Penalized Likelihood Estimation of Generalized Linear Models ——SCAD Estimations of Logistic and Poisson Models

Chen Zhiyuan
School of Statistics, Beijing Normal University

August 2022

目录

摘要

本文主要介绍了广义线性模型的惩罚似然估计，它是线性模型正则化的推广。在首先推导了线性模型的惩罚似然估计和广义线性模型的极大似然估计作为铺垫后，可以非常自然地给出广义线性模型的惩罚似然估计的方法论，算法和程序。

然后，以广义线性模型中最典型的 Poisson 回归和 Logistic 回归为例，模拟了 7 种不同协方差矩阵产生数据的模型使用 SCAD 作为惩罚函数的参数估计。文中比较了自行编写的 LQA 和 MM 算法以及 glmnet 包中的 Lasso 估计和 SIS 包中的 SCAD 估计的效果。并按照 BIC 准则进行了调节参数 λ 的选择。

1 线性模型的惩罚似然估计

定义惩罚最小二乘函数为：

$$Q(\beta) = \frac{1}{2n}\|Y - X\beta\|_2 + \sum_{j=1}^{p} p_\lambda(|\beta_j|)$$

其中 $p_\lambda(|\beta_j|)$ 为惩罚项。惩罚似然估计为使惩罚最小二乘函数达到最小值的点：

$$\hat{\beta} = argmin_\beta\ Q(\beta)$$

但常见的惩罚函数如 Lasso，SCAD 等有些是不可导和非凸的，无法直接优化。常见的算法处理是先近似惩罚项，再用牛顿法求解。

1.1 LQA 算法

正如其名 “Local Quadratic Approximation” 所言，LQA 算法的核心是对惩罚项作局部二次逼近，使惩罚项可导：

$$p_\lambda(|\beta_j|) \approx p_\lambda(|\beta_j^{(0)}|) + \frac{1}{2}\frac{p'_\lambda(|\beta_j^{(0)}|)}{|\beta_j^{(0)}|}(\beta_j^2 - \beta_j^{(0)2})$$

$$\begin{aligned}[p_\lambda(|\beta_j|)]' &= p'_\lambda(|\beta_j|)sgn(\beta_j) \\ &\approx \frac{p'_\lambda(|\beta_j^{(0)}|)}{|\beta_j^{(0)}|}\beta_j\end{aligned}$$

$Q(\beta)$ 的最小化变成一个凸优化问题：

$$
\begin{aligned}
Q'(\beta) &= \frac{1}{n}(X'X\beta - X'Y) + \begin{bmatrix} [p_\lambda(|\beta_1|)]' \\ \vdots \\ [p_\lambda(|\beta_p|)]' \end{bmatrix} \\
&= \frac{1}{n}(X'X\beta - X'Y) + \begin{bmatrix} \frac{p'_\lambda(|\beta_1^{(0)}|)}{|\beta_1^{(0)}|}\beta_1 \\ \vdots \\ \frac{p'_\lambda(|\beta_p^{(0)}|)}{|\beta_p^{(0)}|}\beta_p \end{bmatrix} \\
&= \frac{1}{n}(X'X\beta - X'Y) + \begin{bmatrix} \frac{p'_\lambda(|\beta_1^{(0)}|)}{|\beta_1^{(0)}|} & & & \\ & \frac{p'_\lambda(|\beta_2^{(0)}|)}{|\beta_2^{(0)}|} & & \\ & & \ddots & \\ & & & \frac{p'_\lambda(|\beta_p^{(0)}|)}{|\beta_p^{(0)}|} \end{bmatrix}\beta \\
&= \frac{1}{n}(X'X\beta - X'Y) + \Sigma_\lambda\beta \\
Q''(\beta) &= \frac{1}{n}X'X + \Sigma_\lambda \\
\Sigma_\lambda &= diag\{\frac{p'_\lambda(|\beta_1^{(0)}|)}{|\beta_1^{(0)}|}, ..., \frac{p'_\lambda(|\beta_p^{(0)}|)}{|\beta_p^{(0)}|}\}
\end{aligned}
$$

