01 4.832e-03
#> [46,] 60  9.210e-01 4.403e-03
#> [47,] 60  9.279e-01 4.012e-03
#> [48,] 60  9.341e-01 3.655e-03
#> [49,] 60  9.397e-01 3.331e-03
#> [50,] 60  9.449e-01 3.035e-03
#> [51,] 61  9.498e-01 2.765e-03
#> [52,] 62  9.541e-01 2.520e-03
#> [53,] 63  9.581e-01 2.296e-03
#> [54,] 63  9.617e-01 2.092e-03
#> [55,] 66  9.651e-01 1.906e-03
#> [56,] 68  9.682e-01 1.737e-03
#> [57,] 68  9.710e-01 1.582e-03
#> [58,] 69  9.736e-01 1.442e-03
#> [59,] 69  9.759e-01 1.314e-03
#> [60,] 69  9.781e-01 1.197e-03
#> [61,] 69  9.800e-01 1.091e-03
#> [62,] 69  9.818e-01 9.938e-04
#> [63,] 69  9.834e-01 9.055e-04
#> [64,] 71  9.849e-01 8.250e-04
#> [65,] 71  9.862e-01 7.518e-04
#> [66,] 71  9.874e-01 6.850e-04
#> [67,] 71  9.885e-01 6.241e-04
#> [68,] 71  9.896e-01 5.687e-04
#> [69,] 71  9.905e-01 5.182e-04
#> [70,] 71  9.913e-01 4.721e-04
#> [71,] 73  9.921e-01 4.302e-04
#> [72,] 73  9.928e-01 3.920e-04
#> [73,] 73  9.934e-01 3.571e-04
```

```
#> [74,] 73  9.940e-01 3.254e-04
#> [75,] 73  9.945e-01 2.965e-04
#> [76,] 73  9.950e-01 2.702e-04
#> [77,] 73  9.955e-01 2.462e-04
#> [78,] 74  9.959e-01 2.243e-04
#> [79,] 75  9.962e-01 2.044e-04
#> [80,] 75  9.966e-01 1.862e-04
#> [81,] 75  9.969e-01 1.697e-04
#> [82,] 75  9.971e-01 1.546e-04
#> [83,] 75  9.974e-01 1.409e-04
#> [84,] 75  9.976e-01 1.284e-04
#> [85,] 75  9.978e-01 1.169e-04
#> [86,] 75  9.980e-01 1.066e-04
#> [87,] 75  9.982e-01 9.709e-05
#> [88,] 75  9.984e-01 8.847e-05
#> [89,] 74  9.985e-01 8.061e-05
#> [90,] 74  9.986e-01 7.345e-05
#> [91,] 74  9.988e-01 6.692e-05
#> [92,] 73  9.989e-01 6.098e-05
#> [93,] 73  9.990e-01 5.556e-05
#> [94,] 73  9.991e-01 5.062e-05
#> [1] "done bootstrap 1"


