Structured factorization for single-cell gene expression data: simulations

Antonio Canale, Luisa Galtarossa, Davide Risso, Lorenzo Schiavon, Giovanni Toto

Last modified: March 21, 2025

This vignette contains the code developed and used for computing the aggregated evaluation metrics of the simulation study.

```
library(dplyr)
library(ggplot2)
library(tidyr)

# init color palette
model2color <- c("cosin"= alpha("#9B0014", 0.9),
                 "seurat"     = "#1B9E77",
                 "pca"        = "#BF80BF",
                 "glmpca"     = "#0093D5",
                 "fastglmpca" = "#006BAD",
                 "scGBM"      = "#E6AB02",
                 "newWave"    = "#FB933C")
```

Introduction

To generate the matrix of gene expression, we consider the zero mean data generating process defined as

$$y_{ij} = \lfloor \exp(z_{ij}) \rfloor, \quad z_{ij} = \sum_h C_{hij} + \varepsilon_{ij}, \quad \varepsilon_{ij} \sim N(0, \sigma^2).$$

To illustrate the validity and generality of `COSIN`, we consider three scenarios in which η and λ have different sparsity structures. In particular,

1. in scenario 1 we consider two rank-one contributions which induce two groups of genes:

$$\eta_{\cdot 1}, \eta_{\cdot 2} \sim N_n(0, I_n), \quad \lambda_{\cdot 1} \sim N_p(0, I_p),$$
$$\lambda_{j2} = 0, \quad j > p/2, \quad \lambda_{m2} = 1, \quad m \leq p/2;$$

2. in scenario 2 we consider two rank-one contributions which induce two groups of cells:

$$\lambda_{\cdot 1}, \lambda_{\cdot 2} \sim N_p(0, I_p), \quad \eta_{\cdot 1} \sim N_n(0, I_n),$$
$$\eta_{i2} \sim N(0, 0.05^2), \quad i > n/2, \quad \eta_{l2} = 1, \quad l \leq n/2;$$

3. in scenario 3 we consider three rank-one contributions which induce both two groups of cell and two groups of genes:

$$\begin{aligned}
&\eta_{\cdot 1}, \eta_{\cdot 3} \sim N_n(0, I_n), \quad \lambda_{\cdot 1}, \lambda_{\cdot 2} \sim N_p(0, I_p), \\
&\eta_{i2} \sim N(0, 0.05^2), \quad i > n/2, \quad \eta_{l2} = 1, \quad l \leq n/2, \\
&\lambda_{j3} \sim N_p(0, 0.05^2), \quad j > p/2, \quad \lambda_{m3} = 1, \quad m \leq p/2.
\end{aligned}$$

In the next sections we show how to load matrices, compute the aggregated evaluation metrics and explore the results. We report here only the steps related to the third scenario since it is the only one reported in the paper. The steps for the other two scenarios are the same.

Load tables

We load the tables containing the evaluating metrics related to `COSIN`:

```
# # assigning names to columns (scenario 1)
# colnames_s1 <- c("n", "p", "sigma", "s", "r", "RMSE_1", "RMSE_2", "RMSE_tot",
#                  "F1_1", "F1_2", "F1_tot", "MAD_Y", "MAE_Y", "RMSE_el",
#                  "kstar", "sd_1", "sd_2", "sd_tot", "meta")
# # assigning names to columns (scenario 2)
# colnames_s2 <- c("n", "p", "sigma", "s", "r", "RMSE_1", "RMSE_2", "RMSE_tot",
#                  "MAD_Y", "MAE_Y", "RMSE_el", "kstar", "sd_1", "sd_2", "sd_tot", "meta")
metrics_s3 <- rbind(readRDS("simulations/metrics/table_metrics_meta_NA_p100_s3.RDS"),
                    readRDS("simulations/metrics/table_metrics_nometa_NA_p100_s3.RDS"),
                    readRDS("simulations/metrics/table_metrics_meta_NA_p1000_s3.RDS"),
                    readRDS("simulations/metrics/table_metrics_nometa_NA_p1000_s3.RDS"))
metrics_s3_woNA <- rbind(readRDS("simulations/metrics/table_metrics_meta_woNA_p100_s3.RDS"),
                         readRDS("simulations/metrics/table_metrics_nometa_woNA_p100_s3.RDS"),
                         readRDS("simulations/metrics/table_metrics_meta_woNA_p1000_s3.RDS"),
                         readRDS("simulations/metrics/table_metrics_nometa_woNA_p1000_s3.RDS"))
# assigning names to columns
colnames_s3 <- c("n", "p", "sigma", "s", "r", "RMSE_1", "RMSE_2",
                 "RMSE_3", "RMSE_tot", "MAD_Y", "MAE_Y", "RMSE_el",
                 "kstar", "sd_1", "sd_2", "sd_3", "sd_tot", "meta")
colnames(metrics_s3) <- colnames(metrics_s3_woNA) <- colnames_s3
```

We load the tables containing the evaluating metrics related to `PPCA`:

```
# # assigning names to columns (scenario 1)
# colnames_pca_s1 <- c("n", "p", "sigma", "r", "nPc", "MAE_a", "MAE")
# # assigning names to columns (scenario 2)
# colnames(metrics_pca_s2) <- c("n", "p", "sigma", "r", "nPc", "MAE_a", "MAE")
metrics_pca_s3 <- readRDS("simulations/pca/table_metrics_pca_s3.RDS")
colnames_pca_s3 <- c("n", "p", "sigma", "r", "nPc", "MAE_na", "MAE")

colnames(metrics_pca_s3) <- colnames_pca_s3
```

We load the tables containing the evaluating metrics related to `GLM-PCA`:

```
# # assigning names to columns (scenario 1)
# colnames_glmpca_s1 <- c("n", "p", "sigma", "r", "alpha_a", "MAE_a",
```

```
#                           "MAE", "RMSE_1", "RMSE_2", "RMSE_tot",
#                           "rmse_etaLambda", "meta")
# # assigning names to columns (scenario 2)
# colnames(metrics_glmpca_s2) <- c("n", "p", "sigma", "r", "alpha_a", "MAE_a",
#                                  "MAE", "RMSE_1", "RMSE_2", "RMSE_tot",
#                                  "rmse_etaLambda", "meta")
metrics_glmpca_s3 <- readRDS("simulations/glmpca/table_metrics_nometa_glmpca_s3.RDS")
metrics_glmpca_s3_meta <- readRDS("simulations/glmpca/table_metrics_meta_glmpca_s3.RDS")
colnames_glmpca_s3 <- c("n", "p", "sigma", "r", "alpha_a", "MAE_a",
                        "MAE", "RMSE_1", "RMSE_2", "RMSE_3", "RMSE_tot",
                        "rmse_etaLambda", "meta")
colnames(metrics_glmpca_s3) <- colnames(metrics_glmpca_s3_meta) <- colnames_glmpca_s3
```