牛顿法迭代公式为：

$$
\begin{aligned}
\beta^{(n+1)} &= \beta^{(n)} - [Q''(\beta^{(n)})]^{-1} \cdot Q'[\beta^{(n)}] \\
&= \beta^{(n)} - [\frac{1}{n}X'X + \Sigma_\lambda]^{-1} \cdot [(\frac{1}{n}X'X + \Sigma_\lambda)\beta^{(n)} - \frac{1}{n}X'Y] \\
&= (X'X + n\Sigma_\lambda)^{-1}X'Y \\
\Sigma_\lambda(\beta^{(n)}) &= diag\{\frac{p'_\lambda(|\beta_1^{(n)}|)}{|\beta_1^{(n)}|}, ..., \frac{p'_\lambda(|\beta_p^{(n)}|)}{|\beta_p^{(n)}|}\}
\end{aligned}
$$

LQA 算法要求给定一个阈值 η，每次迭代后将 $|\beta_j| < \eta$ 的分量压缩为 0，并不再回到迭代过程中。这样实际会造成比较大的误差，在编程中也不方便。例如：不改变向量维数的话，0 没有办法回到 Σ_λ 对角线元素的分母上。

1.2 MM 算法

为了解决 LQA 算法中的问题，改进算法 MM 采用了带扰动项的局部二次逼近：

$$
p_\lambda(|\beta_j|) \approx p_\lambda(|\beta_j^{(0)}|) + \frac{1}{2}\frac{p'_\lambda(|\beta_j^{(0)}|)}{|\beta_j^{(0)}| + \tau}(\beta_j^2 - \beta_j^{(0)2})
$$

τ 是一个很小的非负参数，其值一般选为 Li 和 Hunter（2005）推荐的 $\tau = 10^{-8}$。

MM 算法的原始思想其实和 EM 算法相似，即在原目标函数不易优化时，用构造函数的优化结果去近似：每次迭代找到原目标函数的一个上（下）界函数，并求上（下）界函数的最小（大）值。

Li 和 Hunter (2005) 已经证明了损失函数 (或极大似然函数) 加上扰动二次逼近后惩罚项的和 $S_\tau(\beta)$ 是原 $Q(\beta)$ 的一个好的构造函数——尽管他们也建议在每一步迭代中验证单调性是否成立。

方便起见，直接沿用 $S_\tau(\beta)$ 作为牛顿法的优化目标，并不妨仍记为 $Q(\beta)$。这样 MM 算法的主体基本与 LQA 相同:

$$\beta^{(n+1)} = (X'X + n\Sigma_\lambda)^{-1}X'Y$$
$$\Sigma_\lambda(\beta^{(n)}) = diag\{\frac{p'_\lambda(|\beta_1^{(n)}|)}{|\beta_1^{(n)}|+\tau}, ..., \frac{p'_\lambda(|\beta_p^{(n)}|)}{|\beta_p^{(n)}|+\tau}\}$$

2 用牛顿法求解广义线性模型

线性模型的惩罚似然估计算法比较好推导，因为无论 L_2 的损失函数还是二次逼近后的惩罚项都可以直接求导。但是广义线性模型的极大似然估计直接求导比较麻烦，下面借助指数分布族给出牛顿法求解广义线性模型的推导。同时，这样得到的 β 的极大似然估计也可以作为惩罚似然估计的初始迭代值。

2.1 基本概念与公式

2.1.1 指数分布族

Y 的指数族密度与对数似然函数如下:

$$L(\theta) \propto f(Y;\theta,\phi) = e^{\frac{Y\theta-b(\theta)}{a(\phi)}+c(Y,\phi)}$$
$$l(\theta) = logL(\theta) = \frac{Y\theta-b(\theta)}{a(\phi)} + c(Y,\phi)$$

对样本 Y_i 同理，只需加下标再求联合。

2.1.2 典范型的常用量

求一阶和二阶导得到得分函数 $U(\theta)$ 和观测信息 $J(\theta)$:

$$U(\theta) = \frac{\partial l}{\partial \theta} = \frac{y-b'(\theta)}{a(\phi)}$$
$$J(\theta) = -\frac{\partial^2 l}{\partial \theta^2} = \frac{b''(\theta)}{a(\phi)}$$

由 Bartlett 识别又可推导:

$$E(Y) = b'(\theta);\ Var(Y) = b''(\theta)a(\phi)$$

记 $v(\mu) = b''(\theta)$ 为方差函数，$\mu = b'(\theta)$ 为期望函数。

2.1.3 连接函数

在广义线性模型中，Y 不一定服从正态分布，因此需要连接函数 $g(\cdot)$ 来使映射后的条件期望等于线性组合:

$$g(\mu_i) = X_i'\beta$$

也记 $\eta_i = X_i'\beta$，所以广义线性模型的表达式也为 $E(Y_i|X_i) = \mu_i = g^{-1}(\eta_i)$。一般使用的连接函数都为典范连接函数，要求满足 $\eta_i = \theta_i$，此时有推论: $v(\mu_i) = \frac{1}{g'(\mu_i)}$

