Supplementary Material for
Sparse composite likelihood selection

Claudia Di Caterina and Davide Ferrari

1 Introduction

The current report reproduces the figure and numerical results in Section 5.1 of the main text. The outputs have been produced using R version 4.0.3 (R Core Team 2021). The code chunk below loads the necessary R packages.

```
library(matrixcalc)
library(MASS)
library(mvtnorm)
library(ggplot2)
library(plyr)
```

We also provide code to reproduce all simulation results in the paper. The R scripts to carry out the simulation experiments, and the results from those, are provided in the `sparseCLsel.zip` archive. `res_dir` is the directory where the contents of the archive are and needs to be set appropriately.

```
res_dir <- "~/Dropbox/DiCaterinaFerrari2020/Supplementary/sparseCLsel"
```

2 Simulation studies

This section provides the R code that reproduces the output of Section 5.1 in the paper. The function used to summarize the results is reported below.

```
output_fun <- function(res) {
  alphas <- res[[1]]$alphas
  nsim <- length(res)

  lambda1 <- res[[1]]$lambda_seq
  nlambda <- length(lambda1)

  lambda <- res[[1]]$lambdahat
  nlambdahat <- length(lambda)

  null_sample <- which(ldply(res, function(x) is.null(x)) == TRUE)
  eff_nsim <- nsim - length(null_sample)

  d <- attr(data.sim, "d")
  p <- attr(data.sim, "p")
  if(is.null(p)) p <- d * (d - 1)/2
  pstar <- attr(data.sim, "pstar")
  if(is.null(pstar)) {
```

```
    pstar <- attr(data.sim, "n_edg")

    results <- ldply(res, function(x) data.frame(lambda = x$lambda_seq,
                                                 nsel = x$nsel, Delta = unname(x$Delta),
                                                 Doracle = unique(x$Delta_oracle),
                                                 Dmle = unique(x$Delta_mle)))

reshat <- ldply(res, function(x) data.frame(lambdahat = x$lambdahat,
                                            nselhat = x$nselhat,
                                            Dhat = unname(x$Deltahat),
                                            tpphat = unname(x$tpphat),
                                            fdphat = unname(x$fdphat),
                                            fprhat = unname((x$nselhat -
                                            x$tpphat * pstar)/(p - pstar))))

res_set <- data.frame(id = rep(1:eff_nsim, each = nlambda),
                      seq_val = rep(1:nlambda, times = eff_nsim), results)

reshat_set <- data.frame(id = rep(1:eff_nsim, each = nlambdahat),
                         seq_val = rep(1:nlambdahat, times = eff_nsim),
                         reshat)

stats_set <- ddply(res_set, ~ seq_val, function(x) {
  re <- mean(x$Doracle, na.rm = TRUE)/mean(x$Delta, na.rm = TRUE)
  re_mle <- mean(x$Dmle, na.rm = TRUE)/mean(x$Delta, na.rm = TRUE)
  RMSE <- mean(x$Delta, na.rm = TRUE)
  ave_sel <- mean(x$nsel)

  out <- data.frame(RMSE = RMSE, re = re, avr_sel = ave_sel, re_mle = re_mle)
  out
})

statshat_set <- ddply(reshat_set, ~ lambdahat, function(x) {
  ave_TPP <- mean(x$tpphat, na.rm = TRUE)
  ave_FDP <- mean(x$fdphat, na.rm = TRUE)
  ave_TNR <- 1 - mean(x$fprhat, na.rm = TRUE)
  ave_sel <- mean(x$nselhat, na.rm = TRUE)

  out <- data.frame(avrTPP = ave_TPP, avrFDP = ave_FDP, avr_sel = ave_sel,
                    avrTNR = ave_TNR)
  out
})
}
else {
  results <- ldply(res, function(x) data.frame(lambda = x$lambda_seq,
                                               nsel = x$nsel, RMSE = unname(x$rmse),
                                               RMSEoracle = unique(x$rmse_oracle),
                                               RMSEmle = unique(x$rmse_mle)))

  reshat <- ldply(res, function(x) data.frame(lambdahat = x$lambdahat,
                                              nselhat = x$nselhat,
                                              RMSEhat = unname(x$rmsehat),
                                              tpphat = unname(x$tpphat),
```

```
                                                fdphat = unname(x$fdphat),
                                                fprhat = unname((x$nselhat -
                                                  x$tpphat * pstar)/(p - pstar))))

  res_set <- data.frame(id = rep(1:eff_nsim, each = nlambda),
                        seq_val = rep(1:nlambda, times = eff_nsim), results)

  reshat_set <- data.frame(id = rep(1:eff_nsim, each = nlambdahat),
                           seq_val = rep(1:nlambdahat, times = eff_nsim),
                           reshat)

  stats_set <- ddply(res_set, ~ seq_val, function(x) {
    re <- mean(x$RMSEoracle, na.rm = TRUE)/mean(x$RMSE, na.rm = TRUE)
    re_mle <- mean(x$RMSEmle, na.rm = TRUE)/mean(x$RMSE, na.rm = TRUE)
    RMSE <- mean(x$RMSE, na.rm = TRUE)
    ave_sel <- mean(x$nsel)

    out <- data.frame(RMSE = RMSE, re = re, avr_sel = ave_sel, re_mle = re_mle)
    out
  })

  statshat_set <- ddply(reshat_set, ~ lambdahat, function(x) {
    ave_TPP <- mean(x$tpphat, na.rm = TRUE)
    ave_FDP <- mean(x$fdphat, na.rm = TRUE)
    ave_TNR <- 1 - mean(x$fprhat, na.rm = TRUE)
    ave_sel <- mean(x$nselhat, na.rm = TRUE)

    out <- data.frame(avrTPP = ave_TPP, avrFDP = ave_FDP, avr_sel = ave_sel,
                      avrTNR = ave_TNR)
    out
  })
}
  list(stats_set = stats_set, statshat_set = statshat_set)
}
```

