DAGs with NO TEARS: Continuous Optimization for Structure Learning

Pradeep Ravikumar
Machine Learning Department
School of Computer Science
Carnegie Mellon University

Joint work with: Xun Zheng, Bryon Aragam, Eric Xing

Graphical Models

- Graphical Models are families of multivariate distributions with **compact representations**, scaling to very large numbers of variables

- Why do we need compact representations?

    - even for binary random variables, specification of a general multivariate distribution over p variables requires $O(2^p)$ values

        - In general, parametrizing higher-order dependencies among random variables scales poorly (typically exponentially) with number of variables

        - Graphical models only require “local” specifications based on graph neighborhoods (associating variables to graph nodes), and hence scale well to large number of variables

- Undirected graphical models, also called Markov networks, or Markov random fields: family represented by an undirected graph

- Directed graphical models, also called Bayesian networks: family represented by a directed acyclic graph (DAG)

Directed Graphical Models

$X = (X_1, \ldots, X_p) \sim$ directed graphical model with DAG $G = (V, E)$ if:

$$P(X; G) = \prod_{j=1}^{p} P(X_j | X_{\mathrm{pa}_j}),$$

- pa_j is the set of parents of node $j \in V$
- associating variables X_j, for $j \in [p]$, with nodes $j \in V$

- Only requires **local** specifications of conditional distributions of variables given its "parent" variables

Directed Graphical Models

The DAG G encodes the conditional independence assumptions satisfied by resulting distributions simply as:

$$X_j \perp\!\!\!\perp X_{\mathrm{nd}_j} | X_{\mathrm{pa}_j}, \forall j \in V$$

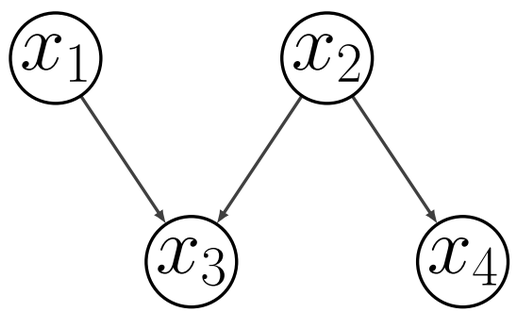
- pa_j is set of parents of node $j \in V$
- nd_j is set of non-descendants of node $j \in V$

- Edges connote “direct dependence” that is more meaningful than high correlation; underlying DAG G an object of interest even when the full multivariate distribution is not

- Applications across biology (Sachs et al., 2005), genetics (Zhang et al., 2013), causal inference (Spirtes et al., 2000), artificial intelligence (Koller and Friedman, 2009), and many more

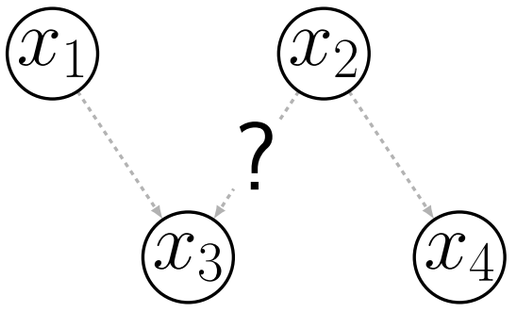
Learning DAGs

- Graphical models: compact models of $p(x_1, \ldots, x_d)$



$$\xrightarrow{sample} \begin{pmatrix} 4.00 & -1.14 & 0.20 & -2.37 \\ -1.05 & 0.35 & -0.66 & -0.39 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

- Structure learning: what graph fits the data best?



$$\xleftarrow{estimate} \begin{pmatrix} 4.00 & -1.14 & 0.20 & -2.37 \\ -1.05 & 0.35 & -0.66 & -0.39 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Learning DAGs

- Two main classes of approaches
- Conditional Independence Test based
    - test which conditional independences hold in the data, find graph that best corresponds to these
        - caveats: many more conditional independences might hold in data (“lack of faithfulness”), sensitive to failure of individual tests + multiplicity of tests; computationally less scalable
- Score based
    - search for graph that optimizes some score (measuring goodness of fit of graph to data)
    - typically local search/greedy algorithms that greedily build graph
    - NP-hard in general, need many model specific heuristics to “get it to work”

