GLAD: Learning Sparse Graph Recovery

Le Song

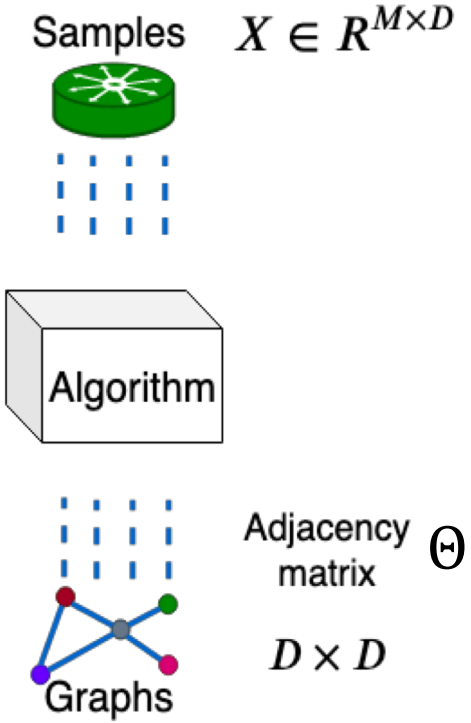
Georgia Tech

Joint work with Harsh Shrivastava, Xinshi Chen, Binghong Chen, Guanghui Lan

Objective

Recovering sparse conditional independence graph G from data

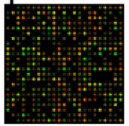
$$\Theta_{ij} = 0 \Leftrightarrow X_i \perp X_j | \, other \; variables$$

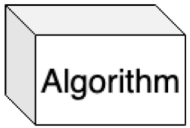


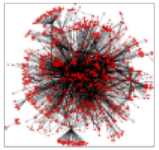
Applications

Biology

Gene Expression data - Microarray experiments



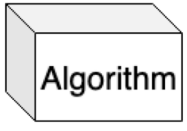




Gene regulatory network

Finance

Time-series features



Relationship between assets

Convex Formulation

- Given M samples from a distribution: $X \in R^{M \times D}$
- Estimate the matrix ‘Θ’ corresponding to the sparse graph

Objective function: L1 regularized log-determinant estimation

$$\widehat{\Theta} = \arg\min_{\Theta \in \mathcal{S}^d_{++}} \; -\log(\det \Theta) + \mathrm{tr}(\widehat{\Sigma}\Theta) + \rho \|\Theta\|_{1,\mathrm{off}}$$

Covariance matrix

$$\hat{\Sigma} = \frac{X^\top X}{M}$$

Regularization Parameter

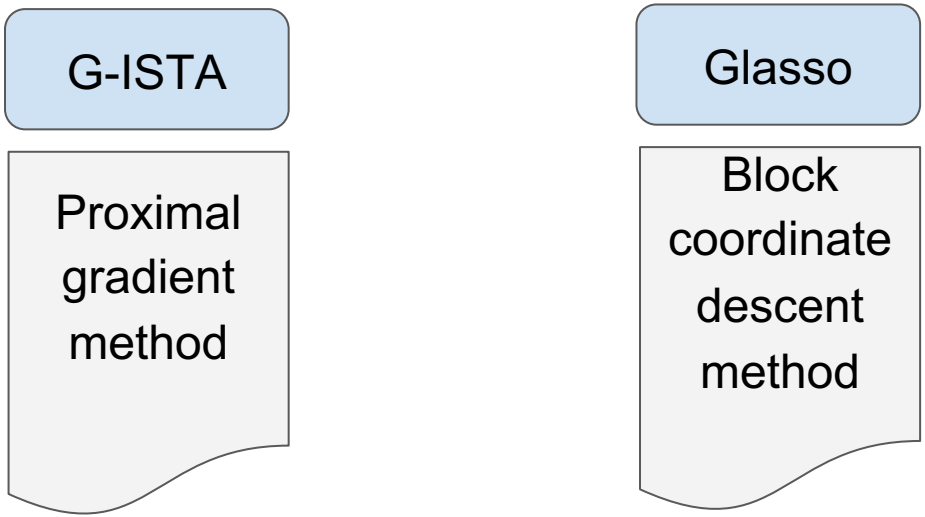
Existing Optimization Algorithms

G-ISTA

Proximal gradient method

$$\Theta_{k+1} \leftarrow \eta_{\xi_k \rho}(\Theta_k - \xi_k(\widehat{\Sigma} - \Theta_k^{-1})), \quad \text{where} \quad [\eta_\rho(X)]_{ij} := \text{sign}(X_{ij})(|X_{ij}| - \rho)_+$$

Existing Optimization Algorithms



Updates each column (and the corresponding row) of the precision matrix iteratively by solving a sequence of lasso problems

