实用统计软件项目报告

晏若儒 PB17081535
骆霄龙 PB18151853

论文: A convex formulation for high-dimensional sparse sliced inverse regression

日期：2020 年 12 月 25 日

目录

摘　　要

本文主要是使用切片逆回归方法处理高维度情形下的稀疏回归问题，通过结合交替方向乘子算法来迭代求解中心降维子空间。对估计误差的上界给出了理论证明并进行了模拟实验。基于我们选择的文章，我们实现了文中的切片逆回归结合交替方向乘子的算法，并复现了文中的实验，取得了比较一致的结果。同时我们将文中的算法在不同模型下与 Lasso，Ridge，ElasticNet 这三个方法进行了比较，来考察不同方法的优劣性。

(代码链接:https://github.com/lxl213)

关键词: 凸优化问题，降维，切片逆回归，交替方向乘子算法,R 语言，不同模型下优化方法比较

1 文章内容概述

1.1 问题背景

考虑在高维情形下的回归问题 (样本数量 n 小于变量数目 d): $y = f(x)$, 当变量数目 d 很大时，我们希望能对变量进行降维，找出显著的变量并尽可能少的损失信息。我们考虑在满足以下条件下的降维问题:

$$y \perp\!\!\!\perp x \mid \left(\beta_1^{\mathrm{T}} x, \ldots, \beta_K^{\mathrm{T}} x\right) \tag{1}$$

其中 $K < min(n, d), \beta_1, \cdots, \beta_K \in \mathbb{R}^d$ $\perp\!\!\!\perp$ 表示独立 . 所以方程 (1) 表示 y 能够被 x 的 K 个线性组合所决定。我们称 $\beta_1, \cdots, \beta_K$ 为充分降维方向 (sufficient dimension reduction directions), $\mathcal{V} = span\beta_1, \cdots, \beta_K$, 由于 $\mathcal{V}$ 不唯一，我们称 $\mathcal{V}_{y|x} = \cap\mathcal{V}$ 为中心降维子空间 (central dimension reduction subspaces)。这篇文章中使用基于切片逆回归法 (sliced inverse regression) 的改进方法来估计 d>n 时的中心降维子空间 $\mathcal{V}_{y|x}$。

1.2 主要工作

- 对论文理论部分进行了较为完善的总结，给出了文章提出方法的主要思路和具体解法
- 对文章算法进行了复现，并进行实验取得了与文章较为一致的成果
- 实现了在文章中三个模型下的其他回归算法，并做图与文章算法进行比较与分析

1.3 主要理论

考虑上述回归模型的一般形式:

$$y = f\left(\beta_1^{\mathrm{T}} x, \ldots, \beta_K^{\mathrm{T}} x, \epsilon\right) \tag{2}$$

在使用切片逆回归法 (下面简称 SIR) 时一个需要一个线性条件, 对 $\forall a \in \mathbb{R}^d$:

$$E\left(a^{\mathrm{T}} x \mid \beta_1^{\mathrm{T}} x, \ldots, \beta_K^{\mathrm{T}} x\right) = b_0 + b_1 \beta_1^{\mathrm{T}} x + \cdots + b_K \beta_K^{\mathrm{T}} x \tag{3}$$

在线性条件 (3) 下, 由 (Li, 1991[1], Theorem 3.1). 问题转化为寻找 $V \in \mathbb{R}^{d \times K}$ 使得:

$$\hat{\Sigma}_{E(x|y)}V = \hat{\Sigma}_x V\Lambda \tag{4}$$

其中 $\hat{\Sigma}_{E(x|y)}$是$\Sigma_{E(x|y)}$ 对估计，$V^{\mathrm{T}}\hat{\Sigma}_x V = I_K$,$\Lambda = \mathrm{diag}\,(\lambda_1, \ldots, \lambda_K) \in \mathbb{R}^{K\times K}$, 由上述方程可知问题又等价于:

$$\underset{V\in\mathbb{R}^{d\times K}}{\text{minimize}} - \mathrm{tr}\left\{V^{\mathrm{T}}\hat{\Sigma}_{E(x|y)}V\right\} \text{ subject to } V^{\mathrm{T}}\hat{\Sigma}_x V = I_K \tag{5}$$