2.2 β 的估计

用牛顿法求解 $l(\beta)$ 的优化的迭代表达式为 $\beta^{(n+1)} = \beta^{(n)} + J[\beta^{(n)}]^{-1} \cdot U[\beta^{(n)}]$。由于 $U(\beta)$ 和 $J(\beta)$ 并不直接是 β 的函数，计算过程需要用到链式法则一步步求导。

2.2.1 U(β) 的计算

分别求导得：

$$\begin{aligned}
\frac{\partial l_i}{\partial \theta_i} &= \frac{Y_i - b'(\theta)}{a(\phi)} \\
\frac{\partial \theta_i}{\partial \mu_i} &= [\frac{\partial b'(\theta_i)}{\partial \theta_i}]^{-1} = \frac{1}{b''(\theta_i)} = \frac{1}{v(\mu_i)} \\
\frac{\partial \mu_i}{\partial \eta_i} &= [\frac{\partial g(\mu_i)}{\partial \mu_i}]^{-1} = \frac{1}{g'(\mu_i)} \\
\frac{\partial \eta_i}{\partial \beta} &= X_i
\end{aligned}$$

又由链式法则得：

$$\begin{aligned}
U(\beta) &= \sum_{i=1}^{n} U_i(\beta) \\
&= \sum_{i=1}^{n} \frac{\partial l_i}{\partial \theta_i} \frac{\partial \theta_i}{\partial \mu_i} \frac{\partial \mu_i}{\partial \eta_i} \frac{\partial \eta_i}{\partial \beta} \\
&= \sum_{i=1}^{n} \frac{Y_i - \mu_i}{a(\phi)} \frac{1}{v(\mu_i) g'(\mu_i)} X_i \\
&= \frac{1}{a(\phi)} X' V^{-1} \Delta^{-1} (Y - \mu) \\
&= \frac{1}{a(\phi)} X' (Y - \mu)
\end{aligned}$$

其中 $V = diag\{v(\mu_1), ..., v(\mu_n)\}$，$\Delta = diag\{g'(\mu_1), ..., g'(\mu_n)\}$。当 $g(\cdot)$ 为典范连接函数时，有 $v(\mu_i) = \frac{1}{g'(\mu_i)}$，最后一步化简能够成立。

2.2.2 J(β) 的计算

同样地，当 $g(\cdot)$ 为典范连接函数时：

$$\frac{\partial U_i}{\partial \mu_i} = \frac{\partial \frac{1}{a(\phi)} (Y_i - \mu_i) X_i}{\partial \mu_i} = -\frac{1}{a(\phi)} X_i$$

由链式法则得：

$$\begin{aligned} J(\beta) &= -\frac{\partial U(\beta)}{\partial \beta'} \\ &= -\sum_{i=1}^{n} \frac{\partial U_i}{\partial \beta'} \\ &= -\sum_{i=1}^{n} \frac{\partial U_i}{\partial \mu_i} \frac{\partial \mu_i}{\partial \eta_i} \frac{\partial \eta_i}{\partial \beta'} \\ &= \sum_{i=1}^{n} \frac{1}{a(\phi)} X_i v(\mu_i) X_i' \\ &= \frac{1}{a(\phi)} X'VX \end{aligned}$$

3 Logistic 和 Poisson 回归的 SCAD 估计算法

有了线性模型的惩罚似然估计和广义线性模型的参数估计求解方法，可以很容易地给出 Logistic 回归和 Poisson 回归的 SCAD 估计算法。

3.1 参数设定

3.1.1 Logistic 回归的 GLM 参数

Logistic 回归假设 $Y \sim B(1, \mu)$，由其指数分布族可得：

$$a(\phi) = 1, \quad \theta = log\frac{\mu}{1-\mu}, \quad b(\theta) = log(1-\mu)$$

进一步可求得：

$$v(\mu) = b''(\theta) = \mu(1-\mu), \quad \mu_i = \frac{e^{X_i'\beta}}{1+e^{X_i'\beta}}$$