Existing Optimization Algorithms

G-ISTA

Proximal gradient method

Glasso

Block coordinate descent method

ADMM

Alternating direction method of multipliers

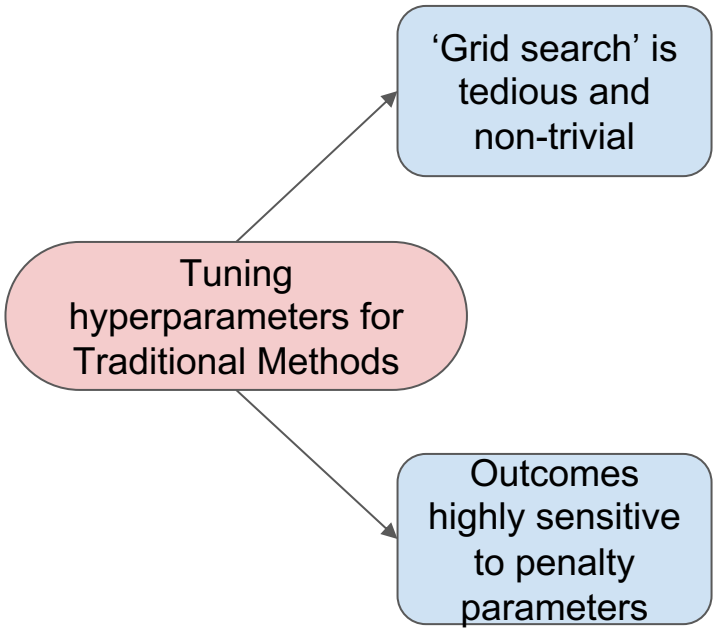
$$-\log(\det\Theta) + \mathrm{tr}(\widehat{\Sigma}\Theta) + \rho\,\|Z\|_1 + \langle\lambda, \Theta - Z\rangle + \tfrac{1}{2}\beta\|Z-\Theta\|_F^2.$$

Taking $U := \lambda/\beta$ as the scaled dual variable, the update rules for the ADMM algorithm are

$$\Theta_{k+1} \leftarrow \big(-Y + \sqrt{Y^\top Y + (4/\beta)I}\big)/2, \text{ where } Y = \widehat{\Sigma}/\beta - Z_k + U_k$$

$$Z_{k+1} \leftarrow \eta_{\rho/\beta}(\Theta_{k+1} + U_k), \quad U_{k+1} \leftarrow U_k + \Theta_{k+1} - Z_{k+1}$$

Hard to Tune Hyperparameters

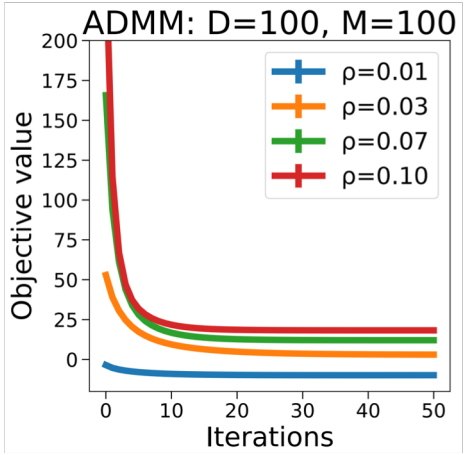


ρ \ β	5	1	0.5	**0.1**	0.01
0.01	-2.51	-2.25	-2.06	-2.06	-2.69
0.03	-5.59	-9.05	9.48	**-9.61**	-9.41
0.07	-9.53	-7.58	-7.42	-7.38	-7.46
0.1	-9.38	-6.51	-6.43	-6.41	-6.50
0.2	-6.76	-4.68	-4.55	-4.47	-4.80

Errors of different parameter combinations

$$-\log(\det\Theta) + \mathrm{tr}(\widehat{\Sigma}\Theta) + \rho\,\|Z\|_1 + \langle\lambda, \Theta - Z\rangle + \tfrac{1}{2}\beta\|Z-\Theta\|_F^2.$$

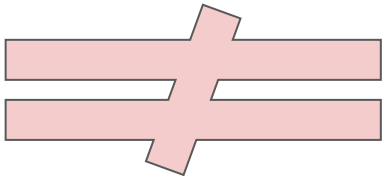
Mismatch in Objectives

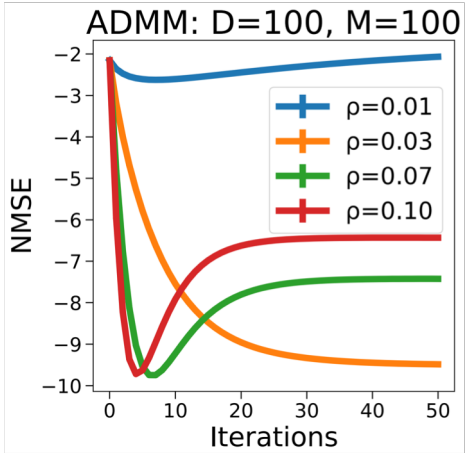


Log-determinant estimator

$$-\log(\det\Theta) + \mathrm{tr}(\widehat{\Sigma}\Theta) + \rho\,\|\Theta\|_{1,\mathrm{off}}$$

mismatch!