2.1 Table 1

The following code chunk uses the image `mt_d100n250set1.rda` and `mt_d100n250set2.rda` to reproduce Table 1 of the main text. These outputs result by running the script `simu_mean.R`, available in the supplementary code archive.

```
setwd(res_dir)
load("mt_d100n250set1.rda")
out_mean1 <- output_fun(res)
rm(res)

load("mt_d100n250set2.rda")
out_mean2 <- output_fun(res)
rm(res)

attach(out_mean1$statshat_set)
res100m1 <- cbind(round(lambdahat, 3), round(avr_sel, 3), round(avrTPP*100, 1),
                  round(avrTNR*100, 1), round(avrFDP*100, 1))
```

```
detach(out_mean1$statshat_set)
attach(out_mean2$statshat_set)
res100m2 <- cbind(round(lambdahat, 3), round(avr_sel, 3), round(avrTPP*100, 1),
                  round(avrTNR*100, 1), round(avrFDP*100, 1))

resm <- cbind(res100m1, res100m2[, -1])

colnames(resm) <- c("lambda", "pstar", "TPP%", "TNP%", "FDP%",
                    "pstar", "TPP%", "TNP%", "FDP%")
rownames(resm) <- c()
resm
```

```
##        lambda  pstar  TPP%  TNP% FDP%  pstar  TPP%  TNP% FDP%
##  [1,]   0.750 52.157 100.0  63.8 51.8 47.902 100.0  69.5 45.4
##  [2,]   1.292 43.231 100.0  75.7 41.7 40.334  99.9  79.5 34.6
##  [3,]   2.225 35.057 100.0  86.6 28.2 32.801  99.9  89.6 19.3
##  [4,]   3.832 29.133 100.0  94.5 13.8 27.846  99.8  96.1  7.7
##  [5,]   6.599 26.038 100.0  98.6  3.9 25.475  99.5  99.2  1.8
##  [6,]  11.365 25.130 100.0  99.8  0.5 24.793  99.0  99.9  0.2
##  [7,]  19.574 24.994  99.9 100.0  0.0 24.448  97.8 100.0  0.0
##  [8,]  33.713 24.973  99.9 100.0  0.0 23.520  94.1 100.0  0.0
##  [9,]  58.062 24.950  99.8 100.0  0.0 20.664  82.7 100.0  0.0
## [10,] 100.000 24.846  99.4 100.0  0.0 15.937  63.7 100.0  0.0
```