进一步的，令 $\Pi = VV^{\mathrm{T}}$,$\mathcal{B} = \left\{\hat{\Sigma}_x^{1/2}\Pi\hat{\Sigma}_x^{1/2} : V^{\mathrm{T}}\hat{\Sigma}_x V = I_K\right\}$,$\mathcal{M}$ 为 $d\times d$ 维的半正定矩阵, 上面问题可写为:

$$\underset{\Pi\in\mathcal{M}}{\text{minimize}} - \mathrm{tr}\left\{\hat{\Sigma}_{E(x|y)}\Pi\right\} \text{ subject to } \hat{\Sigma}_x^{1/2}\Pi\hat{\Sigma}_x^{1/2} \in \mathcal{B} \tag{6}$$

注意到 (8) 是一个非凸优化问题，我们使用凸松弛来求其近似解:

$$\underset{\Pi\in\mathcal{M}}{\text{minimize}} - \mathrm{tr}\left\{\hat{\Sigma}_{E(x|y)}\Pi\right\} \text{ subject to } \left\|\hat{\Sigma}_x^{1/2}\Pi\Sigma_x^{1/2}\right\|_* \leqslant K, \left\|\hat{\Sigma}_x^{1/2}\Pi\hat{\Sigma}_x^{1/2}\right\|_{\mathrm{sp}} \leqslant 1 \tag{7}$$

其中:

$$\left\|\hat{\Sigma}_x^{1/2}\Pi\Sigma_x^{1/2}\right\|_* = \mathrm{trace}\left(\hat{\Sigma}_x^{1/2}\Pi\Sigma_x^{1/2}\right)$$

$$\left\|\hat{\Sigma}_x^{1/2}\Pi\Sigma_x^{1/2}\right\|_{\mathrm{sp}} = \sup_{v:v^{\mathrm{T}}v=1}\left\{\sum_{j=1}^{d}\left(\hat{\Sigma}_x^{1/2}\Pi\Sigma_x^{1/2}v\right)_j^2\right\}^{1/2}$$

现在我们的目标转化为求 Π 满足 (7)。注意到，若 $\Pi_{jj} = 0$，则有 $V_{jk} = 0, \quad k \in (1, \ldots, K)$，即 Π 的第 j 行为 0，表示第 j 个变量没有被选择，故定义:$s = |\,\mathrm{supp}\{\mathrm{diag}(\Pi)\}|$。为了使我们求得的 Π 起到变量选择的作用, 即行具有" 稀疏" 性。我们加上 L_1 正则项，使得优化问题 (7) 变为:

$$\underset{\Pi\in\mathcal{M}}{\text{minimize}} - \mathrm{tr}\left\{\hat{\Sigma}_{E(x|y)}\Pi\right\} + \rho\|\Pi\|_1 \text{ subject to } \left\|\hat{\Sigma}_x^{1/2}\Pi\hat{\Sigma}_x^{1/2}\right\|_* \leqslant K, \left\|\hat{\Sigma}_x^{1/2}\Pi\hat{\Sigma}_x^{1/2}\right\|_{\mathrm{sp}} \leqslant 1 \tag{8}$$

其中:$\|\Pi\|_1 = \sum_{i,j}\left|\Pi_{ij}\right|$, 且 ρ 为正则项系数。

下面介绍文章算法来求解优化问题 (8)

2 算法描述及代码实现

算法背景:要解决优化问题 (8)，注意到我们要优化的 Π 在限制条件中，所以这里我们使用经典的 **linearized alternating direction method of multipliers algorithm**[2] [1] 来解决

相关收敛定理:在这个方法下，我们记 $\mathcal{V}$ 和 $\hat{\mathcal{V}}$ 分别为真实的和估计的中心降为子空间。**定义:**P_Π 为 $P_{\hat{\Pi}}\mathcal{V}$ 和 $\hat{\mathcal{V}}$ 的投影矩阵，我们定义 $\mathcal{V}$ 和 $\hat{\mathcal{V}}$ 的距离为: $D(\mathcal{V}, \hat{\mathcal{V}}) = \left\|P_\Pi - P_{\hat{\Pi}}\right\|_{\mathrm{F}}$