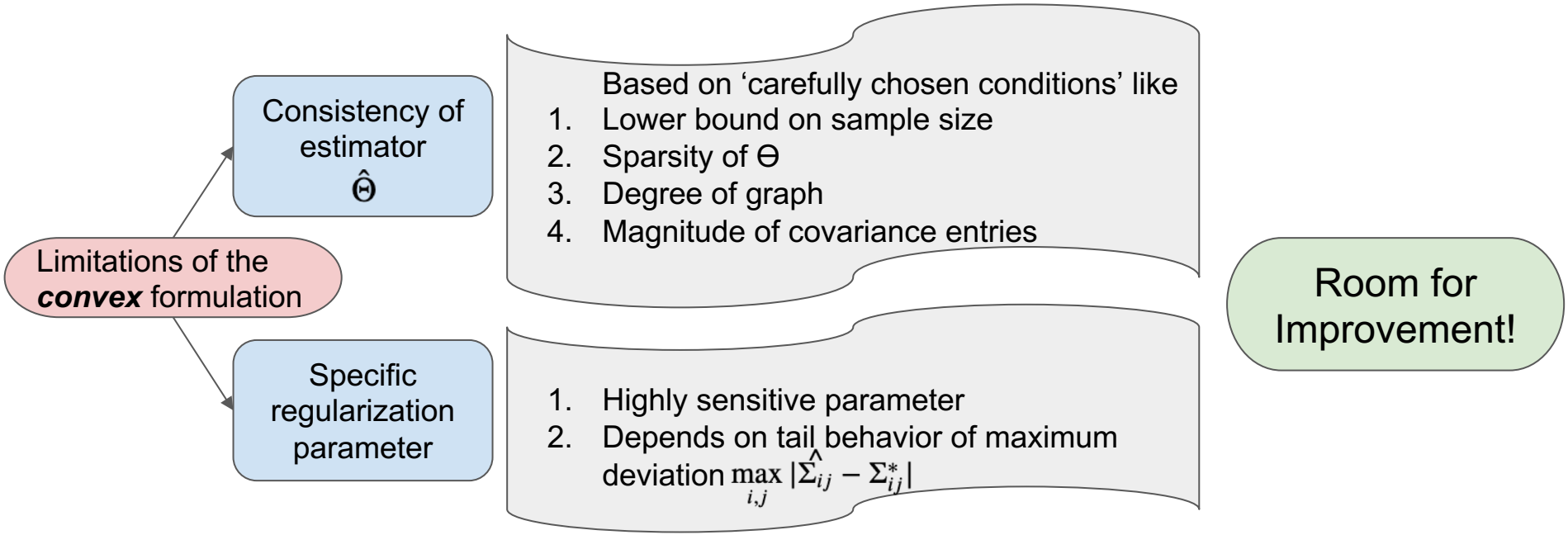




Recovery Objective (NMSE)

$$\|\widehat{\Theta} - \Theta^*\|_F^2 / \|\Theta^*\|_F^2$$

Limitations of Existing Optimization Algorithms



Pradeep Ravikumar, Martin J Wainwright, Garvesh Raskutti, Bin Yu, et al. High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. Electronic Journal of Statistics, 5:935–980, 2011.

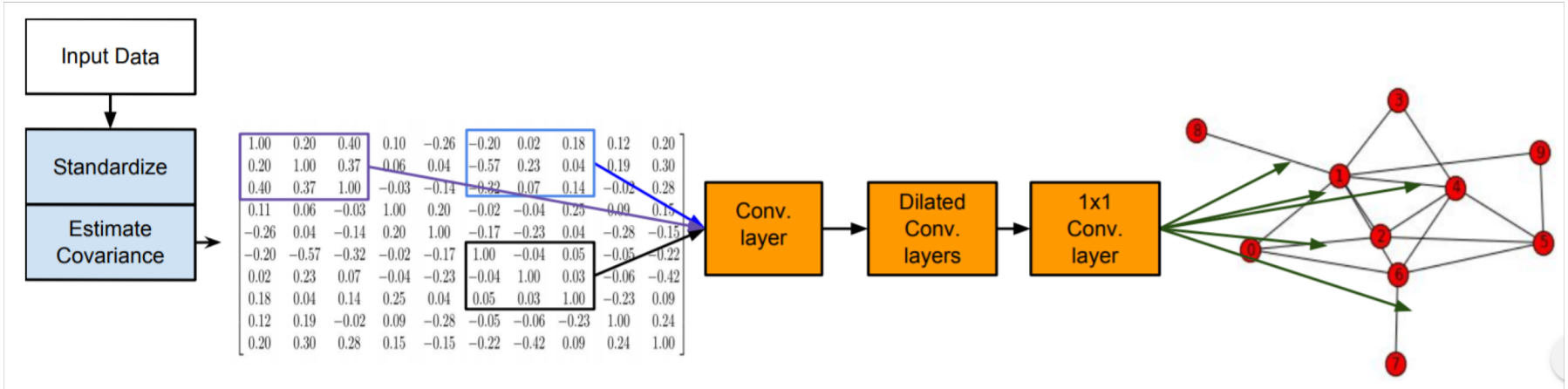
Big Picture Question

- Given a collection of ground truth precision matrix Θ^*, and the corresponding empirical covariance $\hat{\Sigma}$