Learning DAGs

- Two main classes of approaches
- Conditional Independence Test based
    - test which conditional independences hold in the data, find graph that best corresponds to these
        - caveats: many more conditional independences might hold in data (“lack of faithfulness”), sensitive to failure of individual tests + multiplicity of tests; computationally less scalable
- Score based

    “The disadvantage of the score-based approaches (for Bayesian networks) is that they pose a search problem that may not have an elegant and efficient solution” …. Koller and Friedman (2009, pp. 785)

    - NP-hard in general, need many model specific heuristics to “get it to work”

Learning Bayesian Networks vs Markov Networks

	Markov Networks	Bayesian Networks
Cond. Indep. test based	folklore	Spirtes and Glymour (1991)
Score based (local search)	Pietra et al (1997)	Heckerman et al (1995)
Continuous optimization (global)	Meinshausen, Buhlmann 2006, Ravikumar et al (2010, 2011), …	?

Learning DAGs: Problem Setup

- Bayesian network (BN) G with d nodes:

$$p_{joint}(x_1, \ldots, x_d; G) = \prod_{j=1}^{d} p_{cond}(x_j | \boldsymbol{x}_{pa(j)})$$

Structural Equation Models (SEMs) specify the form of these conditional distributions:

$$X_j = T(X_{\mathrm{pa}_j}, \epsilon_j)$$
$$\epsilon_j \sim \text{noise distribution}$$

We will be considering SEMs parameterized by a weighted adjacency matrix W

Linear SEMs

- Weighted adjacency matrix $W \in \mathbb{R}^{d \times d}$:

$$W = \begin{bmatrix} w_{1\to 1} & \cdots & w_{1\to j} & \cdots & w_{1\to d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{d\to 1} & \cdots & w_{d\to j} & \cdots & w_{d\to d} \end{bmatrix}$$

  The j-th column $\boldsymbol{w}_j$: edge weights from $pa(j)$ to j.

- Linear BN:

$$x_j = T(\boldsymbol{x}_{pa(j)}, \varepsilon_j) = \underbrace{\boldsymbol{x}^T \boldsymbol{w}_j}_{linear} + \underbrace{\varepsilon_j}_{zero\ \ mean}$$

Generalized Linear SEMs

- Weighted adjacency matrix $W \in \mathbb{R}^{d \times d}$:

$$W = \begin{bmatrix} w_{1\to 1} & \cdots & w_{1\to j} & \cdots & w_{1\to d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{d\to 1} & \cdots & w_{d\to j} & \cdots & w_{d\to d} \end{bmatrix}$$

  The j-th column $\boldsymbol{w}_j$: edge weights from $pa(j)$ to j.

- Generalized Linear BN:

$$P(x_j \mid \boldsymbol{x}_{pa(j)}) = h(x_j)\, \exp\left(g(x_j)\,(\boldsymbol{x}^T \boldsymbol{w}_j) - A(\boldsymbol{x}^T \boldsymbol{w}_j)\right),$$

  so that:

$$E(x_j \mid \boldsymbol{x}_{pa(j)}) = T(\underbrace{\boldsymbol{x}^T \boldsymbol{w}_j}_{linear}),$$

  for some function $T(\cdot)$. Examples: Logistic, Poisson, $\cdots$

Generative Process: Linear SEMs

- Drawing one sample $\boldsymbol{x}$:
  For each node j in topological order of G,

$$\varepsilon_j \sim \textit{noise distribution}$$

$$x_j = T(\boldsymbol{x}_{pa(j)}, \varepsilon_j)$$

- Let $X \in \mathbb{R}^{n \times d}$ be collection of n such samples:

$$X = \begin{bmatrix} — & \boldsymbol{x}^{(1)} & — \\ & \vdots & \\ — & \boldsymbol{x}^{(n)} & — \end{bmatrix}$$

Generative Process: Linear SEMs

- For j-th dimension of i-th sample,

$$x_j^{(i)} = \boldsymbol{x}^{(i)^T} \boldsymbol{w}_j + \varepsilon_j^{(i)}$$

- Collecting all $i \in [n]$, $j \in [d]$,

$$\underset{(n\times d)}{X} = \underset{(n\times d)(d\times d)}{XW} + \underset{(n\times d)}{E}$$

- Natural loss function:

$$\ell(W; X) = \frac{1}{2n} \|X - XW\|_F^2$$

M-Estimation for DAGs

Given $X \in \mathbb{R}^{n \times d}$, solve

$$\begin{aligned} \min_{W \in \mathbb{R}^{d \times d}} \quad & \ell(W; X) \\ s.t. \quad & G(W) \in DAG \end{aligned}$$

- **Loss function:** log-likelihood of data, with respect to SEM model, given weighted adjacency matrix W
- **Constraint:** that W correspond to some DAG G
  - leads to difficult combinatorial optimization problem

Smooth Characterization of DAGs?

