Simulation sceanrio 1

Yunshan Duan

2024-03-14

Load functions:

```r
## set wd
library(ggplot2); theme_set(theme_bw())
library(dplyr)
library(Seurat)
library(mclust)
library(MCMCpack)
library(mvtnorm)
library(salso)

source("./SASC_func.R")
source("./comparison_func.R")

data_dir <- "../data/simulation_sc1"
output_dir <- "../output/simulation_sc1"
output_data_dir <- "../output/simulation_sc1/data"
output_figure_dir <- "../output/simulation_sc1/figures"

# colors
pal <- c("#ffb6db", "#33A02C", "#b66dff", "#FEC44F", "#41B6C4", "#8E0152", "#0868AC", "#807DBA", "#E7298
         "#00441B", "#525252", "#4D9221", "#8B5742", "#D8DAEB", "#7cdd2d", "#980043", "#8C96C6", "#EC70
         "#FDAE61", "#1D91C0", "#A6DBA0", "#4292C6", "#BF812D", "#01665E", "#41AB5D", "#FE9929", "#25252
names(pal) <- 1:30

set.seed(1999)
```

Load data set:

```r
# load simulated data
sim_data <- readRDS(paste0(data_dir, "/sim_data.rds"))

count <- as.matrix(sim_data@assays$RNA@counts)
gene <- sim_data@reductions$pca@cell.embeddings
gene_umap <- sim_data@reductions$umap@cell.embeddings
condition <- sim_data@meta.data$condition_sim
cell_names <- colnames(count)
celltype_sim <- factor(as.numeric(sim_data@meta.data$celltype_sim))
celltype_type <- sim_data@meta.data$celltype_CLE
celltype_type <- factor(celltype_type, levels = c("C", "A", "B"))

method <- "Truth"
```

```r
plotcluster_func(gene_umap, celltype_sim, method,
                 filename=paste0(output_figure_dir, "/cluster_", method, ".pdf"))
```

Seurat

```r
method <- "Seurat"
### Seurat clustering
t0 <- Sys.time()
sim_data_seurat <- sim_data
sim_data_seurat <- FindNeighbors(sim_data_seurat, dims = 1:10)
sim_data_seurat <- FindClusters(object = sim_data_seurat)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 1601
## Number of edges: 58036
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.7187
## Number of communities: 7
## Elapsed time: 0 seconds
```

```r
cluster <- as.numeric(sim_data_seurat@meta.data$seurat_clusters)
cluster <- factor(cluster, levels=as.character(1:max(cluster)))
runtime <- as.numeric(difftime(Sys.time(), t0, units = "secs"))

### cluster plot
plotcluster_func(gene_umap, cluster, method,
                 filename=paste0(output_figure_dir, "/cluster_", method, ".pdf"))

### cluster weight, type
cluster_num <- cluster_num_func(cluster, condition)

cluster_type <- cluster_type_func(cluster, cluster_num, wdiff_tol)

#### acc
acc_list <- acc_func(predicted=cluster_type, truth=celltype_type)
cat("\n Balanced accuracy: \n", round(acc_list$balanced_acc, 2))
```

```
##
##  Balanced accuracy:
##  1 0.96 0.95
```

```r
# store outputs
results_seurat <- list(cluster = cluster, cluster_type = cluster_type,
                       acc = acc_list$acc,
                       balanced_acc = acc_list$balanced_acc,
                       runtime = runtime)
```