定理 2.1. *记 $\mathcal{V}$ 和 $\hat{\mathcal{V}}$ 分别为真实的和估计的中心降为子空间。设存在 C 使得 $n > Cs^2\log d/\lambda_K^2$, 其中 λ_K 是 $\left\{\Sigma_{E(x|y)}, \Sigma_x\right\}$ 的第 Kth 广义特征值, 设 $\lambda_K K^2 < s\log d$. 且令 $\rho \geqslant C_1(\log d/n)^{1/2}$，其中 C_1 也为一正常数, 那么在文中给出条件下，我们有:*

$$D(\mathcal{V}, \hat{\mathcal{V}}) \leqslant C_2 s(\log d/n)^{1/2}/\lambda_K$$

在实验中，我们依照文章取 $C_2 = 2$，成果取得了与文章类似的结果，从实验的角度验证了这个定理。

[1] See the reference paper for the details

2.1 算法描述

Algorithm 1: Linearized alternating direction of method of multipliers algorithmn

Input: $\hat{\Sigma}_x, \hat{\Sigma}_{E(x|y)}$, tuning parameter ρ, rank constraint K, the L-ADMM parameters v, tolerance level $\epsilon > 0$, and $\tau = 4v\lambda_{\max}^2\left(\hat{\Sigma}_x\right)$,

Initializations: primal variables $\Pi^{(0)} = I_d, H^{(0)} = I_d$, and dual variable $\Gamma^{(0)} = 0.$;

while $\left\|\Pi^{(t)} - \Pi^{(t-1)}\right\|_{\mathrm{F}} \geqslant \epsilon$. **do**

a. $\Pi^{(t+1)} = \mathrm{Soft}\left[\Pi^{(t)} + \hat{\Sigma}_{E(x|y)}/\tau - v\left\{\hat{\Sigma}_x\Pi^{(t)}\hat{\Sigma}_x - \hat{\Sigma}_x^{1/2}\left(H^{(t)} - \Gamma^{(t)}\right)\hat{\Sigma}_x^{1/2}\right\}/\tau, \rho/\tau\right]$

where Soft denotes the soft-thresholding operator, applied elementwise to a matrix, $\mathrm{Soft}\left(A_{ij}, b\right) = \mathrm{sign}\left(A_{ij}\right)\max\left(\left|A_{ij}\right| - b, 0\right)$;

b. $H^{(t+1)} = \sum_{j=1}^{d}\min\left\{1, \max\left(\omega_j - \gamma^*, 0\right)\right\}u_j u_j^{\mathrm{T}}$, where $\sum_{j=1}^{d}\omega_j u_j u_j^{\mathrm{T}}$ is the singular value decomposition of $\Gamma^{(t)} + \hat{\Sigma}_x^{1/2}\Pi^{(t+1)}\hat{\Sigma}_x^{1/2}$, and

$$\gamma^* = \underset{\gamma>0}{\mathrm{argmin}}\gamma, \qquad \text{subject to } \sum_{j=1}^{d}\min\left\{1, \max\left(\omega_j - \gamma, 0\right)\right\} \leqslant K$$

c. $\Gamma^{(t+1)} = \Gamma^{(t)} + \hat{\Sigma}_r^{1/2}\Pi^{(t+1)}\hat{\Sigma}_r^{1/2} - H^{(t+1)}$

end

2.2 重要函数和功能介绍

鉴于代码长度，每个函数仅展示其核心部分，完整代码见链接:[2]

calsigmafit:

