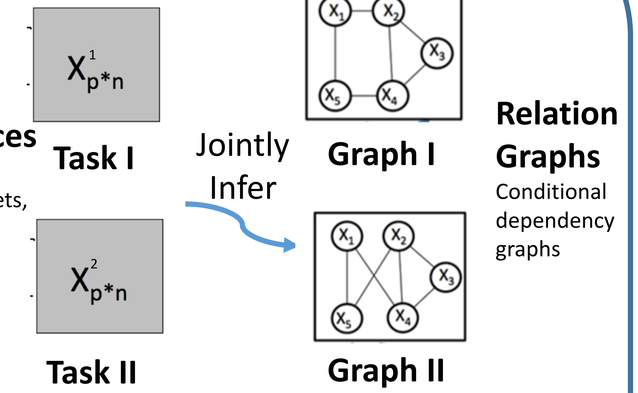
A Fast and Scalable Joint Estimator for Learning Multiple Related Sparse Gaussian Graphical Models

Beilun Wang, Ji Gao and Yanjun Qi

Department of Computer Science, University of Virginia

Introduction



Data matrices (e.g., gene expression datasets, TF ChIP-seq datasets)

Relation Graphs Conditional dependency graphs

Background: Number of feature (p) and Number of tasks (K) are increasing.

The past decade has seen a revolution in collecting largescale heterogeneous data from many scientific fields. For instance, genomic technologies have delivered fast and accurate molecular profiling data across many cellular contexts (e.g., cell lines or stages) from national projects like ENCODE[1].

	Previous	Now
p	Yeast data: 326 genes	Encode project: more than 30000 genes
K	Normal vs Cancer: 2	Encode Project: 147

Computational Complexity:

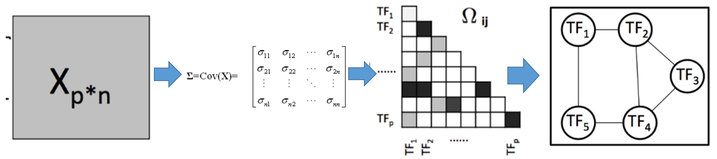
$O(Kp^3) \approx 10^{15}$ multiplication

Memory:

$O(Kp^2) \approx$ 320GB for 147 graphs

Background: sGGM to derive Conditional Independence Graph from data.

Sparse Gaussian Graphical Model is solved by the following three steps: (1) Calculate the sample covariance matrix; (2) estimate the sparse inverse of covariance matrix; (3) extract sparsity pattern in the inverse of covariance matrix. The solution of the second step includes: gLasso, neighborhood selection or Elementary Estimator.



FASJEM: A Fast and Scalable Joint Estimator for Learning Multiple Related Sparse Gaussian Graphical Models

We propose a novel approach, **FASJEM** for fast and scalable joint structure-estimation of multiple sGGMs at a large scale. As the first study of joint sGGM using the M-estimator framework, our work has three major Advantages: (1) Highly parallelizable; (2) Fast and memory efficient; (3) Achieves consistent convergence rate.

Elementary Estimator:	FASJEM:
$\widehat{\Sigma} = (X-\mu)(X-\mu)^T$	$\widehat{\Sigma}^{(i)} = (X^{(i)}-\mu^{(i)})(X^{(i)}-\mu^{(i)})^T$
$\underset{\Omega}{\operatorname{argmin}} \|\|\Omega\|\|_1$ subject to: $\|\|\Omega - [T_v(\widehat{\Sigma})]^{-1}\|\|_\infty \le \lambda_n$	$\underset{\Omega_{tot}}{\operatorname{argmin}} \|\|\Omega_{tot}\|\|_1 + \epsilon\mathcal{R}'(\Omega_{tot})$ $s.t. \|\|\Omega_{tot} - \operatorname{inv}(T_v(\widehat{\Sigma}_{tot}))\|\|_\infty \le \lambda_n$ $\mathcal{R}'^*(\Omega_{tot} - \operatorname{inv}(T_v(\widehat{\Sigma}_{tot}))) \le \epsilon\lambda_n$

