1 Smooth Loss and Smooth Regularizer

1.1 Objective Function

Given smooth loss function, $l\left(\beta_0 + \boldsymbol{x}_j^T\boldsymbol{\beta}; y_j\right)$, and smooth regularization, $R(\beta_0, \boldsymbol{\beta})$, we can get the objective function under the full data like

$$Q^*(\beta_0, \boldsymbol{\beta}) = \sum_{j=1}^{n} l\left(\beta_0 + \boldsymbol{x}_j^T\boldsymbol{\beta}; y_j\right) + R(\beta_0, \boldsymbol{\beta}) \tag{1}$$

The leave-i-out objection function as

$$Q(\beta_0, \boldsymbol{\beta}) = \sum_{j\neq i} l\left(\beta_0 + \boldsymbol{x}_j^T\boldsymbol{\beta}; y_j\right) + R(\beta_0, \boldsymbol{\beta}) \tag{2}$$

1.2 Approximate Leave-i-Out Prediction

Assuming $\hat{\beta}_0, \hat{\boldsymbol{\beta}}$ are the estimated parameters under the full dataset.
Based on Newton's method, we have

$$\begin{bmatrix}\tilde{\beta}_0^{/i}\\ \tilde{\boldsymbol{\beta}}^{/i}\end{bmatrix} = \begin{bmatrix}\hat{\beta}_0\\ \hat{\boldsymbol{\beta}}\end{bmatrix} - \left[\nabla^2 Q\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right)\right]^{-1} \nabla Q\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right) \tag{3}$$

where,

$$\begin{cases} \nabla^2 Q\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right) = \sum_{j\neq i} \begin{bmatrix}1\\ \boldsymbol{x}_j\end{bmatrix}\begin{bmatrix}1 & \boldsymbol{x}_j^T\end{bmatrix} \ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_j^T\hat{\boldsymbol{\beta}}; y_j\right) + \nabla^2 R\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right) \\ \nabla Q\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right) = -\begin{bmatrix}1\\ \boldsymbol{x}_i\end{bmatrix} \dot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}; y_i\right) \end{cases}$$

So, now we have

$$\begin{bmatrix}1 & \boldsymbol{x}_i^T\end{bmatrix}\begin{bmatrix}\tilde{\beta}_0^{/i}\\ \tilde{\boldsymbol{\beta}}^{/i}\end{bmatrix} = \begin{bmatrix}1 & \boldsymbol{x}_i^T\end{bmatrix}\begin{bmatrix}\hat{\beta}_0\\ \hat{\boldsymbol{\beta}}\end{bmatrix} + \begin{bmatrix}1 & \boldsymbol{x}_i^T\end{bmatrix}\left\{\sum_{j\neq i}\begin{bmatrix}1\\ \boldsymbol{x}_j\end{bmatrix}\begin{bmatrix}1 & \boldsymbol{x}_j^T\end{bmatrix}\ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_j^T\hat{\boldsymbol{\beta}}; y_j\right) + \nabla^2 R\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right)\right\}^{-1}\begin{bmatrix}1\\ \boldsymbol{x}_i\end{bmatrix}\dot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}; y_i\right)$$

Using the matrix inversion lemma, we get

$$\begin{bmatrix}1 & \boldsymbol{x}_i^T\end{bmatrix}\begin{bmatrix}\tilde{\beta}_0^{/i}\\ \tilde{\boldsymbol{\beta}}^{/i}\end{bmatrix} = \hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}} + \frac{H_{ii}}{1 - H_{ii}\ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}; y_i\right)}\dot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}; y_i\right) \tag{4}$$

where,

$$\boldsymbol{H} = [\mathbf{1}_n, \boldsymbol{X}]\left\{[\mathbf{1}_n, \boldsymbol{X}]^T\boldsymbol{D}[\mathbf{1}_n, \boldsymbol{X}] + \nabla^2 R\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right)\right\}^{-1}[\mathbf{1}_n, \boldsymbol{X}]^T$$

$$\boldsymbol{D} = \operatorname{diag}\left(\ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_1^T\hat{\boldsymbol{\beta}}; y_1\right), ..., \ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_n^T\hat{\boldsymbol{\beta}}; y_n\right)\right)$$

2 Elastic Net

2.1 Objective Function

Given the loss function,

$$l\left(\beta_0 + \boldsymbol{x}_j^T\boldsymbol{\beta}; y_j\right) = \frac{1}{2}\left(y_j - \beta_0 - \boldsymbol{x}_j^T\boldsymbol{\beta}\right)^2$$

, and the regularization $R(\beta_0, \boldsymbol{\beta}) = \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_2^2$, we can get the objective function under the full data like

$$Q^*(\beta_0, \boldsymbol{\beta}) = \sum_{j=1}^{n} l\left(\beta_0 + \boldsymbol{x}_j^T\boldsymbol{\beta}; y_j\right) + R(\beta_0, \boldsymbol{\beta})$$