- Learn an algorithm f which directly produces an estimate of the precision matrix Θ?

$$\min_f \frac{1}{|\mathcal{D}|} \sum_{(\hat{\Sigma}_i, \Theta_i^*) \in \mathcal{D}} \|\Theta_i - \Theta_i^*\|_F^2, \qquad s.t.\ \Theta_i = f(\hat{\Sigma}_i)$$

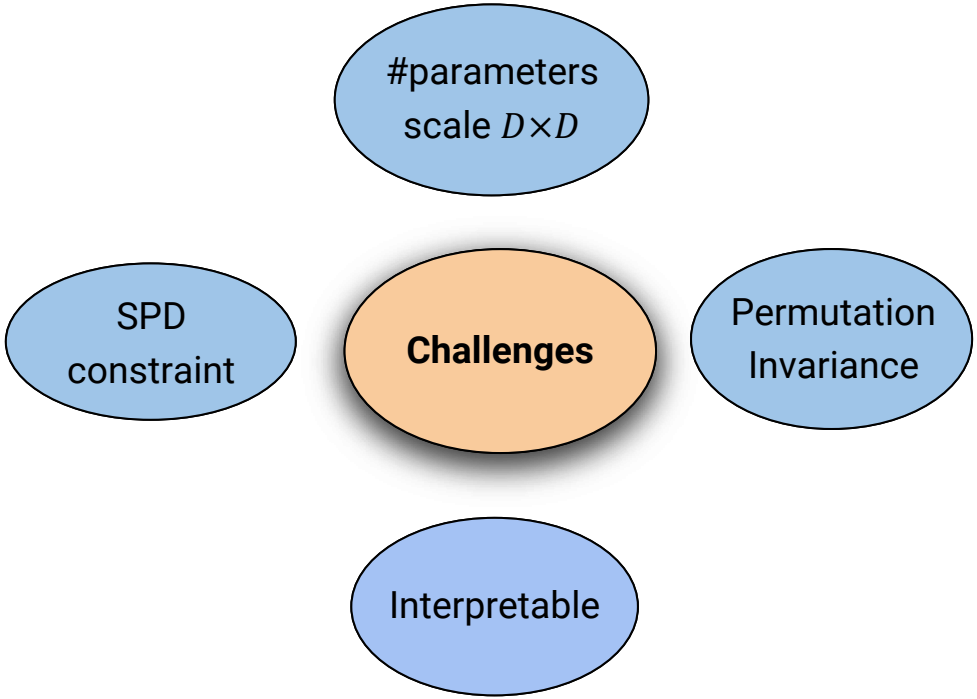
Deep Learning Model Example



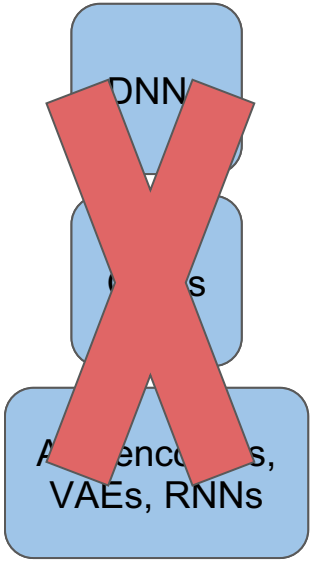
DeepGraph (DG)` architecture.The input is first standardized and then the sample covariance matrix is estimated. A neural network consisting of multiple dilated convolutions (Yu & Koltun, 2015) and a final 1 × 1 convolution layer is used to predict edges corresponding to non-zero entries in the precision matrix.

* DeepGraph-39 model from Fig.2 of “Learning to Discover Sparse Graphical Models” by Belilovsky et. al.

Challenges in Designing Learning Models



Traditional Approaches



GLAD: DL model based on Unrolled Algorithm

Alternating Minimization (AM) algorithm: Objective function

$$\widehat{\Theta}_\lambda, \widehat{Z}_\lambda := \arg\min_{\Theta, Z \in \mathcal{S}^d_{++}} -\log(\det \Theta) + \mathrm{tr}(\widehat{\Sigma}\Theta) + \rho \|Z\|_1 + \frac{1}{2}\lambda \|Z - \Theta\|_F^2$$

AM: Update Equations (Nice closed form updates!)

$$\Theta^{\mathrm{AM}}_{k+1} \leftarrow \frac{1}{2}\Big(-Y + \sqrt{Y^\top Y + \frac{4}{\lambda} I}\Big), \text{ where } Y = \frac{1}{\lambda}\widehat{\Sigma} - Z^{\mathrm{AM}}_k$$

$$Z^{\mathrm{AM}}_{k+1} \leftarrow \eta_{\rho/\lambda}(\Theta^{\mathrm{AM}}_{k+1}), \quad \text{where } \eta_{\rho/\lambda}(\theta) := \mathrm{sign}(\theta)\max(|\theta| - \rho/\lambda, 0)$$

- Unroll to fixed #iterations ‘K’.
- Treat it as a deep model

GLAD: Training

Loss function: Frobenius norm with discounted cumulative reward

$$\min_f \ \text{loss}_f := \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\gamma^{K-k}\left\|\Theta_k^{(i)} - \Theta^{*(i)}\right\|_F^2$$

Optimizer for training: 'Adam'. Learning rate chosen between [0.01, 0.1] in conjunction with Multi-step LR scheduler.