Aggregate metrics

First, we aggregate the evaluation metrics related to `COSIN` applied to data sets with and without NAs:

```
# with NAs
metrics_s3 <- as.data.frame(metrics_s3)
metrics_s3$s <- NULL
metrics_s3$r <- NULL
# aggregating metrics
metrics_s3 <- metrics_s3 %>% group_by(n, p, meta, sigma) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# ordering columns
metrics_s3 <- metrics_s3[, c(1:4, rbind(5:16, 5:16+12))]
# changing colnames
colnames(metrics_s3) <- colnames(metrics_s3) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")

# without NAs
metrics_s3_woNA <- as.data.frame(metrics_s3_woNA)
metrics_s3_woNA$s <- NULL
metrics_s3_woNA$r <- NULL
# filtering columns
metrics_s3_woNA <- metrics_s3_woNA[, c(1:3,4:6,11,16)]
# aggregating metrics
metrics_s3_woNA <- metrics_s3_woNA %>% group_by(n, p, meta, sigma) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# ordering columns
metrics_s3_woNA <- metrics_s3_woNA[, c(1:4, rbind(5:8, 5:8+4))]
# changing colnames
colnames(metrics_s3_woNA) <- colnames(metrics_s3_woNA) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")
```

Then, we aggregate the evaluation metrics related to `GLM-PCA` with meta-covariate applied to data sets without NAs:

```
metrics_glmpca_s3_meta <- as.data.frame(metrics_glmpca_s3_meta)
metrics_glmpca_s3_meta$r <- NULL
```

```
# aggregating metrics
metrics_glmpca_s3_meta <- metrics_glmpca_s3_meta %>% group_by(n, p, meta, sigma) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# ordering columns
metrics_glmpca_s3_meta <- metrics_glmpca_s3_meta[, c(1:4, rbind(c(5:10,12), c(5:10,12)+8))]
# changing colnames
colnames(metrics_glmpca_s3_meta) <- colnames(metrics_glmpca_s3_meta) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")
```

Then, we aggregate the evaluation metrics related to `PPCA` applied to data sets with NAs:

```
metrics_pca_s3 <- as.data.frame(metrics_pca_s3)
metrics_pca_s3$r <- NULL
# aggregating metrics
metrics_pca_s3 <- metrics_pca_s3 %>% group_by(n, p, sigma) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# ordering columns
metrics_pca_s3 <- metrics_pca_s3[, c(1:3, rbind(4:6, 4:6+3))]
# changing colnames
colnames(metrics_pca_s3) <- colnames(metrics_pca_s3) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")
```

Finally, we aggregate the evaluation metrics related to `GLM-PCA` without meta-covariate applied to both kinds of data set:

```
metrics_glmpca_s3 <- as.data.frame(metrics_glmpca_s3)
metrics_glmpca_s3$r <- NULL
# aggregating metrics
metrics_glmpca_s3 <- metrics_glmpca_s3 %>% group_by(n, p, meta, sigma) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# ordering columns
metrics_glmpca_s3 <- metrics_glmpca_s3[, c(1:4, rbind(c(5:10,12), c(5:10,12)+8))]
# changing colnames
colnames(metrics_glmpca_s3) <- colnames(metrics_glmpca_s3) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")
```

Contribution RMSE computed on data sets without NAs

Median of contribution RMSE in 50 replicates, with varying (n, p, σ). Interquartile range is also reported.

```
# COSIN
round(metrics_s3_woNA, 2)
```

```
## # A tibble: 12 x 12
## # Groups:   n, p, meta [6]
##        n     p  meta sigma `RMSE_1 median` `RMSE_1 IQR` `RMSE_2 median`
##    <dbl> <dbl> <dbl> <dbl>           <dbl>        <dbl>           <dbl>
##  1    50   100     0   0.1            0.58         0.29            0.56
```

```
##  2    50   100     0   1             0.6          0.26           0.58
##  3    50   100     1   0.1           0.63         0.25           0.58
##  4    50   100     1   1             0.62         0.3            0.57
##  5   200   100     0   0.1           0.6          0.33           0.51
##  6   200   100     0   1             0.46         0.37           0.45
##  7   200   100     1   0.1           0.54         0.35           0.45
##  8   200   100     1   1             0.48         0.41           0.46
##  9   200  1000     0   0.1           0.86         0.15           0.47
## 10   200  1000     0   1             0.62         0.19           0.53
## 11   200  1000     1   0.1           0.87         0.12           0.47
## 12   200  1000     1   1             0.62         0.12           0.55
## # i 5 more variables: `RMSE_2 IQR` <dbl>, `RMSE_3 median` <dbl>,
## #   `RMSE_3 IQR` <dbl>, `kstar median` <dbl>, `kstar IQR` <dbl>
```

```
# GLM-PCA with meta-covariates
round(metrics_glmpca_s3_meta[,c(1:4,8:10,16:18)], 2)
```

```
## # A tibble: 6 x 10
## # Groups:   n, p, meta [3]
##       n     p  meta sigma `MAE_a IQR` `MAE median` `MAE IQR` `RMSE_3 IQR`
##   <dbl> <dbl> <dbl> <dbl>       <dbl>        <dbl>     <dbl>        <dbl>
## 1    50   100     1   0.1        1.43         0.22      0.06         0.18
## 2    50   100     1   1          2.86         2.26      0.86         0.13
## 3   200   100     1   0.1        1.13         0.28      0.05         0.03
## 4   200   100     1   1          4.19         2.97      0.77         0.07
## 5   200  1000     1   0.1        0.78         0.36      1.38         0.05
## 6   200  1000     1   1          2.46         4.48      5.49         0.05
## # i 2 more variables: `rmse_etaLambda median` <dbl>, `rmse_etaLambda IQR` <dbl>
```

```
# GLM-PCA without meta-covariates
round(metrics_glmpca_s3[,c(1:4,8:10,16:18)], 2)
```

```
## # A tibble: 6 x 10
## # Groups:   n, p, meta [3]
##       n     p  meta sigma `MAE_a IQR` `MAE median` `MAE IQR` `RMSE_3 IQR`
##   <dbl> <dbl> <dbl> <dbl>       <dbl>        <dbl>     <dbl>        <dbl>
## 1    50   100     0   0.1        1.43         0.23      0.06         0.15
## 2    50   100     0   1          2.86         2.8       1.02         0.21
## 3   200   100     0   0.1        1.13         0.29      0.04         0.06
## 4   200   100     0   1          4.19         3.65      1.14         0.09
## 5   200  1000     0   0.1        0.78         0.33      0.03         0.06
## 6   200  1000     0   1          2.46         3.94      1.27         0.11
## # i 2 more variables: `rmse_etaLambda median` <dbl>, `rmse_etaLambda IQR` <dbl>
```