The leave-i-out objection function as

$$\begin{aligned} Q(\beta_0, \boldsymbol{\beta}) =& \sum_{j \neq i} l\left(\beta_0 + \boldsymbol{x}_j^T \boldsymbol{\beta}; y_j\right) + R(\beta_0, \boldsymbol{\beta}) \\ =& \frac{1}{2} \sum_{j \neq i} \left(y_j - \beta_0 - \boldsymbol{x}_j^T \boldsymbol{\beta}\right)^2 + R(\beta_0, \boldsymbol{\beta}) \\ =& \frac{1}{2} \sum_{j=1}^{n} \left(y_j - \beta_0 - \boldsymbol{x}_j^T \boldsymbol{\beta}\right)^2 + R(\beta_0, \boldsymbol{\beta}) - \frac{1}{2}\left(y_i - \beta_0 - \boldsymbol{x}_i^T \boldsymbol{\beta}\right)^2 \end{aligned}$$

2.2 Primal Domain

Under the primal domain, we can notice that the loss function is smooth and the regularizer is not smooth, which has one zero-order singularity $K = \{0\}$ for all of the parameters except β_0.
Assuming $\hat{\beta}_0, \hat{\boldsymbol{\beta}}$ are the estimated parameters under the full dataset.
Define active set $A = \{i : \beta_i \notin K, i = 1, ..., p\}$. Then based on the formula (31), we can compute $\boldsymbol{H}$ as

$$\boldsymbol{H} = [\mathbf{1}_n, \boldsymbol{X}_{\cdot A}] \left\{ [\mathbf{1}_n, \boldsymbol{X}_{\cdot A}]^T \boldsymbol{D} [\mathbf{1}_n, \boldsymbol{X}_{\cdot A}] + \nabla^2 R\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}_A\right) \right\}^{-1} [\mathbf{1}_n, \boldsymbol{X}_{\cdot A}]^T$$

where,

$$\boldsymbol{D} = \operatorname{diag}\left(\ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_1^T \hat{\boldsymbol{\beta}}; y_1\right), ..., \ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_n^T \hat{\boldsymbol{\beta}}; y_n\right)\right) = \boldsymbol{I}$$

$$\nabla^2 R\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}_A\right) = \begin{bmatrix} 0 & & & \\ & 2\lambda_2 & & \\ & & \ddots & \\ & & & 2\lambda_2 \end{bmatrix}$$

At last, we have

$$\begin{bmatrix} 1 & \boldsymbol{x}_i^T \end{bmatrix} \begin{bmatrix} \tilde{\beta}_0^{/i} \\ \hat{\boldsymbol{\beta}}^{/i} \end{bmatrix} = \hat{\beta}_0 + \boldsymbol{x}_i^T \hat{\boldsymbol{\beta}} + \frac{H_{ii}}{1 - H_{ii} \ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T \hat{\boldsymbol{\beta}}; y_i\right)} \dot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T \hat{\boldsymbol{\beta}}; y_i\right)$$

2.3 Proximal Operator

Similar as formula (33), we can define the proximal operatro as

$$\begin{bmatrix} \hat{\beta}_0^{/i} \\ \hat{\boldsymbol{\beta}}^{/i} \end{bmatrix} = \mathbf{prox}_{R(\cdot)} \left(\begin{bmatrix} \hat{\beta}_0^{/i} \\ \hat{\boldsymbol{\beta}}^{/i} \end{bmatrix} - \sum_{j \neq i} \begin{bmatrix} 1 \\ \boldsymbol{x}_j \end{bmatrix} \dot{l}\left(\hat{\beta}_0^{/i} + \boldsymbol{x}_j^T \hat{\boldsymbol{\beta}}^{/i}; y_j\right) \right)$$

where, based on the first order condition,

$$\sum_{j \neq i} \dot{l}\left(\hat{\beta}_0^{/i} + \boldsymbol{x}_j^T \hat{\boldsymbol{\beta}}^{/i}; y_j\right) = 0$$

Define

$$\boldsymbol{u} = \begin{bmatrix} \hat{\beta}_0^{/i} \\ \hat{\boldsymbol{\beta}}^{/i} \end{bmatrix} - \sum_{j \neq i} \begin{bmatrix} 1 \\ \boldsymbol{x}_j \end{bmatrix} \dot{l}\left(\hat{\beta}_0^{/i} + \boldsymbol{x}_j^T \hat{\boldsymbol{\beta}}^{/i}; y_j\right)$$

Hence, for all of the point in the active set, $A = \{i : \beta_i \notin K, i = 1, ..., p\}$, the Jacobian of proximal operator equals to

$$\boldsymbol{J}_{E,E} = [\boldsymbol{J}(\boldsymbol{u})]_{E,E} = \begin{bmatrix} 1 & & & & \\ & 1 + 2\lambda_2 & & & \\ & & 1 + 2\lambda_2 & & \\ & & & \ddots & \\ & & & & 1 + 2\lambda_2 \end{bmatrix}^{-1}, \quad E = 1 \cup \{i + 1 : i \in A\}$$