Gradient Computation through matrix square root in the GLADcell:

For any SPD matrix *X:* $X = X^{1/2}X^{1/2}$

Solve Sylvester's equation for $d(X^{1/2})$:

$$dX = d(X^{1/2})X^{1/2} + X^{1/2}d(X^{1/2})$$

Use Neural Networks for (ρ, λ)

$$\lambda \leftarrow \Lambda_{nn}(\|Z - \Theta\|_F^2, \lambda)$$

$$\rho_{ij} = \rho_{nn}(\Theta_{ij}, \hat{\Sigma}_{ij}, Z_{ij})$$

# of layers = **2**
Hidden unit size = **3**

Minimalist designing of Neural Networks

# of layers = **4**
Hidden unit size = **3**

Non-Linearity:
Hidden layers = '*tanh*'
Final layer = '*sigmoid*'

GLAD

Algorithm 1: GLAD

Function `GLADcell`$(\widehat{\Sigma}, \Theta, Z, \lambda)$**:**

$\lambda \leftarrow \Lambda_{nn}(\|Z - \Theta\|_F^2, \lambda)$

$Y \leftarrow \lambda^{-1}\widehat{\Sigma} - Z$

$\Theta \leftarrow \frac{1}{2}\left(-Y + \sqrt{Y^\top Y + \frac{4}{\lambda} I}\right)$