全部用矩阵形式表示 V 和 μ 比较繁琐，但是在 R 语言里可以方便地计算。

3.1.2 Poisson 回归的 GLM 参数

Poisson 回归假设 $Y \sim P(\mu)$，由其指数分布族可得：

$$a(\phi) = 1, \quad \theta = log\mu, \quad b(\theta) = \mu$$

进一步可求得：

$$v(\mu) = b''(\theta) = \mu^2, \quad \mu_i = e^{X_i'\beta}$$

3.1.3 SCAD 惩罚函数

SCAD 惩罚函数定义为：

$$p_\lambda'(|\theta|) = \lambda\{I(|\theta| \le \lambda) + \frac{(a\lambda - |\theta|)_+}{(a-1)\lambda} I(|\theta| > \lambda)\}$$

参数值一般选为 Fan 和 Li（2001）推荐的 $a = 3.7$。对应的矩阵 Σ_λ 为：

$$\Sigma_\lambda(\beta^{(n)}) = \lambda \begin{bmatrix} \frac{I(|\beta_1^{(n)}| \le \lambda) + \frac{(a\lambda - |\beta_1^{(n)}|)_+}{(a-1)\lambda} I(|\beta_1^{(n)}| > \lambda)}{|\beta_1^{(n)}|} & & \\ & \ddots & \\ & & \frac{I(|\beta_p^{(n)}| \le \lambda) + \frac{(a\lambda - |\beta_p^{(n)}|)_+}{(a-1)\lambda} I(|\beta_p^{(n)}| > \lambda)}{|\beta_p^{(n)}|} \end{bmatrix}$$

3.2 LQA 算法

GLM 的求解是极大化对数似然函数 $l(\beta)$，等价于极小化 $-l(\beta)$，此时可以按既有框架加上惩罚项，最终归结为 $Q(\beta) = \frac{1}{n} l(\beta) - \sum_{j=1}^{p} p_\lambda(|\beta_j|)$ 的优化问题：

$$\begin{aligned} a(\phi) &= 1 \ for \ both \ model \\ Q'(\beta) &= \frac{1}{n} U(\beta) - \Sigma_\lambda \beta \\ Q''(\beta) &= \frac{1}{n} J(\beta) - \Sigma_\lambda \\ \beta^{(n+1)} &= \beta^{(n)} - [\frac{1}{n} J(\beta^{(n)}) - \Sigma_\lambda]^{-1} \cdot [\frac{1}{n} U(\beta^{(n)}) - \Sigma_\lambda \beta^{(n)}] \\ &= \beta^{(n)} - (X'VX - n\Sigma_\lambda)^{-1} \cdot [X'(Y - \mu) - n\Sigma_\lambda \beta^{(n)}] \end{aligned}$$

不过由于 R 语言的精度问题，在程序中这个迭代式常常算不出来，于是做进一步简化：

$$\begin{aligned} [X'VX - n\Sigma_\lambda] \cdot \beta^{(n+1)} &= [X'VX - n\Sigma_\lambda] \cdot \beta^{(n)} - X'(Y - \mu) + n\Sigma_\lambda \beta^{(n)} \\ &= X'VX \cdot \beta^{(n)} - X'(Y - \mu) \\ &= X' \cdot [VX\beta^{(n)} - Y + \mu] \end{aligned}$$