At last, using formula (45) and (45), we have

$$\begin{bmatrix}1 & \boldsymbol{x}_i^T\end{bmatrix}\begin{bmatrix}\tilde{\beta}_0^{/i} \\ \tilde{\boldsymbol{\beta}}^{/i}\end{bmatrix} = \hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}} + \frac{H_{ii}}{1 - H_{ii}\ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}; y_i\right)}\dot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}; y_i\right)$$

where, defining $\boldsymbol{X}^* = \begin{bmatrix}\mathbf{1}_n & \boldsymbol{X}\end{bmatrix}$,

$$\boldsymbol{H} = \boldsymbol{X}^*_{\cdot,E}\left[\boldsymbol{J}_{E,E}\boldsymbol{X}^{*T}_{\cdot,E}\boldsymbol{D}\boldsymbol{X}^*_{\cdot,E} + \boldsymbol{I}_{E,E} - \boldsymbol{J}_{E,E}\right]^{-1}\boldsymbol{J}_{E,E}\boldsymbol{X}^{*T}_{\cdot,E}$$

$$\boldsymbol{D} = \operatorname{diag}\left(\ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_1^T\hat{\boldsymbol{\beta}}; y_1\right), ..., \ddot{l}\left(\hat{\beta}_0 + \boldsymbol{x}_n^T\hat{\boldsymbol{\beta}}; y_n\right)\right)$$

3 GLMNET Optimization

Considering the Elastic Net optimization problem,

$$\min_{\beta_0,\boldsymbol{\beta}} \frac{1}{2n}\sum_{j=1}^{n}\left(y_j - \beta_0 - \boldsymbol{x}_j^T\boldsymbol{\beta}\right)^2 + \frac{1}{2}\lambda(1-\alpha)\left\|\boldsymbol{\beta}\right\|_2^2 + \lambda\alpha\left\|\boldsymbol{\beta}\right\|_1$$

However, in the documentation of 'glmnet' package, it notes that for "gaussian", glmnet standardizes y to have unit variance (using 1/n rather than 1/(n-1) formula) before computing its lambda sequence (and then unstandardizes the resulting coefficients); if you wish to reproduce/compare results with other software, best to supply a standardized y. The coefficients for any predictor variables with zero variance are set to zero for all values of lambda.

It means that the function is actually optimizing the following problem.

$$\min_{\beta_0,\boldsymbol{\beta}} \frac{1}{2n}\sum_{j=1}^{n}\left(y_j^* - \beta_0^* - \boldsymbol{x}_j^T\boldsymbol{\beta}^*\right)^2 + \frac{1}{2}\lambda(1-\alpha)\left\|\boldsymbol{\beta}^*\right\|_2^2 + \lambda\alpha\left\|\boldsymbol{\beta}^*\right\|_1$$

where,

$$\begin{cases} y_j^* = \frac{y_j}{\operatorname{sd}(\boldsymbol{y})} \\ \beta_0^* = \frac{\beta_0}{\operatorname{sd}(\boldsymbol{y})} \\ \boldsymbol{\beta}^* = \frac{\boldsymbol{\beta}}{\operatorname{sd}(\boldsymbol{y})} \end{cases}$$

So, the 'glmnet' function is actually optimizing

$$\begin{aligned}&\min_{\beta_0,\boldsymbol{\beta}}\frac{1}{2n}\sum_{j=1}^{n}\left(y_j^* - \beta_0^* - \boldsymbol{x}_j^T\boldsymbol{\beta}^*\right)^2 + \frac{1}{2}\lambda(1-\alpha)\left\|\boldsymbol{\beta}^*\right\|_2^2 + \lambda\alpha\left\|\boldsymbol{\beta}^*\right\|_1 \\ =&\min_{\beta_0,\boldsymbol{\beta}}\frac{1}{2n}\sum_{j=1}^{n}\left(\frac{y_j}{\operatorname{sd}(\boldsymbol{y})} - \frac{\beta_0}{\operatorname{sd}(\boldsymbol{y})} - \frac{\boldsymbol{x}_j^T\boldsymbol{\beta}}{\operatorname{sd}(\boldsymbol{y})}\right)^2 + \frac{1}{2}\frac{\lambda}{\operatorname{sd}(\boldsymbol{y})^2}(1-\alpha)\left\|\boldsymbol{\beta}\right\|_2^2 + \frac{\lambda}{\operatorname{sd}(\boldsymbol{y})}\alpha\left\|\boldsymbol{\beta}\right\|_1\end{aligned}$$

So, in order to coincide with the target optimization problem, we can rescale $\boldsymbol{y}$ and $\boldsymbol{X}$ by $\operatorname{sd}(\boldsymbol{y})$, λ by $\operatorname{sd}(\boldsymbol{y})^2$, then we can get

$$\begin{aligned}&\min_{\beta_0,\boldsymbol{\beta}}\frac{1}{2n}\sum_{j=1}^{n}\left(y_j - \beta_0 - \boldsymbol{x}_j^T\boldsymbol{\beta}^*\right)^2 + \frac{1}{2}\lambda(1-\alpha)\left\|\boldsymbol{\beta}\right\|_2^2 + \lambda\alpha\left\|\boldsymbol{\beta}\right\|_1 \\ =&\min_{\beta_0,\boldsymbol{\beta}}\frac{1}{2n}\sum_{j=1}^{n}\left(\frac{y_j}{\operatorname{sd}(\boldsymbol{y})} - \beta_0^* - \frac{\boldsymbol{x}_j^T}{\operatorname{sd}(\boldsymbol{y})}\boldsymbol{\beta}\right)^2 + \frac{1}{2}\frac{\lambda}{\operatorname{sd}(\boldsymbol{y})^2}(1-\alpha)\left\|\boldsymbol{\beta}\right\|_2^2 + \frac{\lambda}{\operatorname{sd}(\boldsymbol{y})^2}\alpha\left\|\boldsymbol{\beta}\right\|_1 \\ =&\min_{\beta_0,\boldsymbol{\beta}}\frac{1}{2n}\sum_{j=1}^{n}\left(y_j^* - \beta_0^* - \boldsymbol{x}_j^{*T}\boldsymbol{\beta}\right)^2 + \frac{1}{2}\lambda^*(1-\alpha)\left\|\boldsymbol{\beta}\right\|_2^2 + \lambda^*\alpha\left\|\boldsymbol{\beta}\right\|_1\end{aligned}$$