Out-of-sample MAE computed on data sets with NAs

Median of out-of-sample MAE in 50 replicates, with varying (n, p, σ). Interquartile range is also reported.

```
# COSIN
round(metrics_s3[,c(1:4,15:16)], 4)
```

```
## # A tibble: 12 x 6
## # Groups:   n, p, meta [6]
##        n     p  meta sigma `MAE_Y median` `MAE_Y IQR`
##    <dbl> <dbl> <dbl> <dbl>          <dbl>       <dbl>
##  1    50   100     0   0.1          0.746       0.524
##  2    50   100     0   1            4.61        2.83
##  3    50   100     1   0.1          0.754       0.508
##  4    50   100     1   1            4.56        2.84
##  5   200   100     0   0.1          0.505       0.275
##  6   200   100     0   1            4.55        2.62
##  7   200   100     1   0.1          0.507       0.278
##  8   200   100     1   1            4.54        2.60
##  9   200  1000     0   0.1          0.420       0.112
## 10   200  1000     0   1            4.79        1.49
## 11   200  1000     1   0.1          0.417       0.120
## 12   200  1000     1   1            4.79        1.50
```

```
# PPCA
round(metrics_pca_s3[,c(1:3,5,8)], 4)
```

```
## # A tibble: 6 x 5
## # Groups:   n, p [3]
##       n     p sigma `nPc IQR` `MAE median`
##   <dbl> <dbl> <dbl>     <dbl>        <dbl>
## 1    50   100   0.1      3.75         3.47
## 2    50   100   1        1            5.67
## 3   200   100   0.1      2            3.40
## 4   200   100   1        1            6.22
## 5   200  1000   0.1      0            3.73
## 6   200  1000   1        1.75         6.51
```

```
# GLM-PCA without meta-covariates
round(metrics_glmpca_s3[,c(1:4,6,14)], 4)
```

```
## # A tibble: 6 x 6
## # Groups:   n, p, meta [3]
##       n     p  meta sigma `alpha_a IQR` `RMSE_2 IQR`
##   <dbl> <dbl> <dbl> <dbl>         <dbl>        <dbl>
## 1    50   100     0   0.1           1          1.30
## 2    50   100     0   1             2          0.180
## 3   200   100     0   0.1           0.75       0.414
## 4   200   100     0   1             1.75       0.170
## 5   200  1000     0   0.1           0          0.637
## 6   200  1000     0   1             2          0.0992
```

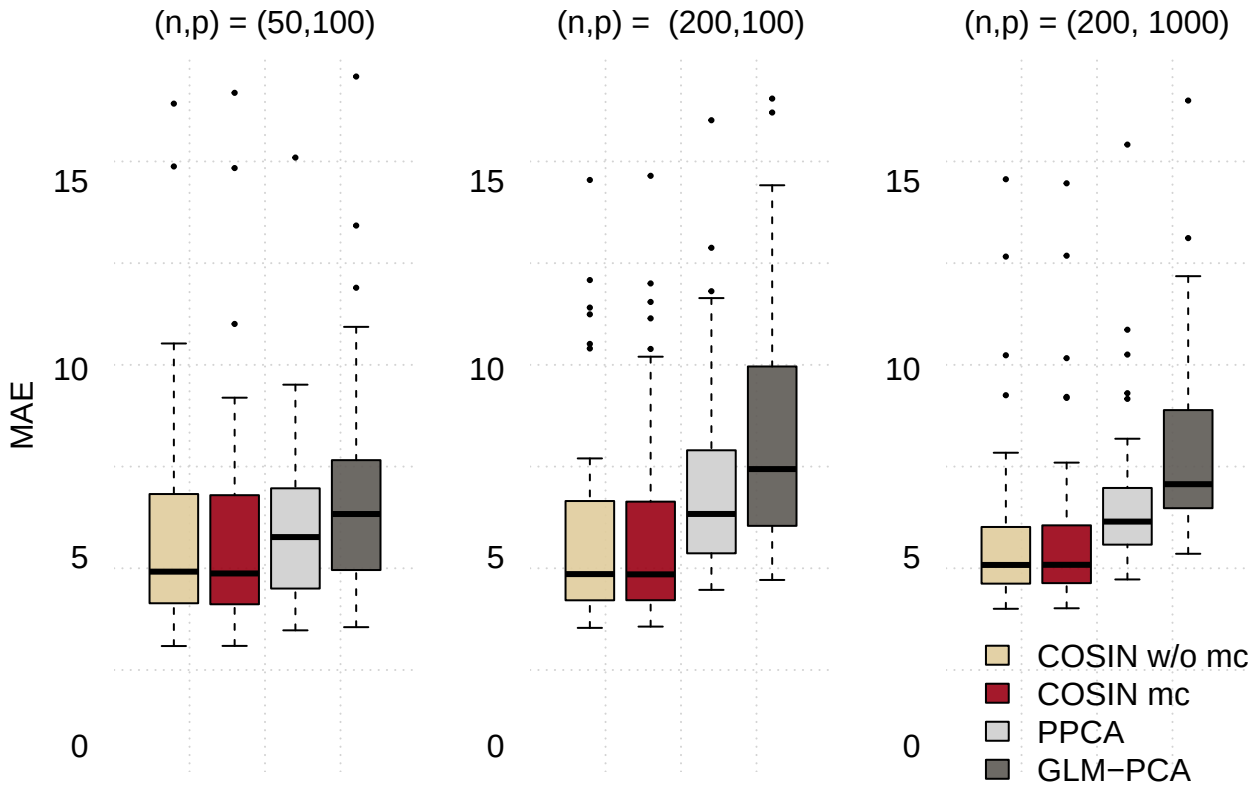
Boxplot of the out-of-sample MAE computed on data sets with NAs