令 $Z = VX\beta^{(n)} - Y + \mu$，则：

$$\beta^{(n+1)} = (X'VX - n\Sigma_\lambda)^{-1} \cdot X'Z$$

这个计算式在 GLM 中也被称为迭代重加权最小二乘法（Iteratively Re-Weighted Least Squares，IWLS）。

算法 3.1: Logistic LQA

$set:$

$thd_1 = 10^{-6},\ thd_2 = 10^{-3}$

$\beta^{(0)} = argmax_\beta\ l(\beta)$

$repeat:$

$\eta = X\beta^{(n)}, \quad \mu = e^\eta/(1_n + e^\eta)$

$V = diag\{\mu * (1-\mu)\}, \quad Z = V \cdot \eta - Y + \mu$

$\Sigma_\lambda = diag\{p'_\lambda(|\beta^{(n)}|)/|\beta^{(n)}|\}$

$\beta^{(n+1)} = (X'VX - n\Sigma_\lambda)^{-1} \cdot X'Z$

$if\ \|\beta^{(n+1)} - \beta^{(n)}\|_2 \le thd_2:$

$break$

$for\ j\ in\ 1, 2, ..., p:$

$if\ |\beta_j^{(n+1)}| \le thd_1:$

$\beta_j^{(n+1)} = 0$

$return\ \beta = \beta^{(n+1)}$

算法 3.2: Poisson LQA

$set:$

$thd_1 = 10^{-6},\ thd_2 = 10^{-3}$

$\beta^{(0)} = argmax_\beta\ l(\beta)$

$repeat:$

$\eta = X\beta^{(n)}, \quad \mu = e^\eta$

$V = diag\{\mu * \mu\}, \quad Z = V \cdot \eta - Y + \mu$

$\Sigma_\lambda = diag\{p'_\lambda(|\beta^{(n)}|)/|\beta^{(n)}|\}$

$\beta^{(n+1)} = (X'VX - n\Sigma_\lambda)^{-1} \cdot X'Z$

$if\ \|\beta^{(n+1)} - \beta^{(n)}\|_2 \le thd_2:$

$break$

$for\ j\ in\ 1, 2, ..., p:$

$if\ |\beta_j^{(n+1)}| \le thd_1:$

$\beta_j^{(n+1)} = 0$

$return\ \beta = \beta^{(n+1)}$

3.3 MM 算法

算法 3.3: Logistic MM

$set:$

$thd_1 = 10^{-6},\ thd_2 = 10^{-3},\ \tau = 10^{-8}$

$\beta^{(0)} = argmax_\beta\ l(\beta)$

$repeat:$

$\eta = X\beta^{(n)}, \quad \mu = e^\eta/(1_n + e^\eta)$

$V = diag\{\mu * (1-\mu)\}, \quad Z = V\cdot\eta - Y + \mu$

$\Sigma_\lambda = diag\{p'_\lambda(|\beta^{(n)}|)/(|\beta^{(n)}| + \tau)\}$

$\beta^{(n+1)} = (X'VX - n\Sigma_\lambda)^{-1}\cdot X'Z$

$for\ j\ in\ 1, 2, ..., p:$

$if\ |\beta_j^{(n+1)}| \le thd_1:$

$\beta_j^{(n+1)} = 0$

$if\ \|\beta^{(n+1)} - \beta^{(n)}\|_2 \le thd_2:$

$break$

$return\ \beta = \beta^{(n+1)}$

算法 3.4: Poisson MM

$set:$

$thd_1 = 10^{-6},\ thd_2 = 10^{-3},\ \tau = 10^{-8}$

$\beta^{(0)} = argmax_\beta\ l(\beta)$

$repeat:$

$\eta = X\beta^{(n)}, \quad \mu = e^\eta$

$V = diag\{\mu * \mu\}, \quad Z = V\cdot\eta - Y + \mu$

$\Sigma_\lambda = diag\{p'_\lambda(|\beta^{(n)}|)/(|\beta^{(n)}| + \tau)\}$

$\beta^{(n+1)} = (X'VX - n\Sigma_\lambda)^{-1}\cdot X'Z$

$for\ j\ in\ 1, 2, ..., p:$

$if\ |\beta_j^{(n+1)}| \le thd_1:$

$\beta_j^{(n+1)} = 0$

$if\ \|\beta^{(n+1)} - \beta^{(n)}\|_2 \le thd_2:$

$break$

$return\ \beta = \beta^{(n+1)}$

算法中有几点需要注意:

（1）算法中设定了两个阈值 thd_1 和 thd_2，前者作用是当迭代中的 β 的某个分量的绝对值小于它时，便将其压缩为零；后者作用是当 β 的两次迭代的二范数大小小于它（即已经收敛）时退出循环结束算法。

（2）LQA 压缩 β 分量的操作思路有两种，一是每次将 β 的分量压缩为零后，当它在下一步迭代中进入 Σ_λ 时，在分母上用一个极小值替代以免分母为零；二是在最后退出循环之后再进行分量压缩。我曾在线性模型上做的实验结果表明两种方法的结果相差无几，本次实验采用后者。

（3）$*$ 和 / 代表两个向量对应位置相乘和相除。

4 参数 λ 的调节准则

经典变量选择的广义信息准则（generalized information criterion, GIC）定义为：

$$GIC_{K_n}(S) = log(\hat{\sigma}_S^2) + \frac{K_n}{n}|S|$$

其中 S 代表一个候选模型，K_n 是一个用来控制变量选择性质的正数，不同的 K_n 对应了不同的信息准则，$|S|$ 代表模型大小。

GIC 截断参数选择准则定义为：

$$GIC_{K_n}(\lambda) = \frac{1}{n}G(Y, \hat{\beta}_\lambda) + \frac{K_n}{n}df_\lambda$$

其中 $G(Y, \hat{\beta_\lambda})$ 代表模型的拟合程度，如残差平方和，β_λ 是当截断参数选为 λ 时对应的回归系数 β 的惩罚似然估计，df_λ 代表自由度，也就是非零系数的个数。