where,

$$\begin{cases} \beta_0^* = \frac{\beta_0}{\operatorname{sd}(\boldsymbol{y})} \\ \lambda^* = \frac{\lambda}{\operatorname{sd}(\boldsymbol{y})^2} \end{cases}$$

4 Multinomial

4.1 Loss function

Assume we have K classes, n observations $\boldsymbol{X}$, and $\boldsymbol{R}^p$ parameters $\boldsymbol{\beta}_k$ for each class.

Define leave-i-out variables as

$$\boldsymbol{y}^{/i}_{(n-1)K\times 1} = \{y_{jk}\}_{j\neq i}^{k=1,\dots,K} = \begin{bmatrix} \begin{bmatrix} y_{11} \\ y_{12} \\ \vdots \\ y_{1K} \end{bmatrix} \\ \begin{bmatrix} y_{21} \\ y_{22} \\ \vdots \\ y_{2K} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} y_{n1} \\ y_{n2} \\ \vdots \\ y_{nK} \end{bmatrix} \end{bmatrix}, \quad \boldsymbol{\mathcal{X}}^{/i}_{(n-1)K\times pK} = \begin{bmatrix} \begin{bmatrix} \boldsymbol{x}_1^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_1^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_1^T \end{bmatrix} \\ \begin{bmatrix} \boldsymbol{x}_2^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_2^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_2^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \boldsymbol{x}_n^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_n^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_n^T \end{bmatrix} \end{bmatrix}, \quad \boldsymbol{\mathcal{B}} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \boldsymbol{\beta}_2 \\ \vdots \\ \boldsymbol{\beta}_K \end{bmatrix}$$

The loss function would be

$$\begin{aligned} l(\boldsymbol{\mathcal{B}}) &= -\left\{ \sum_{j\neq i} \left[\sum_{k=1}^{K} y_{jk} \boldsymbol{x}_j^T \boldsymbol{\beta}_k - \log\left(\sum_{k=1}^{K} e^{\boldsymbol{x}_j^T \boldsymbol{\beta}_k} \right) \right] \right\} \\ &= \sum_{j\neq i} \log\left(\sum_{k=1}^{K} e^{\boldsymbol{x}_j^T \boldsymbol{\beta}_k} \right) - \sum_{j\neq i}^{n} \sum_{k=1}^{K} y_{jk} \boldsymbol{x}_j^T \boldsymbol{\beta}_k \end{aligned}$$

By taking the first order derivative, we get

$$\begin{aligned} \frac{\partial l(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}} &= \begin{bmatrix} \frac{\partial l(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\beta}_1} \\ \vdots \\ \frac{\partial l(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\beta}_K} \end{bmatrix} = \begin{bmatrix} \sum_{j\neq i} \frac{\exp\left(\boldsymbol{x}_j^T \boldsymbol{\beta}_1\right)}{\sum_{k=1}^{K} \exp\left(\boldsymbol{x}_j^T \boldsymbol{\beta}_k\right)} \boldsymbol{x}_j - \sum_{j\neq i} y_{j1} \boldsymbol{x}_j \\ \vdots \\ \sum_{j\neq i} \frac{\exp\left(\boldsymbol{x}_j^T \boldsymbol{\beta}_K\right)}{\sum_{k=1}^{K} \exp\left(\boldsymbol{x}_j^T \boldsymbol{\beta}_k\right)} \boldsymbol{x}_j - \sum_{j\neq i} y_{jK} \boldsymbol{x}_j \end{bmatrix} = \boldsymbol{\mathcal{X}}^{/iT} \begin{bmatrix} \begin{bmatrix} \frac{\exp\left(\boldsymbol{x}_1^T \boldsymbol{\beta}_1\right)}{\sum_{k=1}^{K} \exp\left(\boldsymbol{x}_1^T \boldsymbol{\beta}_k\right)} \\ \vdots \\ \frac{\exp\left(\boldsymbol{x}_1^T \boldsymbol{\beta}_K\right)}{\sum_{k=1}^{K} \exp\left(\boldsymbol{x}_1^T \boldsymbol{\beta}_k\right)} \end{bmatrix} \\ \begin{bmatrix} \frac{\exp\left(\boldsymbol{x}_2^T \boldsymbol{\beta}_1\right)}{\sum_{k=1}^{K} \exp\left(\boldsymbol{x}_2^T \boldsymbol{\beta}_k\right)} \\ \vdots \\ \frac{\exp\left(\boldsymbol{x}_2^T \boldsymbol{\beta}_K\right)}{\sum_{k=1}^{K} \exp\left(\boldsymbol{x}_2^T \boldsymbol{\beta}_k\right)} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \frac{\exp\left(\boldsymbol{x}_n^T \boldsymbol{\beta}_1\right)}{\sum_{k=1}^{K} \exp(\boldsymbol{x}_n^T \boldsymbol{\beta}_k)} \\ \vdots \\ \frac{\exp\left(\boldsymbol{x}_n^T \boldsymbol{\beta}_K\right)}{\sum_{k=1}^{K} \exp(\boldsymbol{x}_n^T \boldsymbol{\beta}_k)} \end{bmatrix} \end{bmatrix}_{(n-1)K\times 1} - \boldsymbol{\mathcal{X}}^{/iT} \boldsymbol{y}^{/i} \\ &= \boldsymbol{\mathcal{X}}^{/iT} \left[\boldsymbol{\mathcal{A}}^{/i}(\boldsymbol{\beta}) - \boldsymbol{y}^{/i} \right] = \boldsymbol{\mathcal{X}}^{/iT} \left(\begin{bmatrix} \boldsymbol{A}_1(\boldsymbol{\beta}) \\ \boldsymbol{A}_2(\boldsymbol{\beta}) \\ \vdots \\ \boldsymbol{A}_n(\boldsymbol{\beta}) \end{bmatrix} - \boldsymbol{y}^{/i} \right) \end{aligned}$$

