Supplementary Material for AISTATS13 Submission #140

A Parallel, Block Greedy Method for Sparse Inverse Covariance Estimation for Ultra-high Dimensions

Prabhanjan Kambadur
IBM T.J. Watson Research Center
pkambadu@us.ibm.com

Aurélie Lozano
IBM T.J. Watson Research Center
aclozano@us.ibm.com

1 Forward Evaluator

Algorithm 1: ForwardEvaluator

Input: S: Sample covariance, Σ: Covariance, W: Inverse covariance, L: likelihood, bs: Block size.
Output: M: selected candidates

$M = MAP(L \rightarrow (row, col, \alpha), bs)$;
for $i = 1 : p$ **do**
  for $j = (i+1) : p$ **do**
    if $(i, j, *) \in W$ **then continue**;
    α_{ij} =ComputeAlpha$(S, \Sigma, (i, j))$;
    if $\alpha_{ij} == \infty$ **then continue**;
    $L_{ij} = L + \log\left((1 + \alpha_{ij}\sigma_{ij})^2 - \alpha_{ij}^2\sigma_{ii}\sigma_{jj}\right) - 2\alpha_{ij}s_{ij}$;
    if (Full(M) && $(L_{ij} \geq$MaxKey$(M))$ **then continue**;
    $M = (\text{Full}(M))?(M \setminus M[\text{MaxKey}(M)]) : M$;
    $M = M \bigcup (L_{ij} \rightarrow (i, j, \alpha_{ij}))$;
return M;

In the forward phase, GreedyInverseCovariance() selects the top bs candidates by calling ForwardEvaluator() (Algorithm 1). Briefly, this algorithm checks each of the (as yet unselected) positions in the upper (or lower) triangular matrix of W and returns a map of the top bs candidates. For each candidate, α_{ij} is computed and subsequently, the likelihood when $\alpha_{ij}(e_{ij} + e_{ji})$ is added to W is computed.

2 Backward Evaluator

To enhance the accuracy of our method, we execute a backward step at the end of each iteration; the goal of BackwardEvaluator() (Algorithm 2) is to remove those (i, j)'s, which contribute the least to L. Briefly, this algorithm checks each of the selected positions in W and returns a map of the bottom bs candidates. For each candidate, we evaluate L in the absence of the candidate, which is the same as adding a new candidate $(i, j, -\alpha_{ij})$ to W.

Algorithm 2: BackwardEvaluator

Input: S: Sample covariance, Σ: Covariance, W: Inverse covariance, L: likelihood, bs: Block size.
Output: M: selected candidates

$M = MAP(L \rightarrow (row, col, \alpha), bs)$;
for $i = 1 : p$ **do**
  for $j = (i+1) : p$ **do**
    if $(i, j, \alpha_{ij}) \notin W$ **then continue**;
    $\gamma_{ij} = -\alpha_{ij}$;
    $L_{ij} = L + \log\left((1 + \gamma_{ij}\sigma_{ij})^2 - \gamma_{ij}^2\sigma_{ii}\sigma_{jj}\right) - 2\gamma_{ij}s_{ij}$;
    if (Full(M) && $(L_{ij} \leq$MinKey$(M))$ **then continue**;
    $M = (\text{Full}(M))?(M \setminus M[\text{MinKey}(M)]) : M$;
    $M = M \bigcup (L_{ij} \rightarrow (i, j, \gamma_{ij}))$;
return M;

Algorithm 3: Update

Input: M: A map of selected candidates, $S_{p,p}$: Sample covariance, $\Sigma_{p,p}$: Covariance, W: Inverse covariance, L: likelihood
Output: $(W, \Sigma, L)^{new}$: Updated values

while $M \neq \emptyset$ **do**
  $(i, j, \alpha) =$ RemoveMaxKey(M);
  $L = L + \log\left((1 + \alpha_{ij}\sigma_{ij})^2 - \alpha_{ij}^2\sigma_{ii}\sigma_{jj}\right) - 2\alpha_{ij}s_{ij}$;
  $W = W + \alpha(e_{ij} + e_{ji})$;
  $\Sigma =$
  $\Sigma - \frac{\alpha(1+\alpha_{ij}\sigma_{ij})(\Sigma_i\Sigma_j^T + \Sigma_j\Sigma_i^T) + \alpha^2(\sigma_{jj}\Sigma_i\Sigma_i^T + \sigma_{ii}\Sigma_j\Sigma_j^T)}{(1+\alpha_{ij}\sigma_{ij})^2 - \alpha_{ij}^2\sigma_{ii}\sigma_{jj}}$;
return (W, Σ, L)

3 Updating state

Once the candidates are selected, the next step is to update L, W and Σ to reflect the addition/deletion of the new candidates by calling Update() (Algorithm 3). Briefly, for each new candidate that we will add/remove, we update the likelihood, add $\alpha(e_{ij}+e_{ji})$ to W, and update Σ ro reflect addition to W. Note that W is stored as (i,j,v) values and that candidates can be added in parallel.