c_selectID <- 2
genes = DEgenes$DE_Subpop2vsRemaining$id[1:200]
genes <- gsub("_.*", "", genes)
LSOLDA_dat2 <- bootstrap_scGPS(nboots = 1,mixedpop1 = mixedpop1, mixedpop2 = mixedpop2, genes=genes, c_s
     cluster_mixedpop2 = cluster_mixedpop2)
#>
#> Call:  glmnet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>       Df      %Dev    Lambda
#>  [1,]  0 1.281e-15 3.350e-01
#>  [2,]  1 5.518e-02 3.053e-01
#>  [3,]  1 1.019e-01 2.782e-01
#>  [4,]  1 1.420e-01 2.534e-01
#>  [5,]  1 1.769e-01 2.309e-01
#>  [6,]  1 2.075e-01 2.104e-01
#>  [7,]  3 2.413e-01 1.917e-01
#>  [8,]  4 2.772e-01 1.747e-01
#>  [9,]  4 3.107e-01 1.592e-01
#> [10,]  4 3.407e-01 1.450e-01
#> [11,]  4 3.675e-01 1.321e-01
#> [12,]  6 3.924e-01 1.204e-01
#> [13,]  6 4.168e-01 1.097e-01
#> [14,]  7 4.389e-01 9.996e-02
#> [15,]  7 4.619e-01 9.108e-02
#> [16,]  9 4.846e-01 8.299e-02
#> [17,]  9 5.062e-01 7.562e-02
#> [18,]  9 5.259e-01 6.890e-02
#> [19,] 10 5.440e-01 6.278e-02
#> [20,] 13 5.622e-01 5.720e-02
```

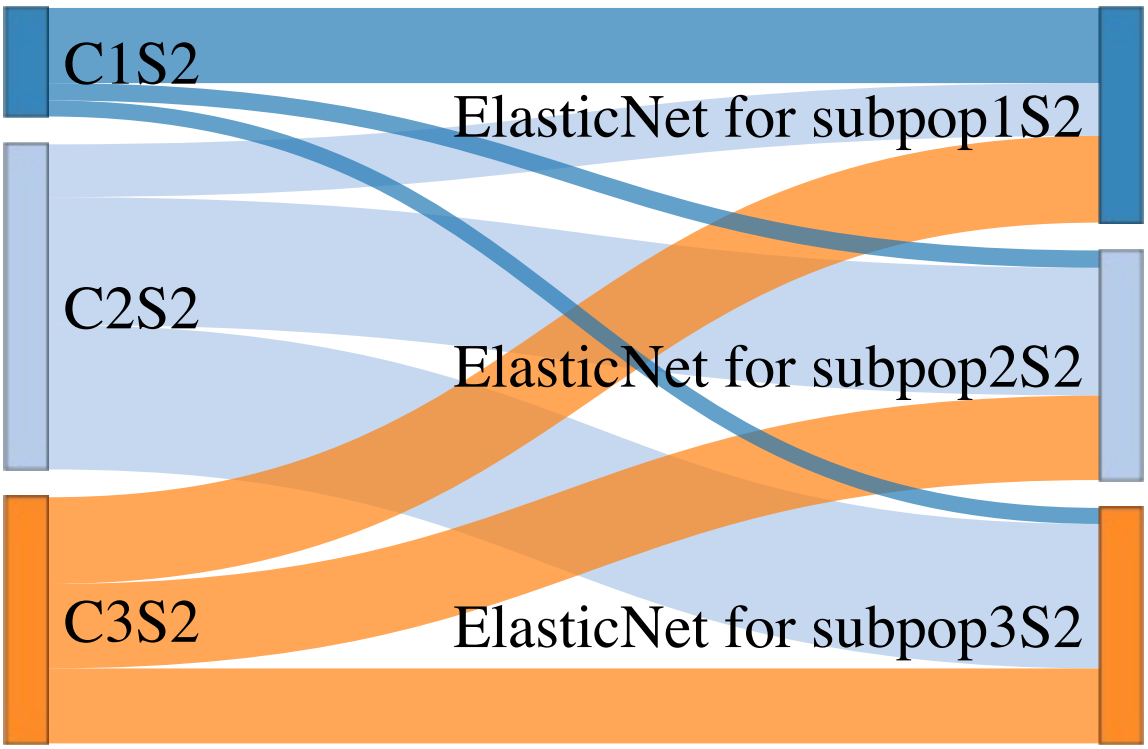
```
#> [21,] 13 5.802e-01 5.212e-02
#> [22,] 16 5.992e-01 4.749e-02
#> [23,] 17 6.183e-01 4.327e-02
#> [24,] 18 6.361e-01 3.943e-02
#> [25,] 19 6.534e-01 3.592e-02
#> [26,] 23 6.711e-01 3.273e-02
#> [27,] 25 6.889e-01 2.983e-02
#> [28,] 25 7.054e-01 2.718e-02
#> [29,] 28 7.215e-01 2.476e-02
#> [30,] 30 7.380e-01 2.256e-02
#> [31,] 31 7.534e-01 2.056e-02
#> [32,] 34 7.685e-01 1.873e-02
#> [33,] 35 7.828e-01 1.707e-02
#> [34,] 35 7.961e-01 1.555e-02
#> [35,] 36 8.085e-01 1.417e-02
#> [36,] 36 8.205e-01 1.291e-02
#> [37,] 40 8.321e-01 1.176e-02
#> [38,] 43 8.438e-01 1.072e-02
#> [39,] 46 8.564e-01 9.766e-03
#> [40,] 45 8.683e-01 8.899e-03
#> [41,] 47 8.791e-01 8.108e-03
#> [42,] 49 8.892e-01 7.388e-03
#> [43,] 49 8.985e-01 6.732e-03
#> [44,] 48 9.070e-01 6.134e-03
#> [45,] 48 9.147e-01 5.589e-03
#> [46,] 51 9.218e-01 5.092e-03
#> [47,] 51 9.284e-01 4.640e-03
#> [48,] 52 9.345e-01 4.228e-03
#> [49,] 54 9.401e-01 3.852e-03
#> [50,] 54 9.452e-01 3.510e-03
#> [51,] 53 9.499e-01 3.198e-03
#> [52,] 53 9.542e-01 2.914e-03
#> [53,] 55 9.582e-01 2.655e-03
#> [54,] 55 9.618e-01 2.419e-03
#> [55,] 56 9.652e-01 2.204e-03
#> [56,] 57 9.682e-01 2.008e-03
#> [57,] 57 9.710e-01 1.830e-03
#> [58,] 58 9.735e-01 1.667e-03
#> [59,] 59 9.759e-01 1.519e-03
#> [60,] 59 9.780e-01 1.384e-03
#> [61,] 59 9.799e-01 1.261e-03
#> [62,] 58 9.817e-01 1.149e-03
#> [63,] 58 9.833e-01 1.047e-03
#> [64,] 58 9.848e-01 9.542e-04
#> [65,] 58 9.861e-01 8.694e-04
#> [66,] 57 9.873e-01 7.922e-04
#> [67,] 58 9.884e-01 7.218e-04
#> [68,] 58 9.894e-01 6.577e-04
#> [69,] 58 9.904e-01 5.993e-04
#> [70,] 59 9.912e-01 5.460e-04
#> [71,] 60 9.920e-01 4.975e-04
#> [72,] 60 9.927e-01 4.533e-04
#> [73,] 61 9.933e-01 4.130e-04
```