Similarly, we can get

$$\frac{\partial^2 l(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}} \partial \boldsymbol{\mathcal{B}}^T} = \boldsymbol{\mathcal{X}}^{/iT} \frac{\partial \boldsymbol{\mathcal{A}}^{/i}(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} = \boldsymbol{\mathcal{X}}^{/iT} \begin{bmatrix} \frac{\partial \boldsymbol{A}_1(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} \\ \frac{\partial \boldsymbol{A}_2(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} \\ \vdots \\ \frac{\partial \boldsymbol{A}_n(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} \end{bmatrix}$$

where,

$$\begin{aligned}
\frac{\partial \boldsymbol{A}_j(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} =& \begin{bmatrix} \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_1)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} & 0 & \cdots & 0 \\ 0 & \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_2)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_K)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} \end{bmatrix}_{K \times K} \begin{bmatrix} \boldsymbol{x}_j^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_j^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_j^T \end{bmatrix}_{K \times pK} \\
& - \begin{bmatrix} \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_1)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} \\ \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_2)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} \\ \vdots \\ \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_K)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} \end{bmatrix}_{K \times 1} \begin{bmatrix} \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_1)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} & \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_2)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} & \cdots & \frac{\exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_K)}{\sum_{k=1}^K \exp(\boldsymbol{x}_j^T \boldsymbol{\beta}_k)} \end{bmatrix}_{1 \times K} \begin{bmatrix} \boldsymbol{x}_j^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_j^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_j^T \end{bmatrix}_{K \times pK} \\
=& \left[\operatorname{diag}\left(\boldsymbol{A}_j(\boldsymbol{\mathcal{B}}) \right) - \boldsymbol{A}_j(\boldsymbol{\mathcal{B}}) \boldsymbol{A}_j(\boldsymbol{\mathcal{B}})^T \right]_{K \times K} \begin{bmatrix} \boldsymbol{x}_j^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_j^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_j^T \end{bmatrix}_{K \times pK}
\end{aligned}$$

So, we have

$$\begin{aligned}
\frac{\partial^2 l(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}} \partial \boldsymbol{\mathcal{B}}^T} =& \boldsymbol{\mathcal{X}}^{/iT} \frac{\partial \boldsymbol{\mathcal{A}}^{/i}(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} = \boldsymbol{\mathcal{X}}^{/iT} \begin{bmatrix} \frac{\partial \boldsymbol{A}_1(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} \\ \frac{\partial \boldsymbol{A}_2(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} \\ \vdots \\ \frac{\partial \boldsymbol{A}_n(\boldsymbol{\mathcal{B}})}{\partial \boldsymbol{\mathcal{B}}^T} \end{bmatrix} \\
=& \boldsymbol{\mathcal{X}}^{/iT} \begin{bmatrix} \left[\operatorname{diag}\left(\boldsymbol{A}_1(\boldsymbol{\mathcal{B}}) \right) - \boldsymbol{A}_1(\boldsymbol{\mathcal{B}}) \boldsymbol{A}_1(\boldsymbol{\mathcal{B}})^T \right]_{K \times K} \begin{bmatrix} \boldsymbol{x}_1^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_1^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_1^T \end{bmatrix}_{K \times pK} \\ \left[\operatorname{diag}\left(\boldsymbol{A}_2(\boldsymbol{\mathcal{B}}) \right) - \boldsymbol{A}_2(\boldsymbol{\mathcal{B}}) \boldsymbol{A}_2(\boldsymbol{\mathcal{B}})^T \right]_{K \times K} \begin{bmatrix} \boldsymbol{x}_2^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_2^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_2^T \end{bmatrix}_{K \times pK} \\ \vdots \\ \left[\operatorname{diag}\left(\boldsymbol{A}_n(\boldsymbol{\mathcal{B}}) \right) - \boldsymbol{A}_n(\boldsymbol{\mathcal{B}}) \boldsymbol{A}_n(\boldsymbol{\mathcal{B}})^T \right]_{K \times K} \begin{bmatrix} \boldsymbol{x}_n^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_n^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_n^T \end{bmatrix}_{K \times pK} \end{bmatrix}_{(n-1)K \times pK} \\
=& \boldsymbol{\mathcal{X}}^{/iT} \boldsymbol{\mathcal{D}}^{/i}_{(n-1)K \times (n-1)K}(\boldsymbol{\mathcal{B}}) \boldsymbol{\mathcal{X}}^{/i}
\end{aligned}$$