对于 GLM，一般采用离差表征拟合程度，定义为 $D(Y; \hat{\mu}_\lambda) = 2Y; Y) - l(\hat{\mu}_\lambda; Y)$，则其参数选择准则为：

$$GIC_{K_n}(\lambda) = \frac{1}{n}D(Y; \hat{\mu}_\lambda) + \frac{K_n}{n}df_\lambda$$

4.1 BIC 准则

当 $K_n = log(n)$ 时，GIC 归结为 BIC 准则：

$$BIC(\lambda) = \frac{1}{n}D(Y; \hat{\mu}_\lambda) + \frac{log(n)}{n}df_\lambda$$

已有定理证明 BIC 可以相合地识别正确模型，对于 GLM 的惩罚似然估计结果有 Oracle 性质。极小化上式 $BIC(\lambda)$ 即可得到 λ。

5 模拟

5.1 问题重述

（1）考虑两种不同的样本量 n 和维数 p，低维高信噪比和较高维低信噪比。分别为 n=100，p=8，4 个非零系数和 n=500，p=40，4 个非零系数。非零系数在 Logistic 模型下全部取 1，Poisson 模型下全部取 0.5，因为在单次

实验中发现目前程序的稳定性和精度不够，太大的系数容易造成数值崩溃。

（2）自变量 X 的协方差矩阵一共有七种情况，分别是单位阵（不同分量不相关），不同分量固定相关系数和分量相距越远相关性越小的矩阵，后两者相关系数分别取 0.3，0.5 和 0.7。

（3）分别对 Logistic 和 Poisson 假定的模型使用以下四种估计：LQA 迭代的 SCAD 估计；MM 迭代的 SCAD 估计；glmnet 包的 Lasso 估计；SIS 包的 SCAD 估计。为保证可比性，所有方法的 λ 都取 glmnet 包中 cv.glmnet 函数用交叉验证方法选出的值。重复模拟 200 次并计算 C、IC 和 AEE 等指标。

（4）Logistic 回归的预测结果为二分类，预测效果可以直接用平均预测准确率（AA）衡量；Poisson 回归适用于事件的发生次数，和连续型变量比较相似，预测效果可以用测试均方误差（MSE）衡量。

（5）对 Poisson 模型中 n=100，p=8，4 个非零系数，算法为 MM，协差阵为第二类矩阵相关系数 0.5 的情况，以 0.1 为步长遍历 λ 从 0 到 5 的不同取值，根据 BIC 准则选出最好的 λ，并同样做 200 次模拟，和原模拟结果比较 C、IC、AEE 等指标。

5.2 结果展示

5.2.1 Logistic 回归模拟结果

首先是四种方法在 Logistic 模型上的表现情况，每个表格列出了一种 (n,p) 取值下所有 7 中协差阵的结果，其中“Oracle”代表准确值。

表 1: Logistic Regression，n = 100，p = 8

		LQA-SCAD	MM-SCAD	glmnet-Lasso	SIS-SCAD	Oracle
Σ_1	C	3.955	0.055	3.965	2.895	4
	IC	1.610	0.990	3.920	0.165	0
	AEE	3.918	4.015	1.253	1.007	0
	AA	0.503	0.502	0.682	0.598	1
Σ_2	C	3.845	0.345	3.945	3.695	4
$\rho = 0.3$	IC	0.310	1.830	3.345	0.185	4
	AEE	10.032	4.080	2.002	3.071	0
	AA	0.583	0.519	0.728	0.691	1
Σ_2	C	3.540	0.370	3.880	3.595	4
$\rho = 0.5$	IC	0.484	1.220	3.020	0.385	0
	AEE	2.157	2.246	2.830	7.283	0
	AA	0.568	0.524	0.745	0.731	1
Σ_2	C	3.200	0.325	3.725	3.345	4
$\rho = 0.7$	IC	0.895	1.165	2.665	0.655	0
	AEE	8.090	5.952	4.023	4.949	0
	AA	0.549	0.507	0.741	0.744	1
Σ_3	C	3.805	0.345	3.970	3.580	4
$\rho = 0.3$	IC	1.945	1.515	3.640	0.200	0
	AEE	3.926	3.394	1.580	2.805	0
	AA	0.502	0.514	0.727	0.685	1
Σ_3	C	3.630	1.295	3.910	3.550	4
$\rho = 0.5$	IC	1.710	1.590	3.250	0.430	0
	AEE	4.038	9.389	2.710	2.661	0
	AA	0.521	0.544	0.726	0.685	1
Σ_3	C	3.330	1.345	3.700	3.105	4
$\rho = 0.7$	IC	1.205	1.315	2.265	0.895	0
	AEE	2.334	9.653	3.514	2.812	0
	AA	0.548	0.566	0.736	0.698	1