4 Refit

Algorithm 4: Refit

Input: W: Inverse covariance, Σ: Covariance, L: likelihood, S: Sample covariance, ϵ: tolerance.
Output: $(W,\Sigma,L)^{new}$: refitted variables.

while *true* **do**
  for $i = 1 : p$ **do**
    for $j = (i+1) : p$ **do**
      if $(i,j,*) \notin W$ **then continue**;
      α_{ij} =ComputeAlpha$(S,\Sigma,(i,j))$;
      if $\alpha_{ij} == \infty$ **then continue**;
      $L_{ij} = L + \log\left((1+\alpha_{ij}\sigma_{ij})^2 - \alpha_{ij}^2\sigma_{ii}\sigma_{jj}\right) - 2\alpha_{ij}s_{ij}$;
      if $L_{ij} - L < \epsilon$ **then return** (W,Σ,L);
      $L = L_{ij}$; $W_{ij} = W_{ij} + \alpha_{ij}$; $W_{ji} = W_{ij}$;
      $\Sigma = \Sigma - \frac{\alpha(1+\alpha_{ij}\sigma_{ij})(\Sigma_i\Sigma_j^T + \Sigma_j\Sigma_i^T) + \alpha^2(\sigma_{jj}\Sigma_i\Sigma_i^T + \sigma_{ii}\Sigma_j\Sigma_j^T)}{(1+\alpha_{ij}\sigma_{ij})^2 - \alpha_{ij}^2\sigma_{ii}\sigma_{jj}}$;

After updating the required values, we re-estimate the value of the model parameters (W,Σ,L) to propogate the selection/removal of the current set of candidates using Refit () (Algorithm 4). Refitting is done using coordinate descent by iteratively re-estimating one position of W in each iteration; the procedure stops when the change in L is smaller than ϵ_r, the tolerance for refitting. Note that we only need to refit those entries in W, which share a common row or column with any of the chosen candidates, which is an optimization that is not shown in Algorithm 4.

5 Proof of Theorem 1

We consider a variant of GINCO, where the foward step breaks if $\delta_f = \sup_{(i,j)\in M} L - (L + \alpha_{i,j}e_{i,j}) < \epsilon_f$, where L is the log likelihood loss upon entering ForwardEvaluator() and M are the candidates selected by ForwardEvaluator(), and the backward step breaks if $\delta_b = \inf_{(i,j)\in M}(L - \alpha_{i,j}e_{i,j}) - L > \epsilon_b\delta_f$, where L is the log likelihood loss upon entering BackwardEvaluator() and M are the candidates selected by BackwardEvaluator(), and consider $\epsilon_b = 1/2$ for simplicity.

We adapt the reasoning in [1] so as to account for blocking. First, regardless of the block size, Lemma 1-3 in [2] insure that the strong convexity and strong smoothness assumptions required by [1] are satisfied. The proof then relies on the following lemmas, which are adapted from [1]. Denote by $\kappa_u = l$ and κ_u the restricted strong convexity and restricted strong smoothness parameters, respectively. These will be specialized in the theorem, using the assumptions on the restricted eigenvalues property. Let $\rho = \kappa_u/\kappa_l$.

Lemma 1 (Stopping Forward Step) *Let W^* and $\mathcal{S}^*$ denote the population inverse covariance matrix and its support, respectively. When GINCO stops with estimated inverse covariance $\hat{W}$, supported on $\hat{\mathcal{S}}$, there holds*

$$\left|L(\hat{W}) - L(\hat{W^*})\right| < \sqrt{2|\mathcal{S}^* - \hat{\mathcal{S}}|\kappa_u\epsilon_{\mathcal{S}}}\|\hat{W} - \hat{W^*}\|_2.$$

Proof The proof for Lemma 1 follows from the exact same arguments as that of Lemma 5 in [1], noting that the breaking condition for GINCO's forward step implies that when the algorithm terminates $L(\hat{W}) - \inf_{(i,j)\in\hat{\mathcal{S}}^c,\alpha} L(\hat{W} + \alpha e_{i,j}) < \epsilon_f$. □

Lemma 2 (Stopping Error Bound) *Let W^* and $\mathcal{S}^*$ denote the population inverse covariance matrix and its support, respectively. When GINCO stops with estimated inverse covariance $\hat{W}$, supported on $\hat{\mathcal{S}}$, there holds*

$$\|\hat{W} - \hat{W^*}\|_2 \le \frac{2}{\kappa_l}\left(c\sqrt{\frac{\log(p)}{n}}\sqrt{|\mathcal{S}^* \cup \hat{\mathcal{S}}|} + \sqrt{2|\mathcal{S}^* \cup \hat{\mathcal{S}}|\rho^2 C_{\min}^2\epsilon_f}\right).$$

Proof The proof follows by the same reasoning as for Lemma 6 in [1] and by noting that Lemma 3 in [2] implies that the l_∞ norm of the gradient of the loss at the true inverse covariance is upper-bounded by λ_n where $\lambda_n \le c\sqrt{\frac{\log(p)}{n}}$. □