Boxplot of the out-of-sample MAE of the competing models under different values of (n, p) with $\sigma^2 = 1$:

```
metrics <- rbind(readRDS("simulations/metrics/table_metrics_meta_NA_p100_s3.RDS"),
                 readRDS("simulations/metrics/table_metrics_nometa_NA_p100_s3.RDS"),
```

```
                readRDS("simulations/metrics/table_metrics_meta_NA_p1000_s3.RDS"),
                readRDS("simulations/metrics/table_metrics_nometa_NA_p1000_s3.RDS"))
colnames(metrics) <- c("n", "p", "sigma", "s", "r", "RMSE_1", "RMSE_2",
                       "RMSE_3", "RMSE_tot", "MAD_Y", "MAE_Y", "RMSE_el",
                       "kstar", "sd_1", "sd_2", "sd_3", "sd_tot", "meta")
metrics <- as.data.frame(metrics)
metrics$n_p_sigma_s <- paste(metrics$n, metrics$p, metrics$sigma, metrics$s, sep="_")
metrics <- metrics[, c("meta", "n_p_sigma_s", "r", "MAE_Y")]


metrics_pca <- readRDS("simulations/pca/table_metrics_pca_s3.RDS")
colnames(metrics_pca) <- c("n", "p", "sigma", "r", "nPc", "MAE_na", "MAE")
metrics_pca <- as.data.frame(metrics_pca)
metrics_pca$n_p_sigma_s <- paste(metrics_pca$n, metrics_pca$p, metrics_pca$sigma, 3, sep="_")
metrics_pca <- metrics_pca[, c("n_p_sigma_s", "r", "MAE_na")]

metrics_glmpca <- readRDS("simulations/metrics/table_metrics_nometa_glmpca_s3.RDS")
colnames(metrics_glmpca) <- c("n", "p", "sigma", "r", "alpha_a", "MAE_a",
                              "MAE", "RMSE_1", "RMSE_2", "RMSE_3", "RMSE_tot",
                              "rmse_etaLambda", "meta")
metrics_glmpca <- as.data.frame(metrics_glmpca)
metrics_glmpca$n_p_sigma_s <- paste(metrics_glmpca$n, metrics_glmpca$p, metrics_glmpca$sigma, 3, sep="_
metrics_glmpca <- metrics_glmpca[, c("meta", "n_p_sigma_s", "r", "MAE_a")]


scenario_list <- c("50_100_1_3", "200_100_1_3", "200_1000_1_3")

datatoplot_list <- list()
for(scenario in scenario_list) {
  datatoplot_list[[scenario]] <- cbind(metrics[metrics$meta==0 & metrics$n_p_sigma_s==scenario,"MAE_Y"]
                                       metrics[metrics$meta==1 & metrics$n_p_sigma_s==scenario,"MAE_Y"]
                                       metrics_pca[metrics_pca$n_p_sigma_s==scenario,"MAE_na"],
                                       metrics_glmpca[metrics_glmpca$n_p_sigma_s==scenario,"MAE_a"])
}

scenario_titles <- list("50_100_1_3" = "(n,p) = (50,100)",
                        "200_100_1_3" = "(n,p) =  (200,100)",
                        "200_1000_1_3" = "(n,p) = (200, 1000)")

unipdred <- scales::alpha("#9B0014", 0.9)
mybeige <- scales::alpha("#e1cd9d", 0.9)
mygrey <- scales::alpha("#5e5a55", 0.9)

par(mfrow=c(1,3), mar=c(3,4.5,3,0), xpd = FALSE)
for(scenario in scenario_list) {
  # 50x3 data set in which each column contains MAE of the 50 simulations computed by a model
  datatoplot <- as.data.frame(datatoplot_list[[scenario]])
  # plot settings
  if(scenario == "50_100_1_3"){
    plot(1, 1, pch = "", xlim = c(0.2,4.8), ylim = c(0, 17.5),
         axes = FALSE, xlab = "", ylab="MAE", cex.lab=1.5)
  }else{
    plot(1, 1, pch = "", xlim = c(0.2,4.8), ylim = c(0, 17.5),
```

```
        axes = FALSE, xlab = "", ylab="")
  }
  par(xpd = FALSE)
  grid(nx = 4, ny = 7)
  # boxplot
  par(xpd = TRUE)
  boxplot(datatoplot, col = c( mybeige, unipdred, "lightgrey", mygrey),
          add = TRUE, axes = FALSE, pch = 20)
  #axis(1, at = c(1, 2, 3), labels = c("COSIN","COSIN2", "GLMPCA"), cex.axis = 1.5, tick = FALSE)
  axis(2, tick = FALSE, las = 1, cex.axis=1.5)
  #abline(h = 0, lwd = 2)
  title(main = scenario_titles[[scenario]], cex.main=1.5, font.main = 1)
  if(scenario =="200_1000_1_3"){
     par(xpd = TRUE)
     legend("bottomright", legend = c("COSIN w/o mc","COSIN mc", "PPCA", "GLM-PCA"),
            fill = c(mybeige, unipdred, "lightgrey", mygrey), cex=1.5, pt.cex = 2,
            bg = "white", horiz =F, inset = -0.05, bty="n")
  }
}
```



Additional simulations: clustering of cells

We consider an additional scenario in which cells are grouped in G groups. We evaluate in terms of Adjusted Rand Index (ARI) the ability of cosin in retrieving the true cell groups, and compare it with other state-of-the-art approaches.

To generate the matrix of gene expression, we consider the same zero mean data generating process,

$$y_{ij} = \lfloor \exp(z_{ij}) \rfloor, \quad z_{ij} = \sum_h C_{hij} + \varepsilon_{ij}, \quad \varepsilon_{ij} \sim N(0, \sigma^2).$$

As in scenario 3 in the previous simulations, we consider three rank-one contributions,

$$\begin{aligned} \eta_{\cdot 1}, \eta_{\cdot 3} &\sim N_n(\mu_{g1}, I_n), \quad \lambda_{\cdot 1}, \lambda_{\cdot 2} \sim N_p(\mu_{g3}, I_p), \\ \eta_{i2} &\sim N(0, 0.05^2), \quad i > n/2, \quad \eta_{l2} = 1, \quad l \leq n/2, \\ \lambda_{j3} &\sim N_p(0, 0.05^2), \quad j > p/2, \quad \lambda_{m3} = 1, \quad m \leq p/2, \end{aligned}$$

where cell i belongs to group $g = 1, \ldots, G$,a nd the group means μ_{gh} are sampled from $N(0, 3^2)$.

```
# loading metric tables
metrics <- readRDS("simulations/clustering/table_metric_clustering_kmeans.RDS")
metrics <- as.data.frame(metrics)
colnames(metrics) <- c("n", "p", "sigma", "J", "s", "r", "sim$k", "K", "cosin",
                       "pca_K", "glmpca_K", "fastglmpca_K", "scGBM_K", "newWave_K",
                       "pca", "glmpca", "fastglmpca", "scGBM", "newWave")
# incorrect number of factors
metrics_K <- metrics[,c("n", "p", "sigma", "J", "cosin", "pca_K", "glmpca_K", "fastglmpca_K", "scGBM_K"
colnames(metrics_K) <- c("n", "p", "sigma", "J", "cosin", "pca", "glmpca", "fastglmpca", "scGBM", "newW
# correct number of factors
metrics_simk <- metrics[,c("n", "p", "sigma", "J", "cosin", "pca", "glmpca", "fastglmpca", "scGBM", "ne
```

`COSIN` does not require pre-specification of the number of latent dimensions, while competing methods necessitate setting the number of latent factors.