where,

$$\boldsymbol{\mathcal{D}}^{/i}(\boldsymbol{\mathcal{B}}) = \begin{bmatrix} \left[\operatorname{diag}\left(\boldsymbol{A}_1(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_1(\boldsymbol{\mathcal{B}})\boldsymbol{A}_1(\boldsymbol{\mathcal{B}})^T\right] & 0 & \cdots & 0 \\ 0 & \left[\operatorname{diag}\left(\boldsymbol{A}_2(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_2(\boldsymbol{\mathcal{B}})\boldsymbol{A}_2(\boldsymbol{\mathcal{B}})^T\right] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \left[\operatorname{diag}\left(\boldsymbol{A}_n(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_n(\boldsymbol{\mathcal{B}})\boldsymbol{A}_n(\boldsymbol{\mathcal{B}})^T\right] \end{bmatrix}$$

4.2 Newton's Method

With Newton's method, we have the one step update as

$$\begin{aligned} \tilde{\boldsymbol{\mathcal{B}}}^{/i} =& \hat{\boldsymbol{\mathcal{B}}} - \left[\boldsymbol{\mathcal{X}}^{/iT}\boldsymbol{\mathcal{D}}^{/i}(\boldsymbol{\mathcal{B}})\boldsymbol{\mathcal{X}}^{/i} + \nabla^2 R(\boldsymbol{\mathcal{B}})\right]^{-1}\left[\boldsymbol{\mathcal{X}}^{/iT}\left(\boldsymbol{\mathcal{A}}^{/iT}(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}^{/i}\right) + \nabla R(\boldsymbol{\mathcal{B}})\right] \\ =& \hat{\boldsymbol{\mathcal{B}}} + \left[\boldsymbol{\mathcal{X}}^T\boldsymbol{\mathcal{D}}(\boldsymbol{\mathcal{B}})\boldsymbol{\mathcal{X}} + \nabla^2 R(\boldsymbol{\mathcal{B}}) - \boldsymbol{X}_i^T\left[\operatorname{diag}\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})^T\right]\boldsymbol{X}_i\right]^{-1}\boldsymbol{X}_i^T\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}_i\right) \end{aligned}$$

where,

$$\boldsymbol{X}_i = \begin{bmatrix} \boldsymbol{x}_i^T & 0 & \cdots & 0 \\ 0 & \boldsymbol{x}_i^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{x}_i^T \end{bmatrix}_{K\times pK}, \quad \boldsymbol{y}_i = \begin{bmatrix} y_{i1} \\ y_{i2} \\ \vdots \\ y_{iK} \end{bmatrix}$$

Defining $\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}}) = \boldsymbol{\mathcal{X}}^T\boldsymbol{\mathcal{D}}(\boldsymbol{\mathcal{B}})\boldsymbol{\mathcal{X}} + \nabla^2 R(\boldsymbol{\mathcal{B}})$, with matrix inversion lemma, we can get

$$\begin{aligned} \tilde{\boldsymbol{\mathcal{B}}}^{/i} =& \hat{\boldsymbol{\mathcal{B}}} + \left[\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}}) - \boldsymbol{X}_i^T\left[\operatorname{diag}\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})^T\right]\boldsymbol{X}_i\right]^{-1}\boldsymbol{X}_i^T\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}_i\right) \\ =& \boldsymbol{\mathcal{B}} + \boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}_i\right) \\ & - \boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\left\{-\left[\operatorname{diag}\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})^T\right]^{-1} + \boldsymbol{X}_i\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\right\}^{-1}\boldsymbol{X}_i\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}_i\right) \end{aligned}$$

4.3 Approximate Leave-i-Out Prediction

Given the approximate leave-i-out estimation, we can do the approximate leave-i-out prediciton as

$$\begin{aligned} \boldsymbol{y}_i^{/i} =& \begin{bmatrix} y_{i1}^{/i} \\ \vdots \\ y_{iK}^{/i} \end{bmatrix} = \boldsymbol{X}_i\tilde{\boldsymbol{\mathcal{B}}}^{/i} \\ =& \boldsymbol{X}_i\boldsymbol{\mathcal{B}} + \boldsymbol{X}_i\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}_i\right) \\ & - \boldsymbol{X}_i\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\left\{-\left[\operatorname{diag}\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\right) - \boldsymbol{A}_i(\boldsymbol{\mathcal{B}})\boldsymbol{A}_i(\boldsymbol{\mathcal{B}})^T\right]^{-1} + \boldsymbol{X}_i\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\right\}^{-1}\boldsymbol{X}_i\boldsymbol{\mathcal{K}}(\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{X}_i^T\left(\boldsymbol{A}_i(\boldsymbol{\mathcal{B}}) - \boldsymbol{y}_i\right) \end{aligned}$$

5 Computational Complexity Analysis

5.1 Regular ALO

Here, we take the loss function

$$l\left(\boldsymbol{x}_j^T\boldsymbol{\beta}; y_j\right) = \left(y_j - \boldsymbol{x}_j^T\boldsymbol{\beta}\right)^2$$

and regularizer

$$R(\boldsymbol{\beta}) = \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_2^2$$

Under the Regular ALO case, we need to do the following linear algebra computation.