表 2: Logistic Regression， n = 500， p = 40

		LQA-SCAD	MM-SCAD	glmnet-Lasso	SIS-SCAD	Oracle
Σ_1	C	3.985	0	4	4	4
	IC	13.295	7.860	3.601	0	0
	AEE	3.909	4.028	0.871	1.173	0
	AA	0.504	0.502	0.698	0.643	1
Σ_2	C	3.980	1.125	4	4	4
$\rho = 0.3$	IC	15.840	16.595	3.582	0	0
	AEE	3.952	6.494	1.554	1.239	0
	AA	0.512	0.501	0.755	0.719	1
Σ_2	C	3.955	1.635	4	4	4
$\rho = 0.5$	IC	25.605	16.415	3.493	0	0
	AEE	1.017	2.776	2.462	2.100	0
	AA	0.524	0.503	0.776	0.751	1
Σ_2	C	3.910	1.845	4	3.995	4
$\rho = 0.7$	IC	24.855	15.505	3.465	0.005	0
	AEE	5.111	1.751	4.684	3.203	0
	AA	0.538	0.507	0.785	0.775	1
Σ_3	C	3.930	0.715	4	4	4
$\rho = 0.3$	IC	13.985	16.010	3.670	0	0
	AEE	3.881	3.832	1.196	1.105	0
	AA	0.510	0.522	0.736	0.678	1
Σ_3	C	3.940	1.340	4	3.995	4
$\rho = 0.5$	IC	10.090	16.540	3.564	0.005	0
	AEE	3.906	2.778	1.894	1.237	0
	AA	0.506	0.519	0.759	0.695	1
Σ_3	C	3.770	1.795	4	3.825	4
$\rho = 0.7$	IC	10.720	17.835	3.348	0.075	0
	AEE	23.928	18.704	3.711	1.891	0
	AA	0.504	0.508	0.782	0.717	1

5.2.2 Poisson 回归模拟结果

同样地，在 Poisson 模型上的模拟结果如下：

表 3: Poisson Regression，n = 100，p = 8

		LQA-SCAD	MM-SCAD	glmnet-Lasso	SIS-SCAD	Oracle
Σ_1	C	2.820	3.160	4	4	4
	IC	2.100	1.650	0	0	0
	AEE	11.898	17.880	0.066	0.325	0
	MSE	6029	3890	309	392	0
Σ_2	C	2.770	3.295	4	3.985	4
$\rho = 0.3$	IC	2.335	1.816	0	0.015	0
	AEE	12.778	6.610	0.054	0.331	0
	MSE	78369	19744	4542	14048	0
Σ_2	C	2.770	3.405	4	3.905	4
$\rho = 0.5$	IC	2.375	1.840	0	0.095	0
	AEE	35.407	5.950	0.0569	0.326	0
	MSE	22973	13192	1394	4849	0
Σ_2	C	2.690	3.415	4	3.605	4
$\rho = 0.7$	IC	2.235	1.715	0	0.395	0
	AEE	52.233	38.039	0.074	0.504	0
	MSE	39192	28656	1907	5483	0
Σ_3	C	2.810	3.125	4	3.975	4
$\rho = 0.3$	IC	2.150	1.555	0	0.025	0
	AEE	183	21.502	0.055	0.328	0
	MSE	2915	2648	579	970	0
Σ_3	C	2.440	3.010	4	3.940	4
$\rho = 0.5$	IC	2.165	1.510	0	0.06	0
	AEE	74.116	22.272	0.049	0.337	0
	MSE	27661	30978	762	1327	0
Σ_3	C	2.460	3.165	4	3.665	4
$\rho = 0.7$	IC	2.050	1.500	0	0.335	0
	AEE	75.625	638	0.065	0.524	0
	MSE	9312	9710	1282	2526	0

