First recall the dual formulation of alo with elastic net penalty, for which the optimization problem is: $\arg\min_{\beta} \frac{1}{2}||y - X\beta||_2^2 + \lambda_1||\beta||_1 + \lambda_2||\beta||_2^2$, also let E be the active sets.

$$y^{/i} = y_i - \frac{u_i}{J_{ii}}$$

where:

$$u = y - X\hat{\beta}$$
$$J = (I + \frac{1}{2\lambda_2} X_E X_E^\tau)^{-1}$$

1 Nesterov Accelerated Gradient

Initialize learning rate γ, momentum ϵ, Initial Guess Inverse B_0
Initialize $Y_0 = X_0$
While Stopping criterion not satisfied

1. $Y_{t+1} = B_t - \gamma((I + A)B_t - I)$
2. $B_{t+1} = Y_{t+1} + \epsilon(Y_{t+1} - Y_t)$

Algorithm 1: Nesterov Accelerated Gradient Descent

1.1 Some intuitions for the algorithm

First recall the simple gradient step for computing the inverse of a positive-definite symmetric matrix A, whose spectrum decomposition is $A = U\Lambda U^\tau$:

$$X_{t+1} = X_t - \epsilon(AX_t - I)$$

Therefore we could compute the frobenius norm with regard to $AX_t - I$:

$$\begin{aligned} ||AX_{t+1} - I||_F^2 &= ||(I - \epsilon\Lambda)(AX_t - I)||_F^2 \\ &= ||U(I - \epsilon\Lambda)U^\tau(AX_t - I)||_F^2 \\ &\leq \max_i (1 - \epsilon\lambda_i)^2 ||AX_t - I||_F^2 \end{aligned}$$

If ϵ is set such that for each eigenvalue λ_i, $(1-\epsilon\lambda_i)^2 < 1-\delta$ for some $0 < \delta < 1$, then we could reduce the error $||AX_t - I||_F^2$ exponentially according to iteration times t.

While in practice of cross-validation, our λ_2 will change within each regression, therefore we can set the initial guess of the inverse directly from the inverse obtained for the previous λ_2. Given the intuition of Nesterov Accelerated Gradient, here I proposed two different NAG algorithms:

1.2 Nesterov Accelerated Gradient for one-sample a time

Denote p be the number of the columns of X_E
Initialize learning rate γ, momentum ϵ, Initial Guess Inverse B_0
Initialize $Y_0 = B_0$
While Stopping criterion not satisfied

1. sample X_j randomly from the columns of X_E
2. $Y_{t+1} = B_t - \gamma((I + \frac{p}{2\lambda_2} X_j X_j^\tau)B_t - I)$
3. $B_{t+1} = Y_{t+1} + \epsilon(Y_{t+1} - Y_t)$

Algorithm 2: NAG for one-sample per time

1.3 Nesterov Accelerated Gradient for a mini-batch a time

Initialize learning rate γ, momentum ϵ, batchsize m, Initial Guess Inverse B_0
Initialize $Y_0 = B_0$
While Stopping criterion not satisfied

1. sample m columns $\tilde{X}_E$ randomly from the columns of X_E
2. $Y_{t+1} = B_t - \gamma((I + \frac{p}{2m\lambda_2} \tilde{X}_E \tilde{X}_E^\tau)B_t - I)$
3. $B_{t+1} = Y_{t+1} + \epsilon(Y_{t+1} - Y_t)$

Algorithm 3: NAG for mini-batch per time

1.4 Implementation results for NAG inverse

Here I show the plot of the risk curve against different λ with $\alpha = 0.5$ as the elastic net penalty and record the computation time for the NAG and the direct inverse:

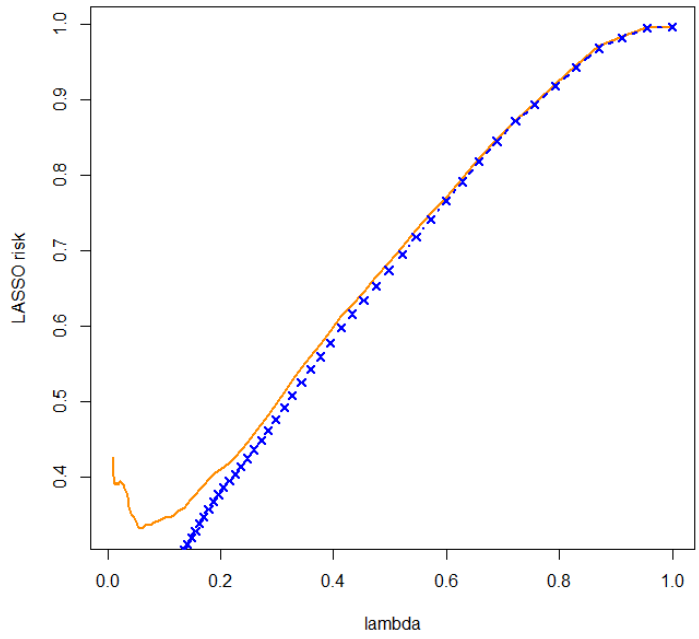

Figure 1: Risk against λ

This method works well when λ is big and the active sets is relatively small. Meanwhile controlling the γ and ϵ, it also computes faster. However, when λ approached 0, it gradually loses its power, it deviates from the leave-one-it risk curve at around the λ which gives the smallest risk.