2.2 Table 2

The following code chunk uses the image `prob_d100n250set1.rda` and `prob_d100n250set2.rda` to reproduce Table 2 of the main text. These outputs result by running the script `simu_prob.R`, available in the supplementary code archive.

```
setwd(res_dir)
load("prob_d100n250set1.rda")
out_prob1 <- output_fun(res)
rm(res)

load("prob_d100n250set2.rda")
out_prob2 <- output_fun(res)
rm(res)

attach(out_prob1$statshat_set)
res100p1 <- cbind(round(lambdahat, 3), round(avr_sel, 3), round(avrTPP*100, 1),
                  round(avrTNR*100, 1), round(avrFDP*100, 1))
detach(out_prob1$statshat_set)
attach(out_prob2$statshat_set)
res100p2 <- cbind(round(lambdahat, 3), round(avr_sel, 3), round(avrTPP*100, 1),
                  round(avrTNR*100, 1), round(avrFDP*100, 1))

resp <- cbind(res100p1, res100p2[, -1])

colnames(resp) <- c("lambda", "pstar", "TPP%", "TNP%", "FDP%",
                    "pstar", "TPP%", "TNP%", "FDP%")
rownames(resp) <- c()
resp
```

```
##       lambda  pstar TPP%  TNP% FDP%  pstar TPP%  TNP% FDP%
##  [1,]  0.200 69.930 99.9  40.0 64.2 53.053 99.4  62.4 51.0
##  [2,]  0.360 62.330 99.7  50.1 59.8 40.558 98.5  78.8 35.4
##  [3,]  0.649 53.742 99.5  61.5 53.5 32.046 96.4  89.4 21.9
##  [4,]  1.170 44.482 99.4  73.8 43.7 26.484 92.4  95.5 11.4
##  [5,]  2.107 35.487 99.4  85.8 29.5 22.766 86.6  98.5  4.4
##  [6,]  3.796 28.839 99.3  94.6 13.5 19.956 78.9  99.7  0.9
##  [7,]  6.840 25.565 99.0  98.9  3.0 17.354 69.2  99.9  0.1
##  [8,] 12.323 24.597 98.1  99.9  0.2 15.117 60.4 100.0  0.0
##  [9,] 22.202 23.264 93.1 100.0  0.0 13.096 52.3 100.0  0.0
## [10,] 40.000 19.006 76.0 100.0  0.0  9.660 38.5 100.0  0.0
```

2.3 Table 3

The following code chunk uses the image `cs_d15n250set1.rda` and `cs_d15n250set2.rda` to reproduce Table 3 of the main text. These outputs result by running the script `simu_corr.R`, available in the supplementary code archive.