SASC

```r
method <- "SASC"
if (file.exists(paste0(output_data_dir, "/results_SASC.rds"))) {
  results_SASC <- readRDS(paste0(output_data_dir, "/results_SASC.rds"))
```

```r
} else {
  #### DMM clustering
  LOCRC_cells <- cell_names[which(condition == "B")]
  y0 <- as.matrix(gene[LOCRC_cells, ])
  YOCRC_cells <- cell_names[which(condition == "A")]
  y1 <- as.matrix(gene[YOCRC_cells, ])
  dim(y0)

  dim(y1)

  n_feat <- dim(y0)[2]
  ncells <- dim(gene)[1]

  H <- 2
  seed <- 1999
  beta0_low <- 1
  beta0_high <- ncells/H/3/2

  t0 <- Sys.time()
  out <- func888(features = gene, y0 = y0, y1 = y1, H = H, seed = seed,
                 cell_names = cell_names, LOCRC_cells = LOCRC_cells, YOCRC_cells = YOCRC_cells,
                 beta0_low = beta0_low, beta0_high = beta0_high, iden_eps = TRUE,
                 niter = 5000, niter_extra = 1000)

  z0_post <- out$z0_post
  z1_post <- out$z1_post
  weights <- out$weights
  weights_mat <- out$weights_mat
  cluster_type_CLE <- out$cluster_type_CLE

  cluster <- rep(NA, nrow(gene))
  names(cluster) <- rownames(gene)
  cluster[LOCRC_cells] <- as.character(z0_post)
  cluster[YOCRC_cells] <- as.character(z1_post)
  cluster <- as.numeric(cluster)
  cluster <- factor(cluster, levels=as.character(1:max(cluster)))
  runtime <- as.numeric(difftime(Sys.time(), t0, units = "secs"))

  ####

  ### cluster plot
  plotcluster_func(gene_umap, cluster, method,
                   filename=paste0(output_figure_dir, "/cluster_", method, ".pdf"))

  # ### cluster type
  Cind <- which(cluster_type_CLE == "C")
  Bind <- which(cluster_type_CLE == "L")
  Aind <- which(cluster_type_CLE == "E")

  cluster_type <- rep(NA, length(cluster))
  cluster_type[which(cluster %in% Cind)] <- "C"
  cluster_type[which(cluster %in% Bind)] <- "B"
  cluster_type[which(cluster %in% Aind)] <- "A"
  cluster_type <- factor(cluster_type, levels = c("C", "A", "B"))

  #### acc
  acc_list <- acc_func(predicted=cluster_type, truth=celltype_type)
  cat("\n Balanced accuracy: \n", round(acc_list$balanced_acc, 2))

  # store outputs
  results_SASC <- list(cluster = cluster, cluster_type = cluster_type,
                       acc = acc_list$acc,
                       balanced_acc = acc_list$balanced_acc,
                       runtime = runtime)
  saveRDS(results_SASC, file = paste0(output_data_dir, "/results_SASC.rds"))
}
```

ZINB-WaVE

```
method <- "ZINB-WaVE"

if (file.exists(paste0(output_data_dir, "/results_zinb.rds"))) {
  results_zinb <- readRDS(paste0(output_data_dir, "/results_zinb.rds"))
} else {
  library("zinbwave")
  library("SummarizedExperiment")

  # construct SummarizedExperiment object
  t0 <- Sys.time()
  colData <- DataFrame(Condition=condition,
                       row.names=colnames(count))

  simdata_zinb <- SummarizedExperiment(assays=list(counts=count),
                                       colData=colData)
  # we filter out the lowly expressed genes,
  # by removing those genes that do not have at least 5 reads in at least 5 samples
  filter <- rowSums(assay(simdata_zinb)>5)>5
  table(filter)
  simdata_zinb <- simdata_zinb[filter,]
  # We next identify the 100 most variable genes
  assay(simdata_zinb) %>% log1p %>% rowVars -> vars
  names(vars) <- rownames(simdata_zinb)
  vars <- sort(vars, decreasing = TRUE)
  head(vars)
  simdata_zinb <- simdata_zinb[names(vars)[1:100],]
  assayNames(simdata_zinb)[1] <- "counts"

  # run ZINB-WaVE

  zinb_out <- zinbwave(simdata_zinb, K = 10, X="~Condition", epsilon=1000)

  W_zinb <- reducedDim(zinb_out)

  zinb_out_seurat <- as.Seurat(x = zinb_out, counts = "counts", data = "counts")

  zinb_out_seurat <- FindNeighbors(zinb_out_seurat, reduction = "zinbwave",
                                    dims = 1:10 #this should match K
  )
  zinb_out_seurat <- FindClusters(object = zinb_out_seurat)

  cluster <- as.numeric(zinb_out_seurat@meta.data$seurat_clusters)
  cluster <- factor(cluster, levels=as.character(1:max(cluster)))
  runtime <- as.numeric(difftime(Sys.time(), t0, units = "secs"))

  ### cluster plot
  plotcluster_func(gene_umap, cluster, method,
                   filename=paste0(output_figure_dir, "/cluster_", method, ".pdf"))

  ### cluster weight, type
  cluster_num <- cluster_num_func(cluster, condition)

  cluster_type <- cluster_type_func(cluster, cluster_num, wdiff_tol)

  #### acc
  acc_list <- acc_func(predicted=cluster_type, truth=celltype_type)
  cat("\n Balanced accuracy: \n", round(acc_list$balanced_acc, 2))

  # store outputs
  results_zinb <- list(cluster = cluster, cluster_type = cluster_type,
                       acc = acc_list$acc,
                       balanced_acc = acc_list$balanced_acc,
                       runtime = runtime)
  saveRDS(results_zinb, file = paste0(output_data_dir, "/results_zinb.rds"))
}
```

