Use of Smoothly Clipped Absolute Deviation (SCAD) Penalty on Sparse Canonical Correlation Analysis

Fan and Li 2001 [1] proposed a non-concave penalty function referred to as the smoothly clipped absolute deviation (SCAD) . The SCAD penalty is given by

$$p_{\lambda}^{SCAD}(\beta_j) = \begin{cases} \lambda|\beta_j| & \text{if } |\beta_j| \leq \lambda; \\ -\left(\frac{|\beta_j|^2 - 2a\lambda|\beta_j| + \lambda^2}{2(a-1)}\right) & \text{if } \lambda < |\beta_j| \leq a\lambda; \\ \frac{(a+1)\lambda^2}{2} & \text{if } |\beta_j| > a\lambda \end{cases}$$

This corresponds to quadratic spline function with knots at λ and $a\lambda$. The function is continuous and the first derivative for some $a > 2$ and $\beta > 0$ can be given by

$$p'_{\lambda}(\beta) = \lambda\{I(\beta \leq \lambda) + \frac{(a\lambda - \beta)_+}{(a-1)\lambda} I(\beta > \lambda)\}$$

The SCAD penalty is continuously differentiable on $(-\infty, 0) \bigcup (0, \infty)$ but singular at 0 with its derivatives zero outside the range $[-a\lambda, a\lambda]$. This results in small coefficients being set to zero, a few other coefficients being shrunk towards zero while retaining the large coefficients as they are. Thus, SCAD can produce sparse set of solution and approximately unbiased coefficients for large coefficients. The solution to the SCAD penalty can be given as

$$\hat{\beta}_j^{SCAD} = \begin{cases} (|\hat{\beta}_j| - \lambda)_+ sign(\hat{\beta}_j) & \text{if } |\hat{\beta}_j| < 2\lambda; \\ \{(a-1)\hat{\beta}_j - sign(\hat{\beta}_j)a\lambda\}/(a-2) & \text{if } 2\lambda < |\hat{\beta}_j| \leq a\lambda; \\ \hat{\beta}_j & \text{if } |\hat{\beta}_j| > a\lambda \end{cases}$$

This thresholding rule involves two unknown parameters λ and a. Theoretically, the best pair (λ, a) could be obtained using two dimensional grids search using some criteria like cross validation methods. However, such an implementation could be computationally expensive. Based on Bayesian statistical point of view and simulation studies, Fan and Li suggested $a = 3.7$ is a good choice for various problems. They further argued that the performance of various variable selection problems do not improve significantly with a selected by data driven methods. In this paper a was set to 3.7 with λ selected by cross validation method.

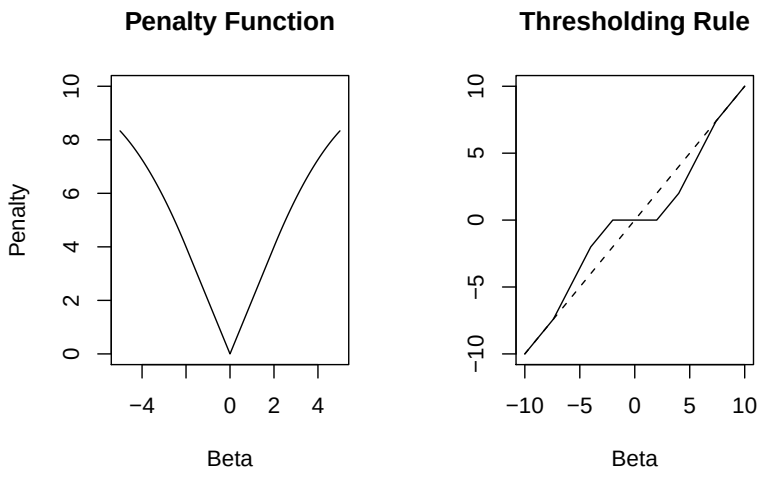

Figure 1: The plots showing the penalty function and the thresholding rule.