Lemma 3 (Stopping Backward Step) *Let W^* and $\mathcal{S}^*$ denote the population inverse covariance matrix and its support, respectively. When GINCO stops with estimated inverse covariance $\hat{W}$, supported on $\hat{\mathcal{S}}$, there holds*

$$\|\hat{\Delta}_{\hat{\mathcal{S}}-\mathcal{S}^*}\|_2^2 \ge \frac{\epsilon_f}{\kappa_u}|\mathcal{S}^* - \hat{\mathcal{S}}|.$$

Proof The proof follows the same arguments as that of Lemma 7 in [1], noting that when GINCO terminates, the backward step with the current $\hat{W}$ has failed to go through, and hence there holds

$$\inf_{(i,j)\in\hat{\mathcal{S}}} L(\hat{W} - \hat{\alpha}_{i,j}e_{i,j}) - L(\hat{W}) > \epsilon_f\epsilon_b,$$

where $\epsilon_b = 1/2$. □

Consider the first time the support size reaches kb at the beginning of the forward step. Let $\Delta^{(k)} = \hat{W}^{(k)}_{\hat{\mathcal{S}}^{(k-1)}} - \hat{\bar{W}}^{(k-1)}$. Following a similar reasoning as in the proof of Lemma 9 in [1], we can show that $\|\Delta^{(k)}\|_2 \leq \frac{2\kappa_u}{\kappa_l^2}\sqrt{(\kappa_u - \kappa_l)b\delta_f^{(k)}}$, where b is the block size, and we obtain the following lemma.

Lemma 4 (General Backward Step) *T*he first time GINCO reaches a support size of $kb > |\mathcal{S}^*| + 4(\kappa_u/\kappa_l)^4 b + 1$ at the begining of the forward step we have

$$\|\hat{W}^{(k-1)}_{\hat{\mathcal{S}}^{(k-1)} - \mathcal{S}^*}\|_2^2 \geq \left(\sqrt{\frac{|\hat{\mathcal{S}}^{(k-1)} - \mathcal{S}^*|}{\kappa_u}} - \frac{2\kappa_u\sqrt{(\kappa_u - \kappa_l)b}}{\kappa_l^2}\right)^2 \delta_f^{(k)}.$$

The following lemma follows from Lemma 11 in [1].
Lemma 5 (General Error Bound) *The first time GINCO reaches a support size of kb at the begining of the forward step we have*

$$\|\hat{W}^{(k-1)}_{\hat{\mathcal{S}}^{(k-1)} - \mathcal{S}^*}\|_2^2 \leq \frac{4\kappa_u|\mathcal{S}^* \cup \hat{\mathcal{S}}^{(k-1)}|\delta_f^{(k)}}{\kappa_l^2}\left(\frac{c\sqrt{\log(p)/n}}{\sqrt{\kappa_u\epsilon_f}} + \sqrt{\frac{2|\hat{\mathcal{S}}^{(k-1)} - \mathcal{S}^*|}{|\mathcal{S}^* \cup \hat{\mathcal{S}}^{(k-1)}|}}\right)^2.$$

This allows us to obtain the last key lemma.

Lemma 6 (Stopping Size) *If*

$$\epsilon_f > \frac{\left(c\sqrt{\frac{\log(p)}{n}}\right)^2}{\kappa_u}\left(\frac{\frac{1}{2\rho}\sqrt{\gamma} - \sqrt{\frac{(\rho^2-\rho)b}{s^*}}}{\sqrt{1+\gamma}} - \sqrt{\frac{2}{2+\gamma}}\right)^{-2}$$

with $\gamma \geq 4\rho^2\left(\sqrt{\frac{(\rho^2-\rho)b}{s^*}} + \sqrt{2}\right)^2$, *then the algorithm stops with $kb \leq (1+\gamma)s^* + 1$, where s^* is the support size of the true inverse covariance.*

Proof Let $kb = (1+\gamma)s^*$, where s^* is the support size of the true inverse covariance. Following the reasoning of the proof of Lemma 8 in [1], and applying Lemma 4 and 5 above, we get

$$\frac{\frac{1}{2\rho}\sqrt{\gamma} - \sqrt{\frac{(\rho^2-\rho)b}{s^*}}}{\sqrt{1+\gamma}} - \sqrt{\frac{2}{2+\gamma}} \leq \frac{c\sqrt{\log(p)/n}}{\sqrt{\kappa_u\epsilon_f}}.$$

□

Then similarly as in Lemma 4 in [1] we can consider $\epsilon_f > c^2\log(p)/(n\kappa_u)\left(\sqrt{\frac{2}{1+\gamma}} - \sqrt{\frac{2}{2+\gamma}}\right)^{-2}$.

Theorem 1 then follows by combining Lemmas 1,2,3,6, using the same arguments as in the proof of Theorem 1 in [1]. □

References

[1] Ali Jalali, Christopher C. Johnson, and Pradeep D. Ravikumar. On learning discrete graphical models using greedy methods. *Advances in Neural Information Processing Systems (NIPS) and extended arxiv version*, 2011.

[2] Christopher C. Johnson, Ali Jalali, and Pradeep D. Ravikumar. High-dimensional sparse inverse covariance estimation using greedy methods. *Journal of Machine Learning Research - Proceedings Track*, 22:574–582, 2012.