Computational complexity and Memory improvements:

References	Computational Complexity	Memory Cost
JGL-Group [8]	$O(Kp^3)$	$O(Kp^2)$
JGL-GroupInf [11]	$O(K^3p^4)$	$O(Kp^2)$
FASJEM Models	$O(Kp^2)$ (if paralleling completely, $O(K)$)	$O(K)$

Theoretical Analysis

Elementary Estimator: $O(\log p/n_i)$ FASJEM: $O(\log(Kp)/n_{tot})$

Convergence rate

Algorithm:

We choose parallel proximal algorithm to solve the problem.

This method can be easy to parallelize in GPU, and therefore achieve much smaller computational complexity and memory cost.

Algorithm 1 Parallel proximal algorithm[4]

input K given data blocks $X^{(1)}, X^{(2)}, \ldots, X^{(K)}$. Hyper-parameter:$\alpha$, ϵ, v, λ_n and γ. Learning rate: $0 < \rho < 2$. Max iteration number $iter$.

output Ω_{tot}

Compute Σ_{tot} from $X^{(1)}, X^{(2)}, \ldots, X^{(K)}$

Initialize $\theta^0 = \operatorname{inv}(T_v(\Sigma_{tot}))$, $\theta^0_j = \operatorname{inv}(T_v(\Sigma_{tot}))$ for $j \in \{1,2,3,4\}$ and $a = \operatorname{inv}(T_v(\Sigma_{tot}))$.

for $i = 0$ **to** $iter$ **do**

  $p^i_1 = \operatorname{prox}_{4\gamma f_1}\theta^i_1$

  $p^i_2 = \operatorname{prox}_{4\gamma f_2}\theta^i_2$

  $p^i_3 = \operatorname{prox}_{4\gamma f_3}\theta^i_3$

  $p^i_4 = \operatorname{prox}_{4\gamma f_4}\theta^i_4$

  $p^i = \frac{1}{4}(\sum_{j=1}^{4}\theta^i_j)$

  for $j = 1,2,3,4$ **do**

    $\theta^{i+1}_j = \theta^i_j + \rho(2p^i - \theta^i - p^i_j)$

  end for

  $\theta^{i+1} = \theta^i + \rho(p^i - \theta^i)$

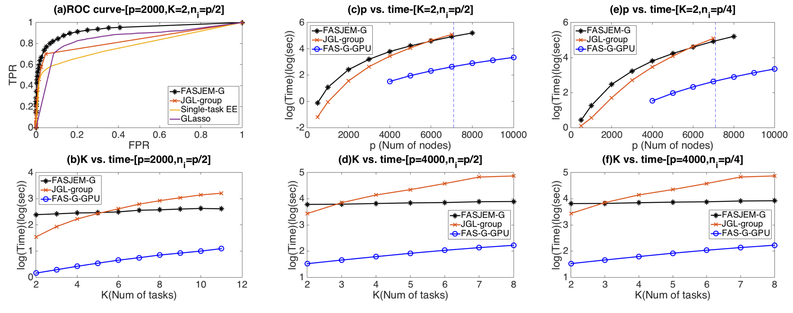
end for

$\Omega_{tot} = \theta^{iter}$

output Ω_{tot}

Experiment Evaluation

We simulate multiple related Gaussian datasets with known Graphs.
(1) Draw a FPR vs. TPR curve. Compare the AUC score;
(2) Compare computation time with different p and K.
FASJEM and FASJEM-GPU (GPU version of FASJEM) achieve the best AUC score and spend least computation time.



Acknowledgement

This work was supported by the National Science Foundation under NSF CAREER award No. 1453580

References

[1] ENCODE Project Consortium et al. An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489(7414):57–74, 2012.