Computer codes for "An Iterative Penalized Least Squares Approach to Sparse Canonical Correlation Analysis" by Qing Mai and Xin Zhang

Qing Mai

Department of Statistics, Florida State University, Tallahassee, FL, USA
mai@stat.fsu.edu
Xin Zhang
Department of Statistics, Florida State University, Tallahassee, FL, USA
henry@stat.fsu.edu

The codes in "SCCA-code.R" implement the SCCA (sparse canonical correlation analysis) method proposed in "An Iterative Penalized Least Squares Approach to Sparse Canonical Correlation Analysis" by Qing Mai (mai@stat.fsu.edu) and Xin Zhang (henry@stat.fsu.edu). To use "SCCA-code.R", please first make sure that the `R` package `glmnet` has been installed. If so, one can use the command

```
source("SCCA-code.R")
```

and proceed with SCCA. An example of applying SCCA on data generated according to Model 6 with $p = q = 200, n = 500$ is available in "example.R".

There are seven functions in "SCCA-code.R". Three functions, `SCCA`, `cv.SCCA.equal`, `init0` are used in "example.R". We describe the usage of these three functions as follows. Brief description of the other four functions are given in "SCCA-code.R" as comments.

1. The function `SCCA`:

   `SCCA` is the main function that produces estimates of the canonical pairs. It returns the estimated canonical pairs `alpha`, `beta`, the initial values `alpha.init`, `beta.init` and the number of iterations at convergence `n.iter.converge`.

   This function has the following arguments:

   `x`: An n by p data matrix.

   `y`: An n by q data matrix.

   `alpha.init`: The initial value for alpha; can be omitted if a value is given to the argument init.method.

beta.init: The initial value for beta; can be omitted if a value is given to the argument init.method.

lambda.alpha: The tuning parameter for alpha.

lambda.beta: The tuning parameter for beta.

niter: The number of maximum iterations; default is 100.

npairs: The number of pairs to be estimated; default is 1.

init.method: Methods of initialization. This argument is ignored if alpha.init and beta.init are specified. Options for init.method include "sparse","uniform","svd","random". The option "sparse" gives the thresholded initial value described in Section 2.3 of the paper. The option "uniform" first sets all the coefficients to be 1 and then rescale them to satisfy the length constraint. The option "svd" gives the directions from singular value decomposition. The option "random" gives random initial values that satisfy the length constraint.

alpha.current: If k canonical pairs have already be obtained and the user desires the k+1 pair, the first k alpha can be supplied to alpha.current to save computation time.

beta.current: If k canonical pairs have already be obtained and the user desires the k+1 pair, the first k beta can be supplied to alpha.current to save computation time.

standardize: Specify if data should be standardized before fitting SCCA; default is TRUE.

eps: Used in the stopping criterion; default is 1e-4.

2. The function cv.SCCA.equal:

   The function cv.SCCA.equal chooses the tuning parameter to be used in SCCA, under the assumption $\lambda_\alpha = \lambda_\beta$. If one wishes to use different tuning parameters for the two directions, the function cv.SCCA can be used. The best tuning parameter is returned as bestlambda. This function has the following arguments:

   x: An n by p data matrix.

   y: An n by q data matrix.

   lambda: Candidate of the tuning parameters.

   nfolds: The number of folds; default is 5.

   alpha.init: The initial value for alpha.

   beta.init: The initial value for beta.

   niter: The number of maximum iterations; default is 20.

   eps: Used in the stopping criterion; default is 1e-3.

3. The function init0:

   The function init0 finds the initial values for the first canonical pair, with the names alpha.init, beta.init.

   This function has the following arguments:

   sigma.YX.hat: The sample covariance between the two high-dimensional measurements.

`sigma.X.hat`: The sample covariance of the high-dimensional measurement `x`.

`sigma.Y.hat`: The sample covariance of the high-dimensional measurement `y`.

`init.method`: The same as that in the function `SCCA`.