1. Estimating the conditional covariance
2. Using cov(E[X|Y])=cov(x)-E[cov(X|Y)] by estimating E[cov(X|Y)]
3. according to $\tilde{T} = \frac{1}{H}\sum_{h=1}^{H}\left\{\frac{1}{n_h}\sum_{i\in S_h}\left(x_i - \bar{x}_{S_h}\right)\left(x_i - \bar{x}_{S_h}\right)^{\mathrm{T}}\right\}$by computing weighted average of the sample covariance

```
f <- matrix(0,n,nslice)
for(k1 in 1:(nslice-1)){
  for(k2 in 1:nslice){
    if (k1==k2){
      f[indexy[[k1]],k2] <- 1 - nindexy[[k2]]/n }
    if (k1!=k2){
      f[indexy[[k1]],k2] <- -nindexy[[k2]]/n} } }
for(k in 1:nslice){
  f[indexy[[nslice]],k] <- -nindexy[[k]]/n }
bigF <- f%*%solve(t(f)%*%f)%*%t(f)
Sigmafit <- t(X)%*%bigF%*%X/(n)
return(Sigmafit)
```

[2]https://github.com/lxl213

ssir:

1. LADMM to solve convex optimazation problem
2. input:cov-xy， cov-x,lambda,epsilon
3. output:a list including object variable pi,H and dual variable gamma

```
sqcovx <- eigencovx$vectors%*%sqrt(diag(pmax(eigencovx$values,0)))%*%t(eigencovx$vectors)
tau <- 4*nu*eigencovx$values[1]^2
while( criteria  > epsilon  && i <= maxiter){
    Pi  <- newPi(covx,sqcovx,covxy,H,Gamma,nu,lambda,Pi,tau)
    H <- newH(sqcovx,Gamma,nu,Pi,K)
    Gamma <- Gamma + sqcovx%*%Pi%*%sqcovx-H
     criteria  <- sqrt(sum((Pi-oldPi)^2))
    oldPi  <- Pi
    i  <- i+1 }
    return( list (Pi=Pi,H=H,Gamma=Gamma,iteration=i,convergence=criteria))
```

Soft , newPi and newH:

1. Soft-thresholding Operator, Update Pi and H

predictpfc:

1. cross-validation to select tuning parameter K and lambda
2. input:pfcobject:the object by training set; K:constraint rank, y ,X
3. output:a vector of prediction y of ytest

```
 if (max(temp$values)<0.01){
   return(rep(mean(y),nrow(Xnew))). }
 temp <- temp$vectors[,1:K]
 RhatX <- X%*%temp
 Xnew <- as.list(data.frame(t(Xnew)))
 predicty  <- function(x){
   temp2 <- x%*%temp
   residual  <- t(t(RhatX)-as.vector(temp2)) #residual in each row.
   weights <- exp(-0.5*apply((residual)^2,1,sum))
   weights <- weights/sum(weights)
   return(sum(weights*y)). }
 yhat <- unlist(lapply(Xnew,predicty))
 return(yhat)  }
```

ssir.cv:

1. Cross-Validation to select the right parameters
2. input:X,y,ks,lambda,nfold,nslice
3. output:a list with elements of a matrix including

```
for(K in Ks){
        initH = initPi = diag(1,p,p)
        initGamma = diag(0,p,p)
        tmp <- 1
        for(lambda in lambdas){
          res <- ssir(Sigmafit,Sigmax,lambda,K,epsilon=5e-04,maxiter=1000,init=TRUE,initPi=initPi,
              initH=initH,initGamma=initGamma,trace=FALSE)
           initPi = res$Pi
           initH = res$H
           initGamma = res$Gamma
           yhat <- predictpfc(res,K,ytrain,Xtrain,Xtest)
                cv.error[[K]][j,tmp] <- sum((ytest-yhat)^2) }
           tmp <- tmp + 1 }
```

obseX:

1. Generate AR-1 type covariance for X which has the required distribution

```
Sigma <- matrix(0.5,p,p)
tmpmat <- matrix(0,p,p)
for(i in 1:(p-1)){
  tmpmat[i,i:p] <- c(0:(length(i:p)-1)) }
tmpmat = tmpmat+t(tmpmat)
Sigma <- Sigma^tmpmat
X <- mvrnorm(n=n,mu=rep(0,p),Sigma=Sigma)
return(X)
```

3 文章实验模拟结果与扩展

3.1 实验简介:

本文主要考虑了在三个不同的模型下检验算法的表现效果。其中一些主要**背景设置**如下:

- 数据 x 服从多元正态分布 $N_d(0,\Sigma_x)$，其协方差矩阵为:$(\Sigma_x)_{ij} = 0 \cdot 5^{|i-j|}$, $1 \leqslant i,j \leqslant d$
- 误差项 $\epsilon \sim N(0,1)$
- 初始迭代: $\Pi^{(0)} = I_d, H^{(0)} = I_d, \Gamma^{(0)} = 0$

三个实验模型:

模型 1: 简单线性回归模型:

$$y = (x_1 + x_2 + x_3)/3^{1/2} + 2\epsilon$$

其中心降维子空间: $\beta = (1_3, 0_{d-3})^{\mathrm{T}}$ and $K = 1$

模型 2: 简单非线性回归模型:

$$y = 1 + \exp\left\{(x_1 + x_2 + x_3)\,/3^{1/2}\right\} + \epsilon$$

其中心降维子空间: $\beta = (1_3, 0_{d-3})^{\mathrm{T}}$ and $K = 1$

模型 3: 非线性回归模型:

$$y = \frac{x_1 + x_2 + x_3}{0.5 + (x_4 + x_5 + 1 \cdot 5)^2} + 0 \cdot 1\epsilon$$

其中心降维子空间: $\beta_1 = (1_3, 0_{d-3})^{\mathrm{T}}, \beta_2 = (0_3, 1_2, 0_{d-5})^{\mathrm{T}}$, and $K = 2$

3.2 实验 1: 论文三个模型下的实验效果 (论文复现部分)

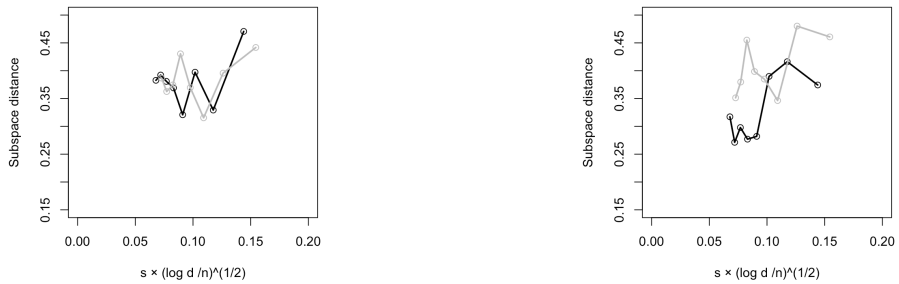
论文结果模拟 1:[3] 实验结果普遍较论文中差一点，原因应为 λ 不如论文中选取那么精细 (CV 选取合适 λ 计算量过大)

表 1: 不同 setting 下的实验结果. 前两个 setting 固定 $\lambda = 0.45$ ，Setting3 固定 $\lambda = 0.20$ 。取 10 次结果平均;

Result	n = 100 and d = 150			n = 200 and d = 150		
	Setting 1	Setting 2	Setting 3	Setting 1	Setting 2	Setting 3
TPR	94.3	91.2	89.7	96.0	93.7	94.6
FPR	4.7	5.1	6.4	4.1	3.9	4.8
corr	85.2	81.8	77.3	89.5	88.4	80.3

TPR: 预测正确/总正样本数,FPR: 预测错误/总正负本数，corr: 相关系数绝对值

论文结果模拟 2: 子空间距离: 实现 setting 1 和 setting 2 下的 (10 次实验结果平均):



(a) Setting 1 case (b) Setting 2 case

图 1: 子空间距离结果, 通过改变 n 得到图中不同的点，其中黑线为 d=100，灰线为 d=200 情形

[3] 由于论文中提到的其他三篇相关文献都没有公开 (需要单独购买). 且方法也和文中提出方法有所差异. 故本次实验未进行复现

3.3 实验 2: 论文三个模型下的其他方法比较 (实验扩展部分)