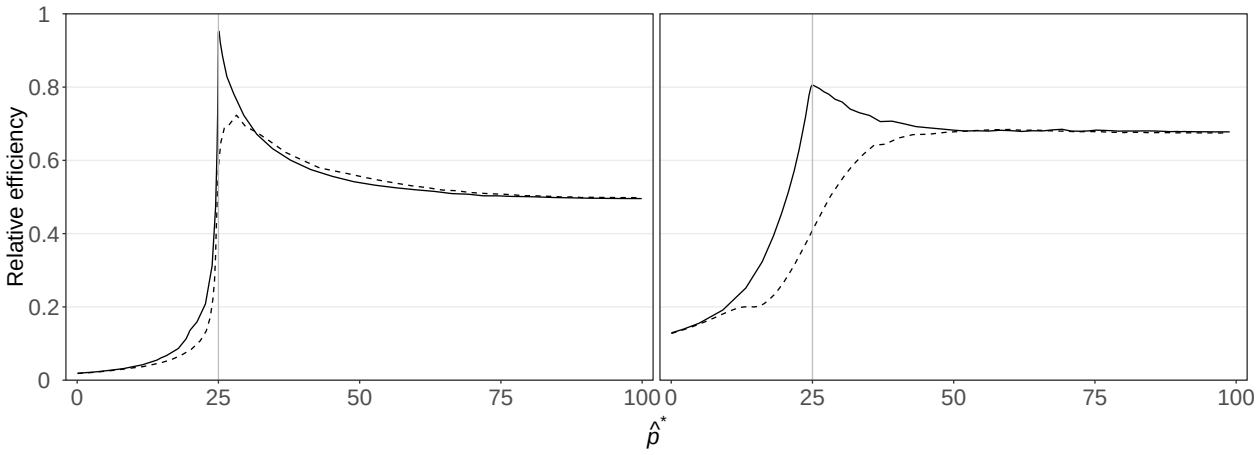
```
setwd(res_dir)
load("cs_d15n250set1.rda")
out_cs1 <- output_fun(res)
rm(res)

load("cs_d15n250set2.rda")
out_cs2 <- output_fun(res)
rm(res)

attach(out_cs1$statshat_set)
res100c1 <- cbind(round(lambdahat, 3), round(avr_sel, 3), round(avrTPP*100, 1),
                  round(avrTNR*100, 1), round(avrFDP*100, 1))
detach(out_cs1$statshat_set)
attach(out_cs2$statshat_set)
res100c2 <- cbind(round(lambdahat, 3), round(avr_sel, 3), round(avrTPP*100, 1),
                  round(avrTNR*100, 1), round(avrFDP*100, 1))

resc <- cbind(res100c1, res100c2[, -1])

colnames(resc) <- c("lambda", "pstar", "TPP%", "TNP%", "FDP%",
                    "pstar", "TPP%", "TNP%", "FDP%")
rownames(resc) <- c()
resc
```

```
##       lambda  pstar TPP%  TNP% FDP%  pstar TPP%  TNP% FDP%
##  [1,]  0.300 31.468 93.3  76.7 69.9 28.607 99.1  80.3 64.7
##  [2,]  0.426 24.960 91.3  83.3 62.7 23.041 98.4  86.1 56.3
##  [3,]  0.604 19.025 89.8  89.4 51.7 17.990 97.1  91.3 44.8
##  [4,]  0.857 14.494 90.1  94.2 36.4 13.997 94.9  95.3 30.7
##  [5,]  1.216 11.494 90.2  97.4 20.2 11.218 92.0  97.9 16.8
##  [6,]  1.726  9.772 89.5  99.1  7.7  9.428 87.5  99.3  6.6
##  [7,]  2.450  8.965 87.7  99.8  2.0  8.214 80.6  99.8  1.8
##  [8,]  3.476  8.453 84.2 100.0  0.3  7.090 70.6 100.0  0.4
##  [9,]  4.933  7.836 78.3 100.0  0.0  5.778 57.7 100.0  0.1
## [10,]  7.000  6.948 69.5 100.0  0.0  4.637 46.4 100.0  0.0
```

2.4 Figure 1

The code chunk below reproduces Figure 1.

```
re <- c(out_mean1$stats_set$re, out_mean2$stats_set$re, out_prob1$stats_set$re,
        out_prob2$stats_set$re)

sel <- c(out_mean1$stats_set$avr_sel, out_mean2$stats_set$avr_sel,
         out_prob1$stats_set$avr_sel,
         out_prob2$stats_set$avr_sel)

setting <- c(rep("no.cor", 100), rep("cor", 100), rep("no.cor", 100),
             rep("cor", 100))

scen <- c(rep("mean", 200), rep("prob", 200))

res <- data.frame(re, sel, setting, scen)

ggplot(res, aes(x = sel, y = re, group = setting)) +
  geom_line(aes(linetype = setting)) +
  scale_linetype_manual(values = c("dashed", "solid")) +
  geom_vline(aes(xintercept = 25), col = "grey") +
  labs(y = "Relative efficiency", x = expression(hat(italic(p))^list("*"))) +
  scale_y_continuous(breaks = seq(0, 1, length = 6), expand = c(0.0, 0.0),
                     labels = c("0", "0.2", "0.4", "0.6", "0.8", "1")) +
  expand_limits(y = c(0, 1)) +
  scale_x_continuous(breaks = seq(0, 100, length = 5), expand = c(0.02, 0.02),
                     labels = c("0", "25", "50", "75", "100")) +
  theme_bw() +
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.y = element_blank(),
        panel.grid.minor.x = element_blank(),
        strip.background = element_blank()) +
  facet_wrap(~ scen) +
  theme(legend.position = "none") +
  theme(strip.text.x = element_blank()) +
  theme(axis.text.x = element_text(size = 18, angle = 0, hjust = .5,
                                   vjust = .5, face = "plain"),
        axis.text.y = element_text(size = 18, angle = 0, hjust = 0.5,
                                   vjust = 0.5, face = "plain"),
        axis.title.x = element_text(size = 18, angle = 0, hjust = .5,
                                    vjust = 0, face = "plain"),
        axis.title.y = element_text(size = 18, angle = 90, hjust = .5,
                                    vjust = .5, face = "plain"))
```



References

R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.