For *all* i, j **do**

$\rho_{ij} = \rho_{nn}(\Theta_{ij}, \widehat{\Sigma}_{ij}, Z_{ij})$

$Z_{ij} \leftarrow \eta_{\rho_{ij}}(\Theta_{ij})$

return Θ, Z, λ

Function `GLAD`$(\widehat{\Sigma})$**:**

$\Theta_0 \leftarrow (\widehat{\Sigma} + tI)^{-1}, \lambda_0 \leftarrow 1$

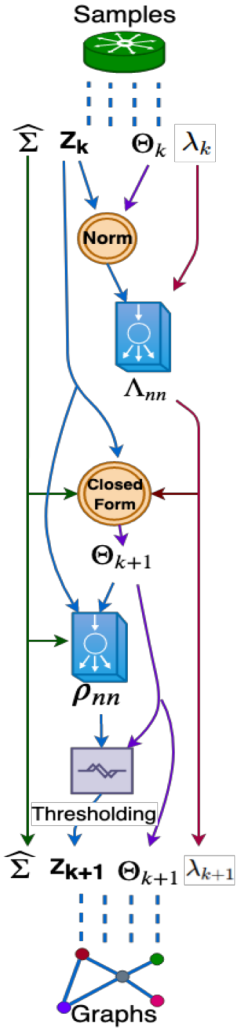
For $k = 0$ *to* $K - 1$ **do**

$\Theta_{k+1}, Z_{k+1}, \lambda_{k+1}$

$\leftarrow$`GLADcell`$(\widehat{\Sigma}, \Theta_k, Z_k, \lambda_k)$

return Θ_K, Z_K

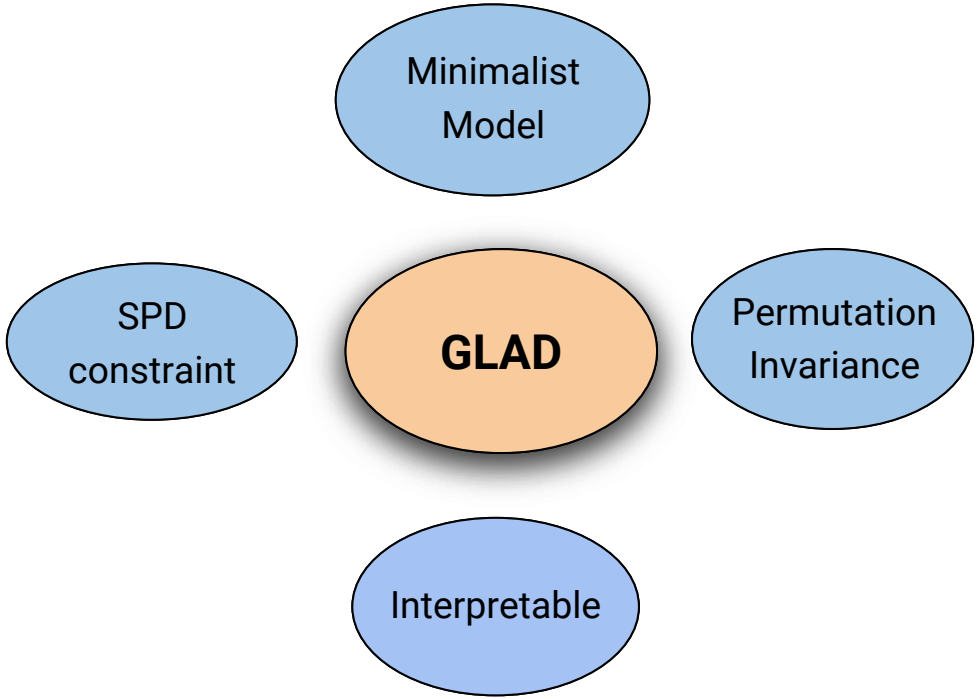
GLADcell



GLAD

Using algorithm structure as inductive bias for designing unrolled DL architectures

Desiderata for GLAD



Algorithm 1: GLAD

Function `GLADcell`$(\widehat{\Sigma}, \Theta, Z, \lambda)$**:**

$\lambda \leftarrow \Lambda_{nn}(\|Z - \Theta\|_F^2, \lambda)$

$Y \leftarrow \lambda^{-1}\widehat{\Sigma} - Z$

$\Theta \leftarrow \frac{1}{2}\left(-Y + \sqrt{Y^\top Y + \frac{4}{\lambda}I}\right)$