time of NAG	time of direct inverse (no parallel)	n	p
1.07	1.30	300	600
12.72	42.73	1000	600
13.12	43.46	1000	1500

2 LiSSA for dual approach

First we rewrite the formula for alo:

$$\begin{aligned} y^{/i} &= y_i - \frac{u_i}{J_{ii}} \\ &= y_i - \frac{1}{\frac{J_{ii}}{u_i}} \end{aligned}$$

where $\frac{J_{ii}}{u_i}$ could be written as the diagonal elements of $Jdiag(1/u)$, also denote x_j as the j-th column of X_E and p as the dimensions of the column of X_E:

$$\begin{aligned} Jdiag(1/u) &= (I + \frac{1}{2\lambda_2} X_E X_E^\tau)^{-1} diag(1/u) \\ &= (I + \sum_{j=1}^{p} \frac{1}{2\lambda_2} x_j x_j^\tau)^{-1} diag(1/u) \\ &= (\sum_{j=1}^{p} (\frac{1}{2\lambda_2} x_j x_j^\tau + \frac{1}{p} I))^{-1} diag(1/u) \\ &= (\frac{1}{p} \sum_{j=1}^{p} (\frac{1}{2\lambda_2} x_j x_j^\tau + \frac{1}{p} I))^{-1} diag(\frac{1}{pu}) \end{aligned}$$

Given the representation of $Jdiag(1/u)$, we could derive the **LiSSA** for it:

Initialize S_1 and S_2

1. for i = 1 to S_1
2. initialize $A_{[i,0]} = diag(\frac{1}{pu})$
3. for j = 1 to S_2
4. sample $x_{[i,j]}$ uniformly from all columns of X_E
5. $A_{[i,j]} = diag(\frac{1}{pu}) + (I - \frac{1}{2\lambda_2} x_{[i,j]} x_{[i,j]}^\tau - \frac{1}{p} I) A_{[i,j-1]} = diag(\frac{1}{pu}) + (1 - \frac{1}{p}) A_{[i,j-1]} - \frac{1}{2\lambda_2} x_{[i,j]} x_{[i,j]}^\tau A_{[i,j-1]}$
6. end for
7. end for

Return $S = \frac{1}{S_1} \sum_{i=1}^{S_1} A_{[i,S_2]}$

Algorithm 4: LiSSA for dual formulation

Finally, we could obtain alo as $y_{alo} = y - 1/diag(A)$

3 LiSSA for Newton Step

In the following of the **LiSSA** algorithm, we denote β_E be the vector that contains the elements of β from the set E, $x_{i,E}$ be the i-th row and columns from the set E of A, $A.col(i)$ be the i-th column of A, and $A.row(i)$ be the i-th row of A. First let's recall the one-step Newton method for obtaining the alo and denoting E as the active set:

$$\beta^{/i} = \hat{\beta} - [\sum_{j \neq i} x_j x_j^\tau + \nabla^2 R(\beta)]^{-1} (y_i - x_i^\tau \hat{\beta}) x_i$$

Heuristically, I replace the Newton step by only updating the coefficients of the active set, thus:

$$\beta_E^{/i} = \hat{\beta}_E - [\sum_{j \neq i} x_{j,E} x_{j,E}^\tau + 2\lambda_2 I]^{-1}(y_i - x_{i,E}^\tau \hat{\beta}_E) x_{i,E}$$

Given the Hessian and gradient of the Newton Step, we derive the **LiSSA** algorithm as:

Initialize S_1 and S_2
Initialize $g = [-\frac{1}{n-1} X_E^\tau (X_E \beta_E - y)] \bigotimes \mathbf{1}^\tau$

1. for i = 1 to S_1
2. $A_{[i,0]} = g$
3. for j = 1 to S_2
4. sample x_k uniformly from all rows of X_E
5. $v = A.col(k)$
6. $A_{[i,j]} = g + (1 - 2\lambda_2) A_{[i,j-1]} - x_k x_k^\tau A_{[i,j-1]}$
7. $A.col(k) = v$

Return $S = \frac{1}{S_1} \sum_{i=1}^{S_1} A_{[i,S_2]}$

Algorithm 5: LiSSA for Newton Step

Furthermore, to finally obtain the alo, let $B = \beta_E \bigotimes \mathbf{1}^\tau - S$, we could obtain the alo prediction for the i-th sample as $y^{/i} = x_{i,E} B.col(i)$