We consider a first setting in which the number of latent dimensions is known.

```
metrics <- metrics_summ <- metrics_simk
# replace NAs with -1
metrics_summ[is.na(metrics_summ)] <- -1

# aggregating metrics
metrics_summ <- metrics_summ %>% group_by(n, p, sigma, J) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# renaming columns
colnames(metrics_summ) <- colnames(metrics_summ) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")
# table
metrics_summ
```

```
## # A tibble: 12 x 16
## # Groups:   n, p, sigma [6]
##        n     p sigma     J `cosin median` `pca median` `glmpca median`
##    <dbl> <dbl> <dbl> <dbl>          <dbl>        <dbl>           <dbl>
##  1    50   100   0.1     5          0.581        0.504           0.460
##  2    50   100   0.1    10          0.331        0.333           0.303
##  3    50   100   1       5          0.431        0.467           0.341
##  4    50   100   1      10          0.269        0.322           0.234
```

```
##  5   200   100   0.1     5          0.625        0.526          0.472
##  6   200   100   0.1    10          0.362        0.362          0.252
##  7   200   100   1       5          0.514        0.489          0.318
##  8   200   100   1      10          0.315        0.376          0.179
##  9   200  1000   0.1     5          0.731        0.420          0.0554
## 10   200  1000   0.1    10          0.492        0.285         -1
## 11   200  1000   1       5          0.590        0.428          0.129
## 12   200  1000   1      10          0.393        0.317         -1
## # i 9 more variables: `fastglmpca median` <dbl>, `scGBM median` <dbl>,
## #   `newWave median` <dbl>, `cosin IQR` <dbl>, `pca IQR` <dbl>,
## #   `glmpca IQR` <dbl>, `fastglmpca IQR` <dbl>, `scGBM IQR` <dbl>,
## #   `newWave IQR` <dbl>
```

```
# ggplot
metrics$np_J <- paste(metrics$n, metrics$p, metrics$J, sep = "_")
metrics <- metrics[,c("np_J","sigma","cosin","pca","glmpca","fastglmpca","scGBM","newWave")]

# reshaping the data to long format
metrics_long <- metrics %>% pivot_longer(cols = c("cosin","pca","glmpca","fastglmpca","scGBM","newWave")
                                         names_to = "model", values_to = "value")

metrics_long$np_J <- factor(metrics_long$np_J, levels = c("50_100_5", "200_100_5", "50_100_10", "200_10
levels(metrics_long$np_J) <- c("n = 50, p = 100, J = 5",
                               "n = 200, p = 100, J = 5",
                               "n = 50, p = 100, J = 10",
                               "n = 200, p = 100, J = 10",
                               "n = 200, p = 1000, J = 5",
                               "n = 200, p = 1000, J = 10")

metrics_long$sigma <- factor(metrics_long$sigma, levels = c(0.1,1.0))
levels(metrics_long$sigma) <- paste0("  = ", levels(metrics_long$sigma))

# count missing values
print(metrics_long %>% group_by(model, np_J, sigma) %>% summarise(missing_count = sum(is.na(value)), .g
```