```
#> [74,] 61 9.939e-01 3.764e-04
#> [75,] 61 9.945e-01 3.429e-04
#> [76,] 62 9.949e-01 3.125e-04
#> [77,] 61 9.954e-01 2.847e-04
#> [78,] 61 9.958e-01 2.594e-04
#> [79,] 62 9.962e-01 2.364e-04
#> [80,] 62 9.965e-01 2.154e-04
#> [81,] 62 9.968e-01 1.962e-04
#> [82,] 62 9.971e-01 1.788e-04
#> [83,] 62 9.974e-01 1.629e-04
#> [84,] 60 9.976e-01 1.484e-04
#> [85,] 60 9.978e-01 1.353e-04
#> [86,] 61 9.980e-01 1.232e-04
#> [87,] 61 9.982e-01 1.123e-04
#> [88,] 61 9.983e-01 1.023e-04
#> [89,] 61 9.985e-01 9.323e-05
#> [90,] 61 9.986e-01 8.494e-05
#> [91,] 62 9.987e-01 7.740e-05
#> [92,] 62 9.988e-01 7.052e-05
#> [93,] 62 9.989e-01 6.426e-05
#> [94,] 62 9.990e-01 5.855e-05
#> [1] "done bootstrap 1"

c_selectID <- 3
genes = DEgenes$DE_Subpop3vsRemaining$id[1:200]
genes <- gsub("_.*", "", genes)
LSOLDA_dat3 <- bootstrap_scGPS(nboots = 1,mixedpop1 = mixedpop1, mixedpop2 = mixedpop2, genes=genes, c_
     cluster_mixedpop2 = cluster_mixedpop2)
#>
#> Call:  glmmet(x = as.matrix(dataset[, -which(colnames(dataset) == "Cluster_class")]),      y = y_cat
#>
#>        Df      %Dev   Lambda
#>   [1,]  0 2.403e-15 0.156200
#>   [2,]  1 6.261e-03 0.149100
#>   [3,]  1 1.199e-02 0.142300
#>   [4,]  1 1.725e-02 0.135900
#>   [5,]  1 2.210e-02 0.129700
#>   [6,]  1 2.659e-02 0.123800
#>   [7,]  1 3.074e-02 0.118200
#>   [8,]  2 3.657e-02 0.112800
#>   [9,]  2 4.337e-02 0.107700
#>  [10,]  2 4.969e-02 0.102800
#>  [11,]  5 5.917e-02 0.098110
#>  [12,]  7 7.316e-02 0.093650
#>  [13,] 10 8.902e-02 0.089400
#>  [14,] 10 1.072e-01 0.085330
#>  [15,] 11 1.248e-01 0.081460
#>  [16,] 13 1.421e-01 0.077750
#>  [17,] 15 1.620e-01 0.074220
#>  [18,] 17 1.811e-01 0.070850
#>  [19,] 19 2.027e-01 0.067630
#>  [20,] 21 2.249e-01 0.064550
#>  [21,] 22 2.462e-01 0.061620
```

