De-bias CCA

Nilanjana laha

9/24/2021

Installation

Install from GitHub with:

```
# install.packages("devtools")
devtools::install_github("nilanjanalaha/de.bias.CCA")
```

```
## Downloading GitHub repo nilanjanalaha/de.bias.CCA@master
```

```
## Error in utils::download.file(url, path, method = method, quiet = quiet,  :
##   cannot open URL 'https://api.github.com/repos/nilanjanalaha/de.bias.CCA/tarball/master'
```

```
library(de.bias.cca)
```

Although de.bias.CCA installs `CVXR", it is recommended to install it if necessary, and load it prior to using`de.bias.CCA".

```
library(CVXR)
```

```
##
## Attaching package: 'CVXR'
```

```
## The following object is masked from 'package:stats':
##
##     power
```

Toy example

We generate p-variate and q-variate standard Gaussian vectors X and Y with covariance $\Sigma_{xy} = \rho\alpha\beta^T$. Thus the canonical correlation is ρ, and the canonical directions are α and β, respectively. We take $n = 500$, $p = 50$, $q = 50$. Also, we take the sparsity of α and β to be 10 and 25, respectively.

```
library(mvtnorm)
```

```
## Warning: package 'mvtnorm' was built under R version 3.6.2
```

```
#Simulate standard normal data matrix: first generate alpha and beta
p <- 50; q <- 50; al <- c(rep(1, 10), rep(0, 40));
be <- c(rep(0,25), rnorm(25,1))
#Normalize alpha and beta
al <- al/sqrt(sum(al^2))
be <- be/sqrt(sum(be^2))
n <- 300; rho <- 0.5
#Creating the covariance matrix
Sigma_mat <- function(p,q,al,be, rho)
{
  Sx <- diag(rep(1,p), p, p)
```

```
  Sy <- diag(rep(1,q), q, q)
  Sxy <- tcrossprod(crossprod(rho*Sx, outer(al, be)), Sy)
  Syx <- t(Sxy)
  rbind(cbind(Sx, Sxy), cbind(Syx, Sy))
}
truesigma <- Sigma_mat(p,q,al,be, rho)
#Simulating the data
Z <- mvtnorm::rmvnorm(n, sigma = truesigma)
x <- Z[,1:p]
y <- Z[,(p+1):(p+q)]
elements <- 1:p
nlC <- log(p+q)/n
```

For implementation, we estimate α and β using the SCCA estimators of Mai and Zhang (2019). To the end, we use the R function cca.mai, which is based on R function SCCA taken from the first author's website.

```
# Mai(2017)'s SCCA estimators
temp <- cca.mai(x,y)
ha <- temp[[1]]
hb <- temp[[2]]
```

Now we apply the de-bias procedure. To that end, we use the R function give_CCA. Type `?give_CCA" to learn about different tuning parameters and the values it return. In particular,`elements" contains some indexes among 1:(p+q), variance estimates of estimators with these indexes are returned. In the following example, we extract the variances only corresponding to $\widehat{\alpha}$.

```
# de-bias

temp <- give_CCA(ha, hb, x, y, elements=1:p)
```