关于 λ 的选择: 注意到 λ 是我们需要预先选取并通过 Cross-Validation 来选取，由于 Cross-Validation 是非常耗时的，所以预先选取合适的 λ 是能有效提升计算速度的。可以看到在各种 K 下，0.3-0.5 之间的阶段的 λ 都能达到有一个比较快的迭代速度和稳定性。而过小或者过大的 λ 会导致需要迭代很多次才能收敛或者直接导致给出偏差极大结果。

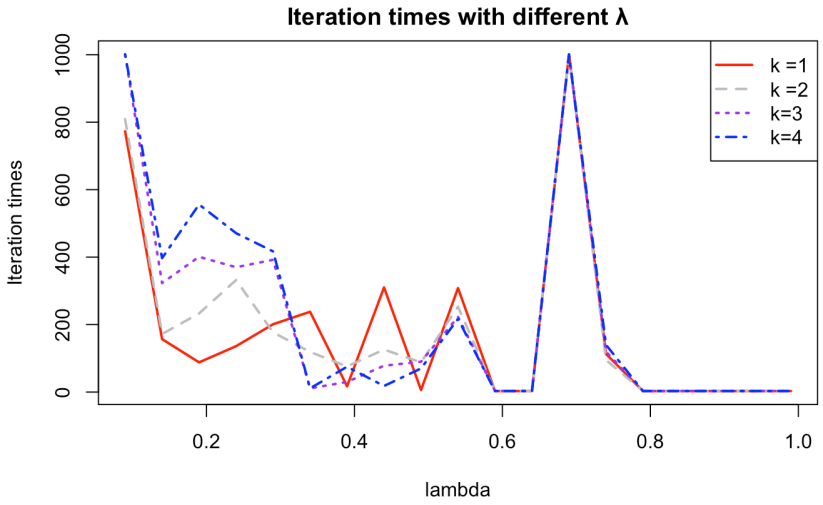


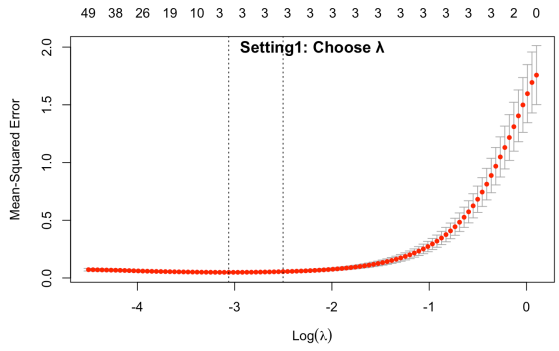
图 2: λ 选取与迭代次数的关系

模型 1 下的其他算法表现: 这里我们使用了 Lasso, Ridge and ElasticNet Regression 方法在其他设置与 SSIR 方法一样的情况下进行对比。其中 ElasticNet Regression 为一种同时结合 L_1, L_2 正则化的方法: $\min_\theta \frac{1}{2m}\left[\sum_{i=1}^{m}\left(\theta^T x^{(i)} - y^{(i)}\right)^2 + \lambda_1 \sum_{j=1}^{n} \|\theta\| + \lambda_2 \sum_{j=1}^{n} \|\theta\|_2^2\right]$

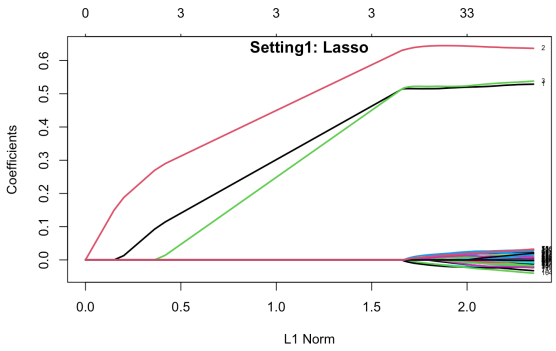
从图 2(下一页) 可以看出 Lasso 和 ElasticNet Regression 都正确的选出了参数，而 Ridge 则有较大误差。这与 L_1 正则可产生选择参数效果密切相关。

模型 2 和 3 下的其他算法表现: 由图三 (下一页) 可以看出在 setting 2 和 3 的非线性情况下 Ridge Regression 取得了相对比较好的结果. 而 Lasso 和 Elnet 表现都不佳。故对于不同模型应选择合适的回归方法，如这三个方法应主要使用于线性模型中。而对于非线性模型，应使用如 SSIR 等更加合适的方法。