```
#>  [22,] 22 2.666e-01 0.058820
#>  [23,] 24 2.862e-01 0.056140
#>  [24,] 25 3.056e-01 0.053590
#>  [25,] 25 3.242e-01 0.051160
#>  [26,] 25 3.418e-01 0.048830
#>  [27,] 28 3.602e-01 0.046610
#>  [28,] 29 3.784e-01 0.044490
#>  [29,] 31 3.960e-01 0.042470
#>  [30,] 33 4.143e-01 0.040540
#>  [31,] 33 4.318e-01 0.038700
#>  [32,] 35 4.487e-01 0.036940
#>  [33,] 36 4.652e-01 0.035260
#>  [34,] 37 4.819e-01 0.033660
#>  [35,] 40 4.983e-01 0.032130
#>  [36,] 42 5.149e-01 0.030670
#>  [37,] 42 5.310e-01 0.029270
#>  [38,] 44 5.465e-01 0.027940
#>  [39,] 48 5.618e-01 0.026670
#>  [40,] 48 5.769e-01 0.025460
#>  [41,] 49 5.914e-01 0.024300
#>  [42,] 49 6.053e-01 0.023200
#>  [43,] 50 6.188e-01 0.022140
#>  [44,] 52 6.320e-01 0.021140
#>  [45,] 52 6.448e-01 0.020180
#>  [46,] 52 6.571e-01 0.019260
#>  [47,] 53 6.687e-01 0.018380
#>  [48,] 54 6.802e-01 0.017550
#>  [49,] 55 6.911e-01 0.016750
#>  [50,] 56 7.018e-01 0.015990
#>  [51,] 57 7.122e-01 0.015260
#>  [52,] 58 7.225e-01 0.014570
#>  [53,] 57 7.326e-01 0.013910
#>  [54,] 57 7.424e-01 0.013280
#>  [55,] 57 7.517e-01 0.012670
#>  [56,] 56 7.608e-01 0.012100
#>  [57,] 57 7.695e-01 0.011550
#>  [58,] 61 7.781e-01 0.011020
#>  [59,] 60 7.864e-01 0.010520
#>  [60,] 60 7.945e-01 0.010040
#>  [61,] 62 8.024e-01 0.009586
#>  [62,] 63 8.101e-01 0.009150
#>  [63,] 64 8.176e-01 0.008734
#>  [64,] 65 8.248e-01 0.008337
#>  [65,] 67 8.319e-01 0.007958
#>  [66,] 68 8.386e-01 0.007597
#>  [67,] 70 8.451e-01 0.007251
#>  [68,] 70 8.514e-01 0.006922
#>  [69,] 68 8.576e-01 0.006607
#>  [70,] 70 8.634e-01 0.006307
#>  [71,] 73 8.690e-01 0.006020
#>  [72,] 75 8.747e-01 0.005747
#>  [73,] 75 8.800e-01 0.005485
#>  [74,] 75 8.852e-01 0.005236
```

```
#>  [75,] 75 8.901e-01 0.004998
#>  [76,] 77 8.949e-01 0.004771
#>  [77,] 78 8.995e-01 0.004554
#>  [78,] 78 9.039e-01 0.004347
#>  [79,] 79 9.080e-01 0.004149
#>  [80,] 80 9.121e-01 0.003961
#>  [81,] 80 9.160e-01 0.003781
#>  [82,] 80 9.197e-01 0.003609
#>  [83,] 81 9.233e-01 0.003445
#>  [84,] 83 9.267e-01 0.003288
#>  [85,] 83 9.300e-01 0.003139
#>  [86,] 84 9.332e-01 0.002996
#>  [87,] 85 9.361e-01 0.002860
#>  [88,] 83 9.390e-01 0.002730
#>  [89,] 84 9.417e-01 0.002606
#>  [90,] 84 9.443e-01 0.002488
#>  [91,] 86 9.468e-01 0.002374
#>  [92,] 86 9.493e-01 0.002267
#>  [93,] 88 9.515e-01 0.002164
#>  [94,] 89 9.537e-01 0.002065
#>  [95,] 88 9.559e-01 0.001971
#>  [96,] 89 9.579e-01 0.001882
#>  [97,] 89 9.598e-01 0.001796
#>  [98,] 89 9.617e-01 0.001715
#>  [99,] 89 9.634e-01 0.001637
#> [100,] 88 9.651e-01 0.001562
#> [1] "done bootstrap 1"