GMM

```
method <- "GMM"

library(mclust)

### GMM clustering
t0 <- Sys.time()
fit = Mclust(gene, G=6)

cluster <- fit$classification
cluster <- factor(cluster, levels=as.character(1:max(cluster)))
runtime <- as.numeric(difftime(Sys.time(), t0, units = "secs"))

### cluster plot
plotcluster_func(gene_umap, cluster, method,
                 filename=paste0(output_figure_dir, "/cluster_", method, ".pdf"))

### cluster weight, type
cluster_num <- cluster_num_func(cluster, condition)

cluster_type <- cluster_type_func(cluster, cluster_num, wdiff_tol)

#### acc
```

```
acc_list <- acc_func(predicted=cluster_type, truth=celltype_type)
cat("\n Balanced accuracy: \n", round(acc_list$balanced_acc, 2))
```

```
##
##  Balanced accuracy:
##  0.96 0.83 0.84
```

```
# store outputs
results_GMM <- list(cluster = cluster, cluster_type = cluster_type,
                    acc = acc_list$acc,
                    balanced_acc = acc_list$balanced_acc,
                    runtime = runtime)
```

Summary of results

```
### all clustering accuracy
b_acc_all <- rbind(results_SASC$balanced_acc,
                   results_GMM$balanced_acc,
                   results_seurat$balanced_acc,
                   results_zinb$balanced_acc)
rownames(b_acc_all) <- c("SASC", "GMM", "Seurat", "ZINB-WaVE")

cat("\n Balanced accuracy: \n")
```

```
##
##  Balanced accuracy:
```

```
round(b_acc_all, 2)
```

```
##              C    A    B
## SASC      0.95 0.98 0.92
## GMM       0.96 0.83 0.84
## Seurat    1.00 0.96 0.95
## ZINB-WaVE 0.24 0.54 0.47
```

```
################ ARI #####################

ari <- c(
  adjustedRandIndex(celltype_sim, results_SASC$cluster),
  adjustedRandIndex(celltype_sim, results_GMM$cluster),
  adjustedRandIndex(celltype_sim, results_seurat$cluster),
  adjustedRandIndex(celltype_sim, results_zinb$cluster)
)
names(ari) <-  c("SASC", "GMM", "Seurat", "ZINB-WaVE")

cat("\n ARI: \n", round(ari, 2))
```

```
##
##  ARI:
##  0.85 0.5 0.6 0.18
```

```
### save running times
runtime_tab <- matrix(c(results_SASC$runtime,
                        results_GMM$runtime,
                        results_seurat$runtime,
                        results_zinb$runtime), nrow = 1)
```

```r
colnames(runtime_tab) <- c("SASC", "GMM", "Seurat", "ZINB-WaVE")
rownames(runtime_tab) <- c("run time")
cat("\n Run time: \n", round(runtime_tab, 2))
```

```
##
##  Run time:
##  137.46 3.63 0.68 495.2
```

```r
### save tables
balanced_acc <- as.matrix(b_acc_all)
results <- cbind(ari, balanced_acc, matrix(runtime_tab, ncol = 1))
rownames(results) <- c("SASC", "GMM", "Seurat", "ZINB-WaVE")
colnames(results) <- c("ARI", "Acc C", "Acc A", "Acc B", "RunTime")

library(xtable)

print(xtable(results, label = "table:sim_results",
             caption = "Results of simulation study."),
      file = paste0(output_dir, "/sim_results_table.tex"))
```