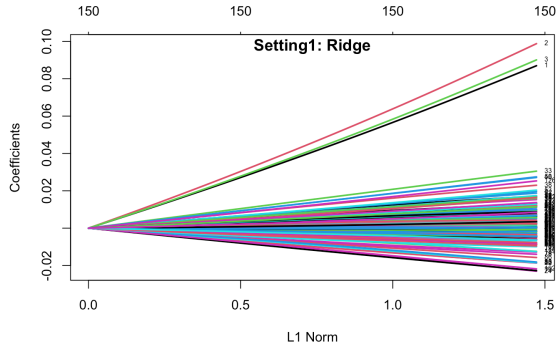
经实验我们发现，SSIR 方法能非常好的在非线性模型下选出显著的变量，即该方法表现受到模型线性/非线性影响较小，适用性广，但其缺点是迭代求解以及要选择合适参数（参数对结果影响很大），计算速度较慢。所以可在非线性模型情形下在选择使用此法求解得到比较好的结果。



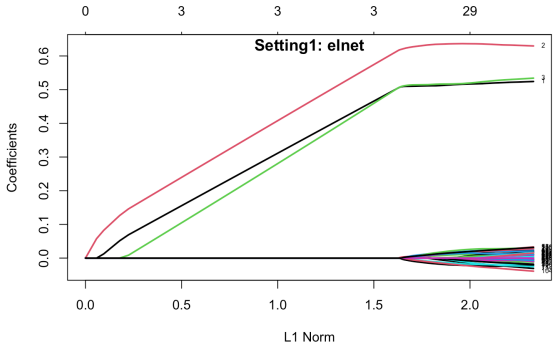
(a) CV Selection



(b) Lasso Regression

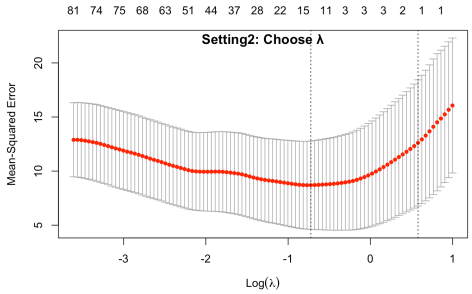


(c) Ridge Regression

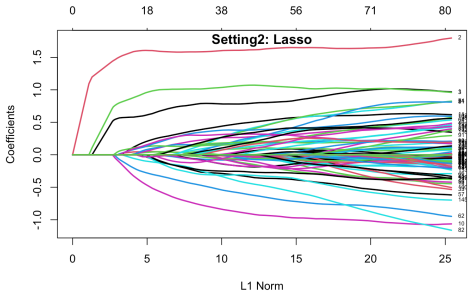


(d) ElasticNet Regression

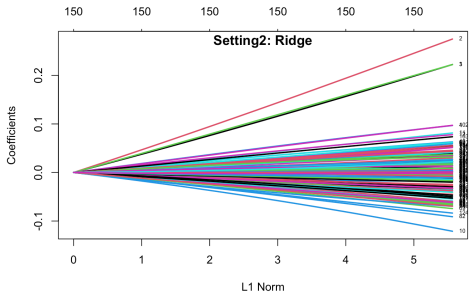
图 3: Setting 1 下 Lasso, Ridge and ElasticNet Regression 的表现