1 Estimation of the loading vectors

Parkhomenko et al. developed an iterative algorithm for estimating the loading vectors which is given as follows.

Update until convergence

1. Initialize u^i,
2. Normalize: u^i,
3. Apply penalty function on u, where update has the form $u^i \leftarrow$ penalized u^j
4. Normalize u^i

Do the samething for v too. Here u and v are left and right singular vectors respectively. For details look at [2].

R code

The SCAD penalty function is implemented in the third step of the algorithm mentioned above. R code for the SCAD penalty function is given as

```
# u.initial and v.initial are vectors of starting values
# lambda.u and lambda.v are preselected sparseness parameters
# lambda.u and lambda.v are estimated using cross validation

# Update left singular vector
```

```
#-------------------------------------------------
# Initialize
vx <-  k %*% v.initial  ## k is regularized correlation matrix

# Normalize
norm.vx <- as.numeric(sqrt(t(vx)%*%vx))
if(norm.vx==0) norm.vx <- 1
vx <- vx / norm.vx

# Implement SCAD Penalty for u here
names(vx) <- 1:length(vx) # Names of vx will be used to sort them
# Split to 3 pieces according to scad criteria
# First piece
vx1 <- vx[abs(vx)<=2*lambda.u]
u.new1 <- abs(vx1) - lambda.u
u.new1 <- (u.new1 + abs(u.new1))/2
u.new1 <- u.new1*sign(vx1)
# Second piece
vx2 <- vx[abs(vx)>2*lambda.u & abs(vx)<=3.7*lambda.u]
u.new2 <- (2.7*vx2-sign(vx2)*3.7*lambda.u)/1.7
# Third piece
vx3 <- vx[abs(vx)>3.7*lambda.u]
u.new3 <- vx3

u.new <- c(u.new1,u.new2,u.new3)
u.new <- u.new[names(vx)]   # Puts the data back to the original order
u.new <- as.numeric(u.new)  # Remove the names

# Normalize again
norm.u.new <- as.numeric(sqrt(t(u.new)%*%u.new))
if(norm.u.new==0) norm.u.new <- 1
u.new <- u.new / norm.u.new
# This completes updating left singular vector u

#####################################################################
# Update right singular vector
#-----------------------------------------------------------
# Initialize
ux <- t(k) %*% u.new

# Normalize
norm.ux <- as.numeric(sqrt(t(ux)%*%ux))
if(norm.ux==0) norm.ux <- 1
ux <- ux / norm.ux

# Implement Smoothly Clipped Absolute Deviation Penalty (SCAD) for v here
```

```
names(ux) <- 1:length(ux) # Names on ux will be used to sort them
# Split to 3 pieces according to scad criteria
# First piece
ux1 <- ux[abs(ux)<=2*lambda.v]
v.new1 <- abs(ux1) - lambda.v
v.new1 <- (v.new1 + abs(v.new1))/2
v.new1 <- v.new1*sign(ux1)
# Second piece
ux2 <- ux[abs(ux)>2*lambda.v & abs(ux)<=3.7*lambda.v]
v.new2 <- (2.7*ux2-sign(ux2)*3.7*lambda.v)/1.7
# Third piece
ux3 <- ux[abs(ux)>3.7*lambda.v]
v.new3 <- ux3

v.new <- c(v.new1,v.new2,v.new3)
v.new <- v.new[names(ux)]   # Puts the data back to the original order
v.new <- as.numeric(v.new)  # Remove the names

# Normalize again
norm.v.new <- as.numeric(sqrt(t(v.new)%*%v.new))
if(norm.v.new==0) norm.v.new <- 1
v.new <- v.new / norm.v.new
# This completes updating right singular vector v
```

References

[1] Fan Jianqing and Li Runze. Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties. *Journal of American Statistical Association*, 96:1348–1360, 2001.

[2] Parkhomenko Elena, Tritchler David, and Beyene Joseph. Sparse canonical correlation analysis with application to genomic data integration. *Statistical Applications in Genetics and Molecular Biology*, 8, 2009.