For *all* i, j **do**

$\rho_{ij} = \rho_{nn}(\Theta_{ij}, \widehat{\Sigma}_{ij}, Z_{ij})$

$Z_{ij} \leftarrow \eta_{\rho_{ij}}(\Theta_{ij})$

return Θ, Z, λ

Function `GLAD`$(\widehat{\Sigma})$**:**

$\Theta_0 \leftarrow (\widehat{\Sigma} + tI)^{-1}, \lambda_0 \leftarrow 1$

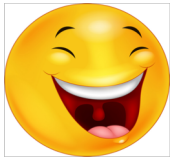
For $k = 0$ *to* $K - 1$ **do**

$\Theta_{k+1}, Z_{k+1}, \lambda_{k+1}$

$\leftarrow$`GLADcell`$(\widehat{\Sigma}, \Theta_k, Z_k, \lambda_k)$

return Θ_K, Z_K

GLAD: Graph recovery **L**earning **A**lgorithm using **D**ata-driven training



Experiments: Convergence

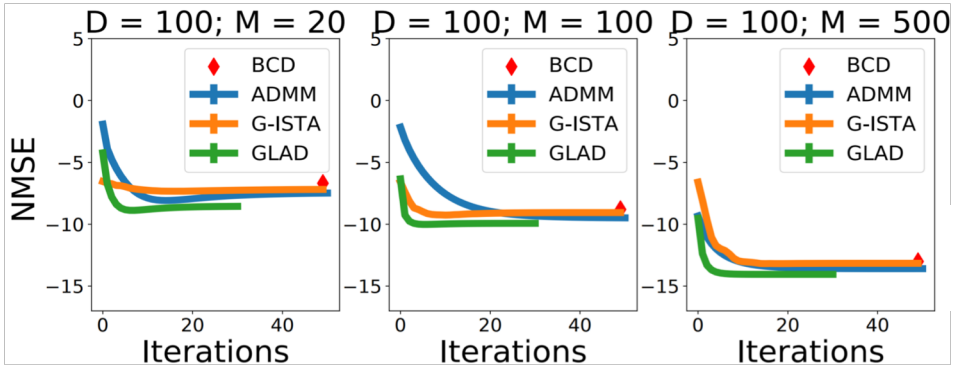
Train/finetuning using 10 random graphs

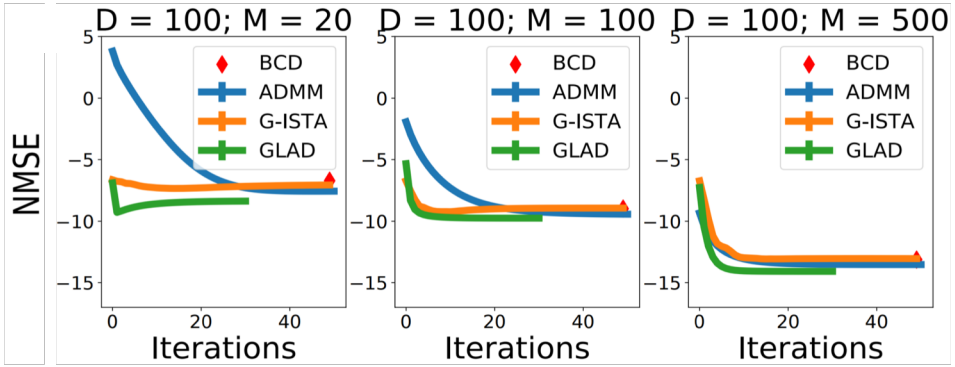
Test on 100 random graphs

GLAD vs traditional methods

Fixed Sparsity level s=0.1

Mixed Sparsity level $s \backsim U(0.05, 0.15)$



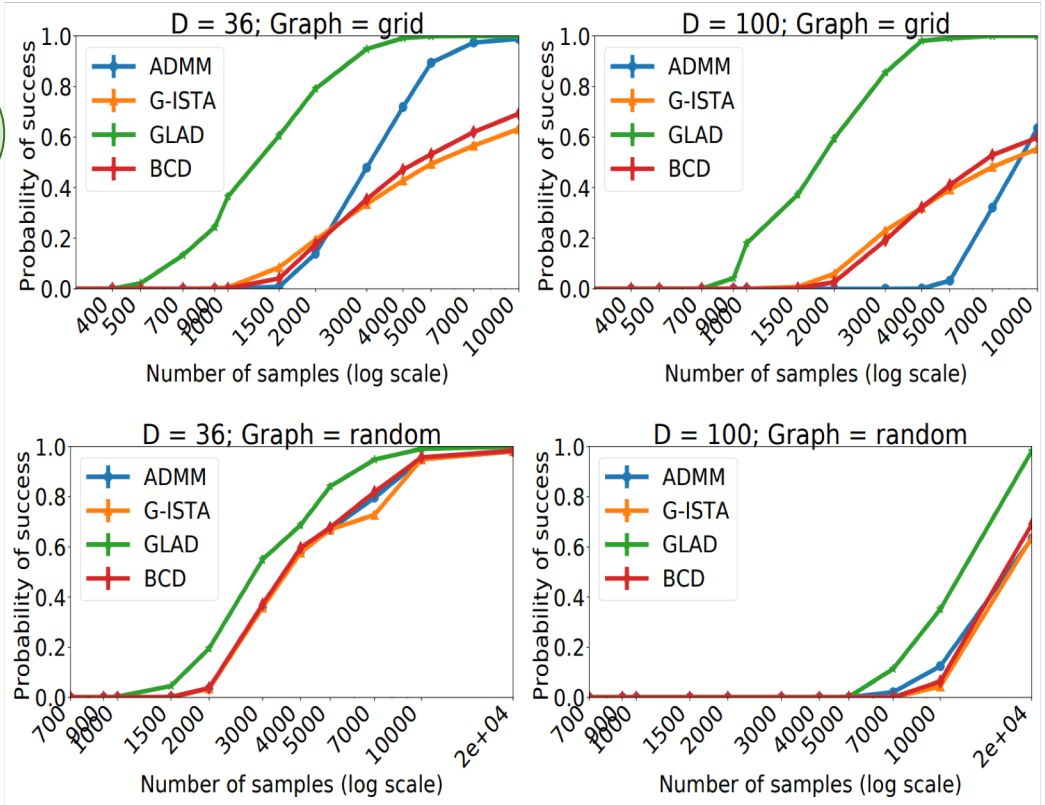


Experiments: Recovery probability

Sample complexity for model selection consistency

PS is non-zero if all graph edges are recovered with correct signs

GLAD able to recover true edges with considerably fewer samples



Experiments: Data Efficiency (cont...)

GLAD vs CNN*

Training graphs
100 vs 100,000

# of parameters
<25 vs >>>25

Runtime
< 30 mins vs
several hours

Methods	M=15	M=35	M=100
BCD	0.578±0.006	0.639±0.007	0.704±0.006
CNN	0.664±0.008	0.738±0.006	0.759±0.006
CNN+P	0.672±0.008	0.740±0.007	0.771±0.006
GLAD	0.788±0.003	0.811±0.003	0.878±0.003

AUC` on 100 test graphs, Gaussian random graph sparsity=0.05 and edge values sampled from ~U(-1, 1).

* DeepGraph-39 model from “Learning to Discover Sparse Graphical Models” by Belilovsky et. al.

` Table 1. of Belilovsky et. al.

Gene Regulation Data: SynTReN details

Synthetic gene expression data generator creating biologically plausible networks

Models biological & correlation noises

SynTReN

The topological characteristics of generated networks closely resemble transcriptional networks