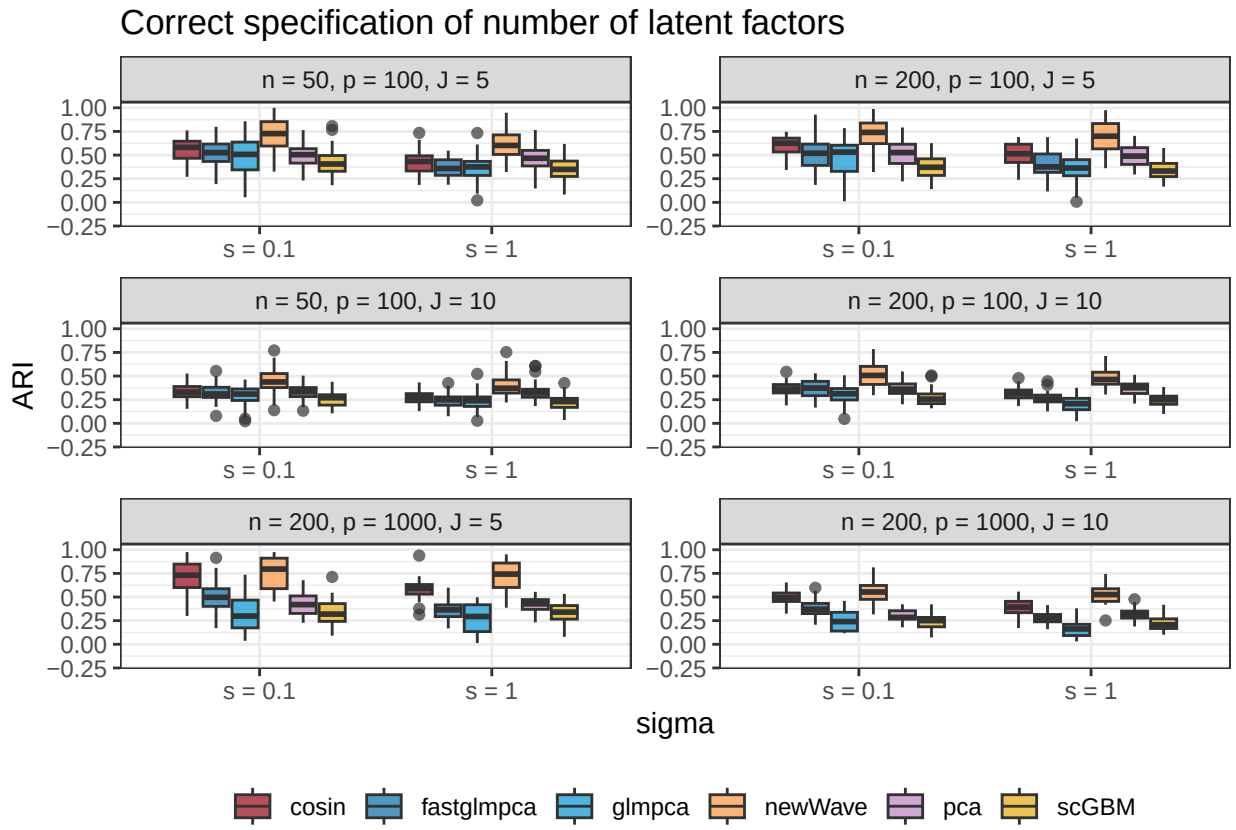
```
## # A tibble: 72 x 4
##    model      np_J                     sigma   missing_count
##    <chr>      <fct>                    <fct>           <int>
##  1 cosin      n = 50, p = 100, J = 5     = 0.1             0
##  2 cosin      n = 50, p = 100, J = 5     = 1               0
##  3 cosin      n = 200, p = 100, J = 5    = 0.1             0
##  4 cosin      n = 200, p = 100, J = 5    = 1               0
##  5 cosin      n = 50, p = 100, J = 10    = 0.1             0
##  6 cosin      n = 50, p = 100, J = 10    = 1               0
##  7 cosin      n = 200, p = 100, J = 10   = 0.1             0
##  8 cosin      n = 200, p = 100, J = 10   = 1               0
##  9 cosin      n = 200, p = 1000, J = 5   = 0.1             0
## 10 cosin      n = 200, p = 1000, J = 5   = 1               0
## 11 cosin      n = 200, p = 1000, J = 10  = 0.1             0
## 12 cosin      n = 200, p = 1000, J = 10  = 1               0
## 13 fastglmpca n = 50, p = 100, J = 5     = 0.1             2
## 14 fastglmpca n = 50, p = 100, J = 5     = 1               2
## 15 fastglmpca n = 200, p = 100, J = 5    = 0.1             1
## 16 fastglmpca n = 200, p = 100, J = 5    = 1               1
```

```
## 17 fastglmpca n = 50, p = 100, J = 10    = 0.1            1
## 18 fastglmpca n = 50, p = 100, J = 10    = 1              0
## 19 fastglmpca n = 200, p = 100, J = 10   = 0.1            1
## 20 fastglmpca n = 200, p = 100, J = 10   = 1              0
## 21 fastglmpca n = 200, p = 1000, J = 5   = 0.1            0
## 22 fastglmpca n = 200, p = 1000, J = 5   = 1              0
## 23 fastglmpca n = 200, p = 1000, J = 10  = 0.1            0
## 24 fastglmpca n = 200, p = 1000, J = 10  = 1              1
## 25 glmpca     n = 50, p = 100, J = 5     = 0.1            7
## 26 glmpca     n = 50, p = 100, J = 5     = 1              5
## 27 glmpca     n = 200, p = 100, J = 5    = 0.1            8
## 28 glmpca     n = 200, p = 100, J = 5    = 1              8
## 29 glmpca     n = 50, p = 100, J = 10    = 0.1            5
## 30 glmpca     n = 50, p = 100, J = 10    = 1              2
## 31 glmpca     n = 200, p = 100, J = 10   = 0.1           15
## 32 glmpca     n = 200, p = 100, J = 10   = 1             11
## 33 glmpca     n = 200, p = 1000, J = 5   = 0.1           14
## 34 glmpca     n = 200, p = 1000, J = 5   = 1             10
## 35 glmpca     n = 200, p = 1000, J = 10  = 0.1           20
## 36 glmpca     n = 200, p = 1000, J = 10  = 1             20
## 37 newWave    n = 50, p = 100, J = 5     = 0.1            0
## 38 newWave    n = 50, p = 100, J = 5     = 1              0
## 39 newWave    n = 200, p = 100, J = 5    = 0.1            0
## 40 newWave    n = 200, p = 100, J = 5    = 1              0
## 41 newWave    n = 50, p = 100, J = 10    = 0.1            0
## 42 newWave    n = 50, p = 100, J = 10    = 1              0
## 43 newWave    n = 200, p = 100, J = 10   = 0.1            0
## 44 newWave    n = 200, p = 100, J = 10   = 1              0
## 45 newWave    n = 200, p = 1000, J = 5   = 0.1            0
## 46 newWave    n = 200, p = 1000, J = 5   = 1              0
## 47 newWave    n = 200, p = 1000, J = 10  = 0.1            0
## 48 newWave    n = 200, p = 1000, J = 10  = 1              0
## 49 pca        n = 50, p = 100, J = 5     = 0.1            0
## 50 pca        n = 50, p = 100, J = 5     = 1              0
## 51 pca        n = 200, p = 100, J = 5    = 0.1            0
## 52 pca        n = 200, p = 100, J = 5    = 1              0
## 53 pca        n = 50, p = 100, J = 10    = 0.1            0
## 54 pca        n = 50, p = 100, J = 10    = 1              0
## 55 pca        n = 200, p = 100, J = 10   = 0.1            0
## 56 pca        n = 200, p = 100, J = 10   = 1              0
## 57 pca        n = 200, p = 1000, J = 5   = 0.1            0
## 58 pca        n = 200, p = 1000, J = 5   = 1              0
## 59 pca        n = 200, p = 1000, J = 10  = 0.1            0
## 60 pca        n = 200, p = 1000, J = 10  = 1              0
## 61 scGBM      n = 50, p = 100, J = 5     = 0.1            0
## 62 scGBM      n = 50, p = 100, J = 5     = 1              0
## 63 scGBM      n = 200, p = 100, J = 5    = 0.1            0
## 64 scGBM      n = 200, p = 100, J = 5    = 1              0
## 65 scGBM      n = 50, p = 100, J = 10    = 0.1            0
## 66 scGBM      n = 50, p = 100, J = 10    = 1              0
## 67 scGBM      n = 200, p = 100, J = 10   = 0.1            0
## 68 scGBM      n = 200, p = 100, J = 10   = 1              0
## 69 scGBM      n = 200, p = 1000, J = 5   = 0.1            0
## 70 scGBM      n = 200, p = 1000, J = 5   = 1              0
```

```
## 71 scGBM      n = 200, p = 1000, J = 10   = 0.1              0
## 72 scGBM      n = 200, p = 1000, J = 10   = 1                0
```

```
# creating ggplot
ggplot(metrics_long, aes(x = sigma, y = value, fill = model)) +
  geom_boxplot(alpha = 0.7) +
  facet_wrap(~ np_J, scales = "free", ncol = 2) +
  scale_fill_manual(values = model2color) +
  scale_y_continuous(position = "left", breaks = seq(-0.25, 1, by = 0.25)) +
  coord_cartesian(ylim = c(-0.2, 1)) +
  theme_bw() +
  theme(legend.position="bottom", legend.box = "horizontal") +
  guides(fill = guide_legend(nrow = 1)) +
  labs(title = "Correct specification of number of latent factors", x = "sigma", y = "ARI", fill = "")
```

```
## Warning: Removed 134 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



We consider a second setting in which the number of latent dimensions must be identified. We set it as the number of principal components required to explain 90% of the variability in log-transformed counts.