#prepare table input for sankey plot

reformat_LASSO <-function(c_selectID = NULL, s_selectID = NULL, LSOLDA_dat = NULL,
                          nPredSubpop = row_cluster, Nodes_group = "#7570b3"){
  LASSO_out <- summary_prediction_lasso(LSOLDA_dat=LSOLDA_dat, nPredSubpop = nPredSubpop)
  LASSO_out <-as.data.frame(LASSO_out)
  temp_name <- gsub("LASSO for subpop", "C", LASSO_out$names)
  temp_name <- gsub(" in target mixedpop", "S", temp_name)
  LASSO_out$names <-temp_name
  source <-rep(paste0("C",c_selectID,"S",s_selectID), length(temp_name))
  LASSO_out$Source <- source
  LASSO_out$Node <- source
  LASSO_out$Nodes_group <- rep(Nodes_group, length(temp_name))
  colnames(LASSO_out) <-c("Value", "Target", "Source", "Node", "NodeGroup")
  LASSO_out$Value <- as.numeric(as.vector(LASSO_out$Value))
  return(LASSO_out)
}

LASSO_C1S2  <- reformat_LASSO(c_selectID=1, s_selectID =2, LSOLDA_dat=LSOLDA_dat1,
                          nPredSubpop = row_cluster, Nodes_group = "#7570b3")

LASSO_C2S2  <- reformat_LASSO(c_selectID=2, s_selectID =2, LSOLDA_dat=LSOLDA_dat2,
                          nPredSubpop = row_cluster, Nodes_group = "#1b9e77")

LASSO_C3S2  <- reformat_LASSO(c_selectID=3, s_selectID =2, LSOLDA_dat=LSOLDA_dat3,
```

```
                          nPredSubpop = row_cluster, Nodes_group = "#e7298a")


combined <- rbind(LASSO_C1S2,LASSO_C2S2,LASSO_C3S2 )
combined <- combined[is.na(combined$Value) != TRUE,]
combined_D3obj <-list(Nodes=combined[,4:5], Links=combined[,c(3,2,1)])

library(networkD3)

Node_source <- as.vector(sort(unique(combined_D3obj$Links$Source)))
Node_target <- as.vector(sort(unique(combined_D3obj$Links$Target)))
Node_all <-unique(c(Node_source, Node_target))

#assign IDs for Source (start from 0)
Source <-combined_D3obj$Links$Source
Target <- combined_D3obj$Links$Target

for(i in 1:length(Node_all)){
  Source[Source==Node_all[i]] <-i-1
  Target[Target==Node_all[i]] <-i-1
}

combined_D3obj$Links$Source <- as.numeric(Source)
combined_D3obj$Links$Target <- as.numeric(Target)
combined_D3obj$Links$LinkColor <- combined$NodeGroup

#prepare node info
node_df <-data.frame(Node=Node_all)
node_df$id <-as.numeric(c(0, 1:(length(Node_all)-1)))

suppressMessages(library(dplyr))
Color <- combined %>% count(Node, color=NodeGroup) %>% select(2)
node_df$color <- Color$color

suppressMessages(library(networkD3))
p1<-sankeyNetwork(Links =combined_D3obj$Links, Nodes = node_df,  Value = "Value", NodeGroup ="color", Li
                  fontSize = 22 )
p1
```




```
#saveNetwork(p1, file = paste0(path,'Subpopulation_Net.html'))
##R Setting Information
#sessionInfo()
#rmarkdown::render("/Users/quan.nguyen/Documents/Powell_group_MacQuan/AllCodes/scGPS/vignettes/vignette
#rmarkdown::render("/Users/quan.nguyen/Documents/Powell_group_MacQuan/AllCodes/scGPS/vignettes/vignette
```