表 4: Poisson Regression，n = 500，p = 40

		LQA-SCAD	MM-SCAD	glmnet-Lasso	SIS-SCAD	Oracle
Σ_1	C	3.090	3.475	4	4	4
	IC	18.205	9.570	0	0	0
	AEE	448	112	0.025	0.260	0
	MSE	4375	5242	1669	1770	0
Σ_2	C	3.270	3.895	4	4	4
$\rho = 0.3$	IC	18.695	9.025	0	0	0
	AEE	153	872	0.014	0.252	0
	MSE	17617	19258	2378	6945	0
Σ_2	C	3.190	3.945	4	4	4
$\rho = 0.5$	IC	18.820	6.125	0	0	0
	AEE	36.978	435	0.017	0.273	0
	MSE	10858	24247	3859	9768	0
Σ_2	C	3.155	3.960	4	3.945	4
$\rho = 0.7$	IC	19.195	14.160	0	0.055	0
	AEE	135	339	0.021	0.310	0
	MSE	201641	34872	4547	28674	0
Σ_3	C	2.930	3.480	4	4	4
$\rho = 0.3$	IC	18.190	8.756	0	0	0
	AEE	20.668	239	0.013	0.256	0
	MSE	8179	1873	2951	4117	0
Σ_3	C	2.665	3.530	4	4	4
$\rho = 0.5$	IC	17.930	8.740	0	0	0
	AEE	134	410	0.009	0.257	0
	MSE	36986	30784	3301	12318	0
Σ_3	C	2.430	3.670	4	3.995	4
$\rho = 0.7$	IC	18.280	10.721	0	0.005	0
	AEE	334	245	0.008	0.283	0
	MSE	83576	17903	4314	19404	0

5.2.3 用 BIC 选取调节参数

用 BIC 准则选取 λ 的模拟结果如下：

表 5: Poisson Regression，n = 100，p = 8

MM-SCAD				
Σ_2	C	IC	AEE	MSE
$\rho = 0.5$	4	1.025	1.447	1836

6 分析结论与体会

（1）LQA 算法在 Logistic 回归上表现比 MM 更好，MM 在 Logistic 上几乎没法使用，不知道是不是因为 τ 的数值不太适合，而 MM 在 Poisson 回归上比 LQA 表现更好。glmnet 中的 Lasso 方法和 SIS 中的 SCAD 方法估计效果基本都很好，但是在 Logistic 模型中前者倾向于把更多的系数估计为非零（C 和 IC 都大），也许是因为在这种情况下 SCAD 的压缩更狠。

（2）随着 n 和 p 都增大，信噪比降低的情况下，所有方法的效果都会变差，但没有断崖式的变化。协差阵的相关系数增大同样会是估计效果恶化，总体上变量间相关性越小越好估计（第一类好于第三类好于第二类）。

（3）在 C 和 IC 的表现上 LQA、MM 和程序包的 Lasso、SCAD 差别不大，但在 AEE 和 AA/MSE 的表现上 LQA、MM 严重失真（通常比后者大一个数量级），尤其在 n 和 p 都增大后更为明显。这应该表示我的程序目前只能做到变量选择，还不能在压缩的同时精确而稳定地估计选出的系数。

（4）使用 BIC 准则选取 λ 的 MM 算法估计优于同等条件下用 glmnet 的交叉验证选取 λ 的结果。

（5）对于（3），个人认为全矩阵运算的算法虽然简单美观，公式明了，但是难免在程序中会由于精度问题造成程序崩溃或者算法不收敛。比如在单次实验中我发现当 p 接近于 n 或者 $\tau = 1^{-8}$ 这样参数取得比较极端时，由于 R 语言的精度不够，计算中非常容易积累误差导致最终报错。本程序最终加入了大量退出条件来保证稳定运行，但这也导致它不是在所有情况下都有效的。于是总结了以下几点供日后的学习中参考：

（a）矩阵运算化简时尽量减少求逆次数和加法乘法的混合运算。

（b）学习一下怎么在 R 中使用更精细的数据结构，或者插入 C 语言。

（c）有时可以把矩阵再转化会求和形式。

7 参考文献

[1] Fan, J and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. J.Amer. Statist. Assoc. 96, 1348–1360.

[2] Hunter, D. and Li, R. (2005). Variable selection using MM algorithms. Ann. Statist. 33, 1617–1642.

[3] 李高荣，吴密霞. 2021. 多元统计分析. 北京：科学出版社

[4] Heather Turner. 2008. Introduction to Generalized Linear Models. https://statmath.wu.ac.at/courses/heather_turner/glmCourse_001.pdf

[5] Fan, Y. and Tang, C. (2013) Tuning parameter selection in high dimensional penalized likelihood. J. R. Statist. Soc. B 75, 531–552.