```
metrics <- metrics_summ <- metrics_K
# replace NAs with -1
metrics_summ[is.na(metrics_summ)] <- -1
```

```r
# aggregating metrics
metrics_summ <- metrics_summ %>% group_by(n, p, sigma, J) %>%
  summarise_all(.funs = c("median" = median, "IQR" = IQR))
# renaming columns
colnames(metrics_summ) <- colnames(metrics_summ) %>%
  stringr::str_replace_all("_median", " median") %>%
  stringr::str_replace("_IQR", " IQR")
# table
metrics_summ
```

```
## # A tibble: 12 x 16
## # Groups:   n, p, sigma [6]
##        n     p sigma     J `cosin median` `pca median` `glmpca median`
##    <dbl> <dbl> <dbl> <dbl>          <dbl>        <dbl>            <dbl>
##  1    50   100   0.1     5          0.581       0.529            0.551
##  2    50   100   0.1    10          0.331       0.372            0.332
##  3    50   100   1       5          0.431       0.292            0.407
##  4    50   100   1      10          0.269       0.243            0.279
##  5   200   100   0.1     5          0.625       0.610            0.404
##  6   200   100   0.1    10          0.362       0.404            0.335
##  7   200   100   1       5          0.514       0.401            0.300
##  8   200   100   1      10          0.315       0.347            0.245
##  9   200  1000   0.1     5          0.731       0.530            0.140
## 10   200  1000   0.1    10          0.492       0.391           -1
## 11   200  1000   1       5          0.590       0.0517           0.174
## 12   200  1000   1      10          0.393       0.173           -1
## # i 9 more variables: `fastglmpca median` <dbl>, `scGBM median` <dbl>,
## #   `newWave median` <dbl>, `cosin IQR` <dbl>, `pca IQR` <dbl>,
## #   `glmpca IQR` <dbl>, `fastglmpca IQR` <dbl>, `scGBM IQR` <dbl>,
## #   `newWave IQR` <dbl>
```

```r
# ggplot
metrics$np_J <- paste(metrics$n, metrics$p, metrics$J, sep = "_")
metrics <- metrics[,c("np_J","sigma","cosin","pca","glmpca","fastglmpca","scGBM","newWave")]
# reshaping the data to long format
metrics_long <- metrics %>% pivot_longer(cols = c("cosin","pca","glmpca","fastglmpca","scGBM","newWave"
                                         names_to = "model", values_to = "value")
metrics_long$np_J <- factor(metrics_long$np_J, levels = c("50_100_5", "200_100_5", "50_100_10", "200_10
levels(metrics_long$np_J) <- c("n = 50, p = 100, J = 5",
                               "n = 200, p = 100, J = 5",
                               "n = 50, p = 100, J = 10",
                               "n = 200, p = 100, J = 10",
                               "n = 200, p = 1000, J = 5",
                               "n = 200, p = 1000, J = 10")
metrics_long$sigma <- factor(metrics_long$sigma, levels = c(0.1,1.0))
levels(metrics_long$sigma) <- paste0("  = ", levels(metrics_long$sigma))
# count missing values
print(metrics_long %>% group_by(model, np_J, sigma) %>% summarise(missing_count = sum(is.na(value)), .g
```

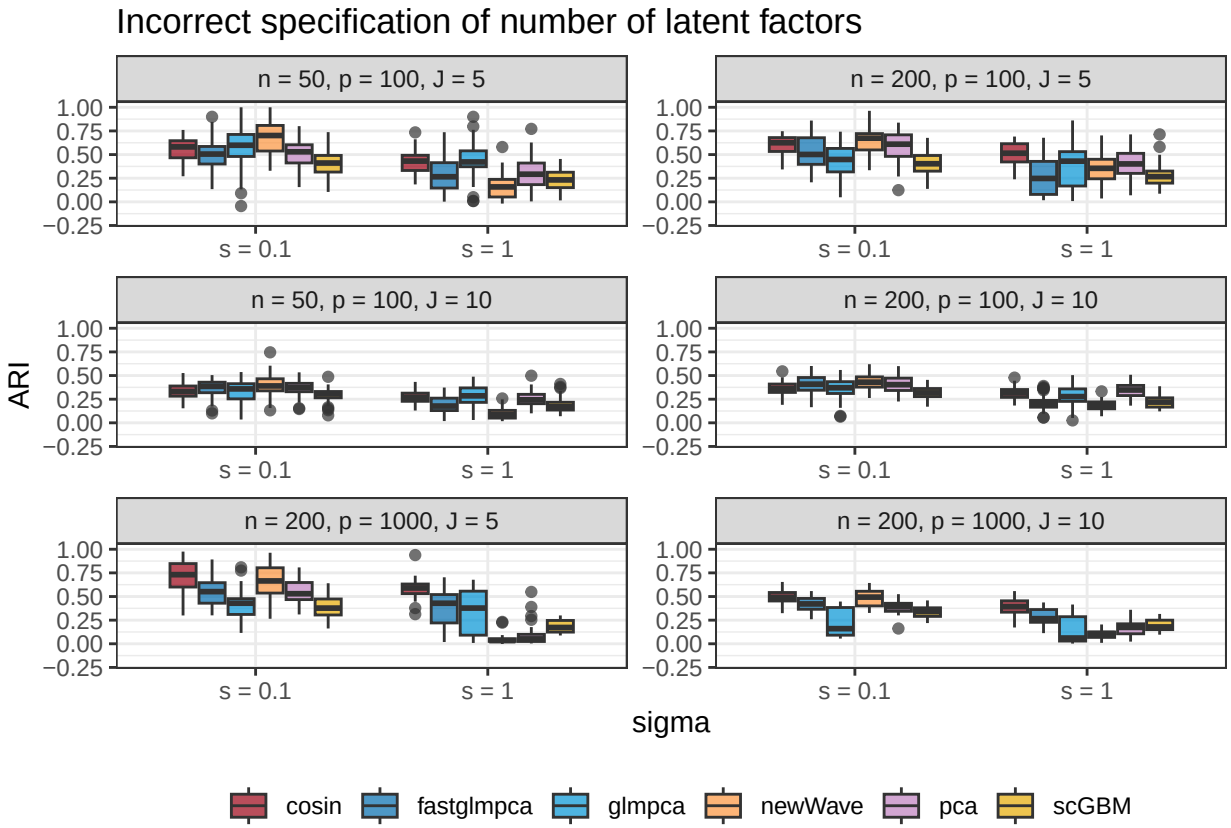
```
## # A tibble: 72 x 4
##    model      np_J                     sigma   missing_count
##    <chr>      <fct>                    <fct>           <int>
##  1 cosin      n = 50, p = 100, J = 5     = 0.1             0
```