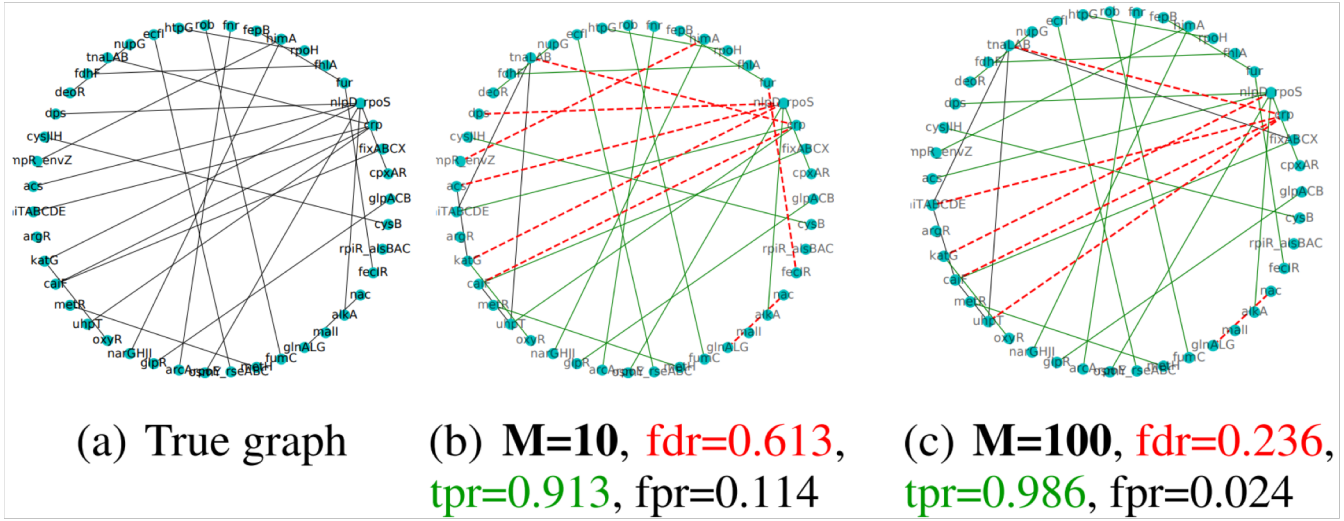
Contains instances of Ecoli bacteria and other true interaction networks

Gene Regulation Data: Ecoli Network Predictions

GLAD trained on Erdos-Renyi graphs of dimension=25.

# of train/valid graphs were 20/20.

M samples were generated per graph



(a) True graph

(b) **M=10**, fdr=0.613, tpr=0.913, fpr=0.114

(c) **M=100**, fdr=0.236, tpr=0.986, fpr=0.024

Recovered graph structures for a sub-network of the E. coli consisting of 43 genes and 30 interactions with increasing samples. All noises sampled ~U(0.01, 0.1)
Increasing the samples reduces the fdr by discovering more true edges.

Theoretical Analysis: Assumptions

Assumption 1 (Tail conditions). *There exists* $v_* \in (0, \infty]$ *and a function* $h : \mathbb{N} \times (0, \infty) \to (0, \infty)$ *such that* $\forall i, j, \mathbb{P}[|\widehat{\Sigma}^m_{ij} - (\Theta^*)^{-1}_{ij}| \geq \delta] \leqslant 1/h(m, \delta), \ \forall \delta \in (0, 1/v_*]$. *Further, assume* h *is monotonically increasing in sample size* m *and* δ *and define* $\bar{\delta}_h(m, r) := \arg\max \{\delta | h(m, \delta) \leqslant r\}$.

Ensures that sample sizes are large enough for an accurate estimation of the covariance matrix

Assumption 2 (Incoherence condition). *Denote the Hession by* $\Gamma^* := \Theta^{*-1} \otimes \Theta^{*-1}$, *the indices of nonzero entries by* $S := \{(i, j) | \Theta^*_{ij} \neq 0\}$ *and its complement set by* S^c. *There exists* $\alpha \in (0, 1]$ *such that* $\max_{e \in S^c} \left\| \Gamma^*_{eS} (\Gamma^*_{SS})^{-1} \right\|_1 \leqslant (1 - \alpha)$.

Restricts the interaction between edge and non-edge terms in the precision matrix

Consistency Analysis

Recalling AM

$$\Theta_{k+1}^{\mathrm{AM}} \leftarrow \frac{1}{2}\Big(- \qquad \frac{1}{\lambda}\widehat{\Sigma} - Z_k^{\mathrm{AM}}$$

$$Z_{k+1}^{\mathrm{AM}} \leftarrow \eta_{\rho/\lambda}(\qquad (|\theta| - \rho/\lambda, 0)$$

An adaptive sequence of penalty parameters should achieve a better error bound

Summary

Optimal parameter values depends on the tail behavior and the prediction error

Hard to choose these parameters manually

ssumption 1 and 2, suppose th $, d^{\tau})$

he sample size is larger than

Then with probability at least $1 - 1/d^{\tau-2}$,

$$\left\|\Theta_k^{AM} - \Theta^*\right\| \lesssim C \qquad \lambda^{-1/2} + \mathcal{O}\left(\delta_h(m, d^{\tau})\lambda^{-1/2}\right) + \mathcal{O}\left(\delta_h(m, d^{\tau})\right)$$

where $C_{\rho,\lambda,\Theta_0^{AM},Z_0^{AM}}$ *initialization* $\|\Theta_0^{AM} - \widehat{\Theta}_\lambda\|_F$ *and* λ.

Conclusion

Unrolled DL architecture, GLAD, for sparse graph recovery

Empirically, GLAD is able to reduce sample complexity

Empirical evidence that learning can improve graph recovery

Highlighting the potential of using algorithms as inductive bias for DL architectures

Thank you!