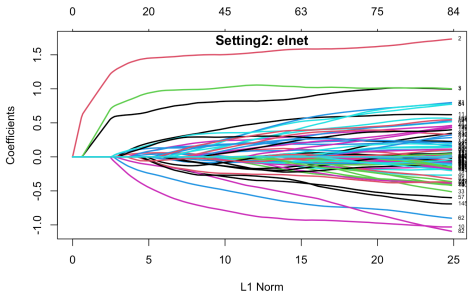
(a) CV Selection



(b) Lasso Regression

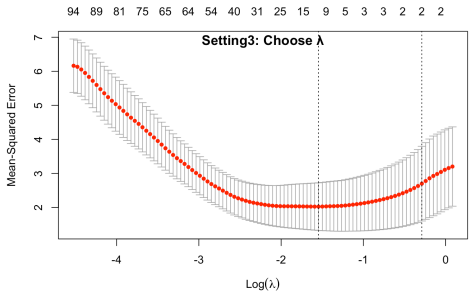


(c) Ridge Regression

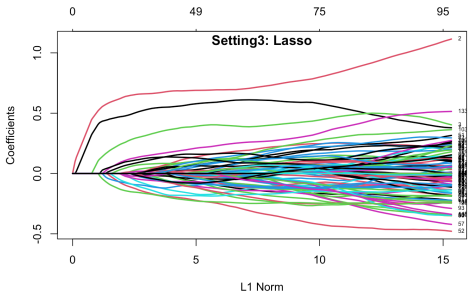


(d) ElasticNet Regression

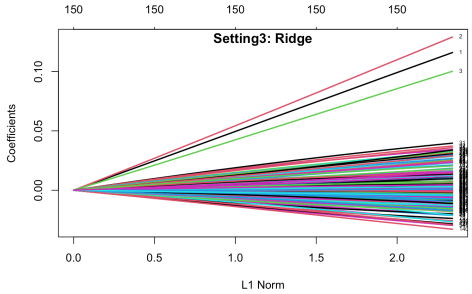
图 4: Setting 2 下 Lasso, Ridge and ElasticNet Regression 的表现



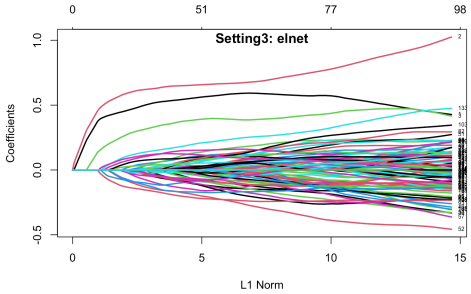
(a) CV Selection



(b) Lasso Regression



(c) Ridge Regression



(d) ElasticNet Regression

图 5: Setting 3 下 Lasso, Ridge and ElasticNet Regression 的表现

4 总结与后续讨论

项目总结:

- 通过一次完整的阅读和复现过程对整篇文献的理论和实验部分都有了较深刻的理解。
- 同时通过复现算法与不断的实验与调试最终得到结果让我们的理论和代码能力大大提高。
- 最后我们进一步的通过在模型下实现 Lasso，Ridge 和 Elastic Net 回归方法以更深的理解不同方法的优劣和在处理实际问题时如何选择方法来获得理想的结果。

可以继续优化的地方:

- 考虑到 cross-validation 是计算量的主要来源部分，所以对 K 值的估计也很大程度影响计算速度和最后的结果。能否有好的方法给出一个 K 的大致范围是可以考虑的问题。
- 方法在更加复杂的模型下是否仍能有比较好的稳定性？以及在更高维度 ($p > 10^4$) 的情形下表现如何，都是值得考虑和研究的问题。

4.1 任务分工:

鉴于各自完成的工作量，我们一致认为两人应平分在这次 Project 中的贡献。

1. 骆霄龙: 主要负责实验部分以及实验扩展部分代码，撰写项目报告。
2. 晏若儒: 主要负责复现论文方法，并进行代码优化，展示 PPT 制作以及 Presentation。

参考文献

[1] LI K C. Sliced inverse regression for dimension reduction[J/OL]. Journal of the American Statistical Association, 1991, 86(414):316-327. http://www.jstor.org/stable/2290563.

[2] ZHANG X, BURGER M, OSHER S. A unified primal-dual algorithm framework based on bregman iteration[J]. Journal of Scientific Computing, 2011, 46(1):20-46.

[3] TAN K M, WANG Z, ZHANG T, et al. A convex formulation for high-dimensional sparse sliced inverse regression[J]. Biometrika, 2018, 105(4):769-782.

[4] 朱利平, 於州. 切片逆回归的样条逼近[J]. 中国科学 A 辑, 2007.