```
##  2 cosin      n = 50, p = 100, J = 5     = 1              0
##  3 cosin      n = 200, p = 100, J = 5    = 0.1            0
##  4 cosin      n = 200, p = 100, J = 5    = 1              0
##  5 cosin      n = 50, p = 100, J = 10    = 0.1            0
##  6 cosin      n = 50, p = 100, J = 10    = 1              0
##  7 cosin      n = 200, p = 100, J = 10   = 0.1            0
##  8 cosin      n = 200, p = 100, J = 10   = 1              0
##  9 cosin      n = 200, p = 1000, J = 5   = 0.1            0
## 10 cosin      n = 200, p = 1000, J = 5   = 1              0
## 11 cosin      n = 200, p = 1000, J = 10  = 0.1            0
## 12 cosin      n = 200, p = 1000, J = 10  = 1              0
## 13 fastglmpca n = 50, p = 100, J = 5     = 0.1            0
## 14 fastglmpca n = 50, p = 100, J = 5     = 1              1
## 15 fastglmpca n = 200, p = 100, J = 5    = 0.1            2
## 16 fastglmpca n = 200, p = 100, J = 5    = 1              1
## 17 fastglmpca n = 50, p = 100, J = 10    = 0.1            1
## 18 fastglmpca n = 50, p = 100, J = 10    = 1              0
## 19 fastglmpca n = 200, p = 100, J = 10   = 0.1            1
## 20 fastglmpca n = 200, p = 100, J = 10   = 1              0
## 21 fastglmpca n = 200, p = 1000, J = 5   = 0.1            0
## 22 fastglmpca n = 200, p = 1000, J = 5   = 1              0
## 23 fastglmpca n = 200, p = 1000, J = 10  = 0.1            3
## 24 fastglmpca n = 200, p = 1000, J = 10  = 1              0
## 25 glmpca     n = 50, p = 100, J = 5     = 0.1            9
## 26 glmpca     n = 50, p = 100, J = 5     = 1              6
## 27 glmpca     n = 200, p = 100, J = 5    = 0.1            7
## 28 glmpca     n = 200, p = 100, J = 5    = 1              5
## 29 glmpca     n = 50, p = 100, J = 10    = 0.1            4
## 30 glmpca     n = 50, p = 100, J = 10    = 1              2
## 31 glmpca     n = 200, p = 100, J = 10   = 0.1           14
## 32 glmpca     n = 200, p = 100, J = 10   = 1             10
## 33 glmpca     n = 200, p = 1000, J = 5   = 0.1           13
## 34 glmpca     n = 200, p = 1000, J = 5   = 1              7
## 35 glmpca     n = 200, p = 1000, J = 10  = 0.1           20
## 36 glmpca     n = 200, p = 1000, J = 10  = 1             18
## 37 newWave    n = 50, p = 100, J = 5     = 0.1            0
## 38 newWave    n = 50, p = 100, J = 5     = 1              0
## 39 newWave    n = 200, p = 100, J = 5    = 0.1            0
## 40 newWave    n = 200, p = 100, J = 5    = 1              0
## 41 newWave    n = 50, p = 100, J = 10    = 0.1            0
## 42 newWave    n = 50, p = 100, J = 10    = 1              0
## 43 newWave    n = 200, p = 100, J = 10   = 0.1            0
## 44 newWave    n = 200, p = 100, J = 10   = 1              0
## 45 newWave    n = 200, p = 1000, J = 5   = 0.1            0
## 46 newWave    n = 200, p = 1000, J = 5   = 1              0
## 47 newWave    n = 200, p = 1000, J = 10  = 0.1            0
## 48 newWave    n = 200, p = 1000, J = 10  = 1              0
## 49 pca        n = 50, p = 100, J = 5     = 0.1            0
## 50 pca        n = 50, p = 100, J = 5     = 1              0
## 51 pca        n = 200, p = 100, J = 5    = 0.1            0
## 52 pca        n = 200, p = 100, J = 5    = 1              0
## 53 pca        n = 50, p = 100, J = 10    = 0.1            0
## 54 pca        n = 50, p = 100, J = 10    = 1              0
## 55 pca        n = 200, p = 100, J = 10   = 0.1            0
```

```
## 56 pca        n = 200, p = 100, J = 10    = 1              0
## 57 pca        n = 200, p = 1000, J = 5    = 0.1            0
## 58 pca        n = 200, p = 1000, J = 5    = 1              0
## 59 pca        n = 200, p = 1000, J = 10   = 0.1            0
## 60 pca        n = 200, p = 1000, J = 10   = 1              0
## 61 scGBM      n = 50, p = 100, J = 5      = 0.1            0
## 62 scGBM      n = 50, p = 100, J = 5      = 1              0
## 63 scGBM      n = 200, p = 100, J = 5     = 0.1            0
## 64 scGBM      n = 200, p = 100, J = 5     = 1              0
## 65 scGBM      n = 50, p = 100, J = 10     = 0.1            0
## 66 scGBM      n = 50, p = 100, J = 10     = 1              0
## 67 scGBM      n = 200, p = 100, J = 10    = 0.1            0
## 68 scGBM      n = 200, p = 100, J = 10    = 1              0
## 69 scGBM      n = 200, p = 1000, J = 5    = 0.1            0
## 70 scGBM      n = 200, p = 1000, J = 5    = 1              0
## 71 scGBM      n = 200, p = 1000, J = 10   = 0.1            0
## 72 scGBM      n = 200, p = 1000, J = 10   = 1              0
```

```
# creating ggplot
ggplot(metrics_long, aes(x = sigma, y = value, fill = model)) +
  geom_boxplot(alpha = 0.7) +
  facet_wrap(~ np_J, scales = "free", ncol = 2) +
  scale_fill_manual(values = model2color) +
  scale_y_continuous(position = "left", breaks = seq(-0.25, 1, by = 0.25)) +
  coord_cartesian(ylim = c(-0.2, 1)) +
  theme_bw() +
  theme(legend.position="bottom", legend.box = "horizontal") +
  guides(fill = guide_legend(nrow = 1)) +
  labs(title = "Incorrect specification of number of latent factors", x = "sigma", y = "ARI", fill = "")
```

```
## Warning: Removed 124 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```


Incorrect specification of number of latent factors
n = 50, p = 100, J = 5
n = 200, p = 100, J = 5
n = 50, p = 100, J = 10
n = 200, p = 100, J = 10
n = 200, p = 1000, J = 5
n = 200, p = 1000, J = 10
ARI
1.00
0.75
0.50
0.25
0.00
−0.25
s = 0.1
s = 1
sigma
cosin
fastglmpca
glmpca
newWave
pca
scGBM