Assuming the active set is E, then we have

$$\left(\boldsymbol{X}^T\boldsymbol{X}\right)_{E,E} + \nabla^2 R\left(\boldsymbol{\beta}_E\right), \quad O(|E|^2)$$

$$\left[\left(\boldsymbol{X}^T\boldsymbol{X}\right)_{E,E}+\nabla^2 R\left(\boldsymbol{\beta}_E\right)\right]^{-1},\quad O(|E|^3)$$

$$\boldsymbol{H}=\boldsymbol{X}_{\cdot E}\left[\left(\boldsymbol{X}^T\boldsymbol{X}\right)_{E,E}+\nabla^2 R\left(\boldsymbol{\beta}_E\right)\right]^{-1}\boldsymbol{X}_{\cdot E}^T,\quad O(n|E|^2)+O(n|E|)\to O\left(n|E|^2\right)$$

$$\frac{\operatorname{diag}\left(\boldsymbol{H}\right)}{1-\operatorname{diag}\left(\boldsymbol{H}\right)}.\times\left(\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{y}\right),\quad O(n)+O(np)+O(n)+O(n)\to O(np)$$

So, the final computational complexity for regular ALO method is

$$O(|E|^3)+O\left(n|E|^2\right)\to O\left(|E|^2\max\{n,|E|\}\right)$$

5.2 Block Inversion Lemma in ALO

Here, we take the loss function

$$l\left(\boldsymbol{x}_j^T\boldsymbol{\beta};y_j\right)=\left(y_j-\boldsymbol{x}_j^T\boldsymbol{\beta}\right)^2$$

and regularizer

$$R(\boldsymbol{\beta})=\lambda_1\left\|\boldsymbol{\beta}\right\|_1+\lambda_2\left\|\boldsymbol{\beta}\right\|_2^2$$

Under the Block Inversion ALO case, we need to do the following linear algebra computation.

Assume the last active set is E_t.
At this step, we need to do the following

$$E_t=E_{t1}\cup E_{t2}\xrightarrow{\text{drop}}E_{t1}\xrightarrow{\text{add}}E_{t+1}=E_{t1}\cup E_{t3}$$

At the drop step,

$$\begin{aligned}\boldsymbol{A}_t^{-1}&=\left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_tE_t}^{-1}=\begin{bmatrix}\left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t1}E_{t1}} & \left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t1}E_{t2}}\\ \left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t2}E_{t1}} & \left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t2}E_{t2}}\end{bmatrix}^{-1}\\ &=\begin{bmatrix}\boldsymbol{A}_{11} & \boldsymbol{A}_{12}\\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22}\end{bmatrix}\\ \Rightarrow\left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t1}E_{t1}}^{-1}&=\boldsymbol{A}_{11}-\boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\end{aligned}$$

$$\Rightarrow O(|E_{t2}|^3)+O(|E_{t1}||E_{t2}|^2)+O(|E_{t1}|^2|E_{t2}|)+O(|E_{t1}|^2)\to O(|E_{t2}|\max\{|E_{t1}|,|E_{t2}|\}^2)$$

At the addition step,

$$\begin{aligned}\boldsymbol{A}_{t+1}^{-1}&=\left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t+1}E_{t+1}}^{-1}=\begin{bmatrix}\left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t1}E_{t1}} & \left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t1}E_{t3}}\\ \left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t3}E_{t1}} & \left(\boldsymbol{X}^T\boldsymbol{X}+\nabla^2R(\boldsymbol{\beta})\right)_{E_{t3}E_{t3}}\end{bmatrix}^{-1}\\ &=\begin{bmatrix}\boldsymbol{A}_{11} & \boldsymbol{A}_{13}\\ \boldsymbol{A}_{31} & \boldsymbol{A}_{33}\end{bmatrix}^{-1}\\ &=\begin{bmatrix}\boldsymbol{A}_{11}^{-1}+\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{13}\boldsymbol{E}\boldsymbol{A}_{31}\boldsymbol{A}_{11}^{-1} & -\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{13}\boldsymbol{E}\\ -\boldsymbol{E}\boldsymbol{A}_{31}\boldsymbol{A}_{11}^{-1} & \boldsymbol{E}\end{bmatrix},\quad where\ \boldsymbol{E}=\left(\boldsymbol{A}_{33}-\boldsymbol{A}_{31}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{13}\right)^{-1}\end{aligned}$$

$$\Rightarrow\begin{cases}O(|E_{t1}|^2|E_{t3}|)+O(|E_{t1}||E_{t3}|^2)+O(|E_{t3}|^3)\to O(|E_{t3}|\max\{|E_{t1}|,|E_{t3}|\}^2)\\ O(|E_{t1}||E_{t3}|^2)+O(|E_{t1}|^2|E_{t3}|)\to O(|E_{t1}||E_{t3}|\max\{|E_{t1}|,|E_{t3}|\})\\ O(|E_{t1}||E_{t3}|^2)\end{cases}$$

$$\Rightarrow O(|E_{t3}|\max\{|E_{t1}|,|E_{t3}|\}^2)$$

At the Schulz Iteration step, for start point,

$$\boldsymbol{V}_0=\left[\boldsymbol{I}-(\lambda_{t+1,2}-\lambda_{t,2})\boldsymbol{A}_{t+1}^{-1}\right]\boldsymbol{A}_{t+1}^{-1},\quad O(|E_{t+1}|^3)$$

for each iteration,

$$\boldsymbol{V}_{k+1} = \boldsymbol{V}_k(2\boldsymbol{I} - \boldsymbol{F}_{t+1}\boldsymbol{V}_k), \quad where\ \boldsymbol{F}_{t+1} = \left(\boldsymbol{X}^T\boldsymbol{X} + \nabla^2 R(\boldsymbol{\beta}_{t+1})\right)_{E_{t+1}E_{t+1}}, \quad O(|E_{t+1}|^3)$$

At the final step,

$$\boldsymbol{H} = \boldsymbol{X}_{\cdot E_{t+1}}\boldsymbol{V}_{\inf}\boldsymbol{X}^T_{\cdot E_{t+1}}, \quad O(n|E_{t+1}|^2) + O(n|E_{t+1}|) \to O\left(n|E_{t+1}|^2\right)$$

$$\frac{\operatorname{diag}(\boldsymbol{H})}{1-\operatorname{diag}(\boldsymbol{H})}. \times (\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}), \quad O(n) + O(np) + O(n) + O(n) \to O(np)$$

So, the final computational complexity for Block Inversion ALO method is

$$O(|E_{t+1}|^3) + O\left(n|E_{t+1}|^2\right) \to O\left(|E_{t+1}|^2 \max\{n, |E_{t+1}|\}\right)$$

6 Cholesky Decompostion in Elastic Net

6.1 Definition

Define the original design matrix as $\boldsymbol{X}_{n\times p}$, active set E_t corresponding to the parameter λ_t. The second order derivative of regularization function is $\nabla^2 R(\boldsymbol{\beta})$

6.2 Cholesky Decompostion

Here, assuming we already know the Cholesky decomposition of

$$\boldsymbol{X}^T_{\cdot E_{t-1}}\boldsymbol{X}_{\cdot E_{t-1}} = \boldsymbol{L}_{t-1}\boldsymbol{L}^T_{t-1}$$

and at this step, we want to find out the Cholesky decomposition of

$$\boldsymbol{X}^T_{\cdot E_t}\boldsymbol{X}_{\cdot E_t} + \nabla^2 R(\boldsymbol{\beta}_t)$$

Under the Elastic Net case, we have

$$\begin{aligned}
\nabla^2 R(\boldsymbol{\beta}_t) &= \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 2\lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & 2\lambda_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 2\lambda_2 \end{bmatrix} \\
&= \begin{bmatrix} 0 \\ \sqrt{2\lambda_2} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} 0 & \sqrt{2\lambda_2} & 0 & \cdots & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \sqrt{2\lambda_2} \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & \sqrt{2\lambda_2} & \cdots & 0 \end{bmatrix} + \cdots \\
&= \boldsymbol{r}_2\boldsymbol{r}_2^T + \boldsymbol{r}_3\boldsymbol{r}_3^T + \cdots + \boldsymbol{r}_{|E_t|}\boldsymbol{r}_{|E_t|}^T
\end{aligned}$$

So, at each step, we can first update the Cholesky decomposition of

$$\boldsymbol{X}^T_{\cdot E_{t-1}}\boldsymbol{X}_{\cdot E_{t-1}} = \boldsymbol{L}_{t-1}\boldsymbol{L}^T_{t-1} \overset{Cholesky\ Update}{\Longrightarrow} \boldsymbol{X}^T_{\cdot E_t}\boldsymbol{X}_{\cdot E_t} = \boldsymbol{L}_t\boldsymbol{L}_t^T$$

Then, based on the rank-one update, we can find out

$$\boldsymbol{X}^T_{\cdot E_t}\boldsymbol{X}_{\cdot E_t} + \nabla^2 R(\boldsymbol{\beta}_t) = \boldsymbol{X}^T_{\cdot E_t}\boldsymbol{X}_{\cdot E_t} + \boldsymbol{r}_2\boldsymbol{r}_2^T + \boldsymbol{r}_3\boldsymbol{r}_3^T + \cdots + \boldsymbol{r}_{|E_t|}\boldsymbol{r}_{|E_t|}^T$$

Here we can use Cholesky rank-one update to find the Cholesky decomposition of $\boldsymbol{X}^T_{\cdot E_t}\boldsymbol{X}_{\cdot E_t} + \nabla^2 R(\boldsymbol{\beta}_t)$.