Smooth Characterization of DAG

Can we find a smooth function $h : \mathbb{R}^{d \times d} \to \mathbb{R}$ such that

$$h(W) = 0 \iff G(W) \in DAG$$

holds?

- will enable solvers based on continuous optimization, similar to Markov networks, in contrast to solving via constraint estimation, or local search in space of DAGs

Finite Power Series?

Consider binary adjacency matrix $B \in \{0,1\}^{d \times d}$

- Idea:

$$(B^k)_{ij} = \text{num of } k\text{-step paths from } i \text{ to } j$$

In other words,

$$\operatorname{tr}(B^k) = 0 \iff \text{no } k\text{-cycles}$$

- Candidate:

$$h(B) = \operatorname{tr}\left(\sum_{k=1}^{d} B^k\right) = 0 \iff G(B) \in DAG$$

Finite Power Series?

Consider binary adjacency matrix $B \in \{0,1\}^{d \times d}$

- Idea:

$$(B^k)_{ij} = \text{num of } k\text{-step paths from } i \text{ to } j$$

In other words,

$$\operatorname{tr}(B^k) = 0 \iff \text{no } k\text{-cycles}$$

- Candidate:

$$h(B) = \operatorname{tr}\left(\sum_{k=1}^{d} B^k\right) = 0 \iff G(B) \in DAG$$

Caveat: number of k cycles increases exponentially (ill conditioned)

Infinite Power Series?

- Idea: push k to infinity

$$\sum_{k=0}^{\infty} B^k = (I - B)^{-1}$$

- Candidate:

$$h(B) = \operatorname{tr}(I - B)^{-1} - d = 0 \iff G(B) \in DAG$$

Infinite Power Series?

- Idea: push k to infinity

$$\sum_{k=0}^{\infty} B^k = (I - B)^{-1}$$

- Candidate:

$$h(B) = \operatorname{tr}(I - B)^{-1} - d = 0 \iff G(B) \in DAG$$

Caveat: requires invertibility of I - B
(i.e. that spectral radius of B < 1)

Matrix Exponential

- Idea: is there a series that always converges? Yes!

$$e^B = I + B + \frac{1}{2!}B^2 + \frac{1}{3!}B^3 + \cdots$$

- Candidate:

$$h(B) = \operatorname{tr}(e^B) - d = 0 \iff G(B) \in DAG$$

Matrix Exponential

- Idea: is there a series that always converges? Yes!

$$e^B = I + B + \frac{1}{2!}B^2 + \frac{1}{3!}B^3 + \cdots$$

- Candidate:

$$h(B) = \mathrm{tr}(e^B) - d = 0 \iff G(B) \in DAG$$

Caveat: requires that adjacency matrix B be binary

Matrix Exponential for General Adjacency Matrices

- Idea: for nonnegative matrix $S \in \mathbb{R}_+^{d \times d}$,

$$(S^k)_{ij} = \text{sum of weight products long } k\text{-step paths from } i \text{ to } j$$

$$\operatorname{tr}(S^k) = 0 \iff \text{no } k\text{-cycles}$$

- Real to nonnegative:

$$S = W \circ W$$

- Candidate:

$$h(W) = \operatorname{tr}(e^{W \circ W}) - d = 0 \iff G(W) \in DAG$$

Smooth Characterization of DAGs

Smooth Characterization of DAG

Can we find a smooth function $h : \mathbb{R}^{d\times d} \to \mathbb{R}$ such that

$$h(W) = 0 \iff G(W) \in DAG$$

holds?

Answer: Yes!

$$h(W) = \operatorname{tr}(e^{W\circ W}) - d = 0 \iff G(W) \in DAG$$

Furthermore, it has a simple gradient

$$\nabla h(W) = (e^{W\circ W})^T \circ 2W$$

New M-estimator for DAGs

Given $X \in \mathbb{R}^{n \times d}$, solve

$$\min_{W \in \mathbb{R}^{d \times d}} \quad \ell(W; X)$$
$$s.t. \quad h(W) = 0$$

Optimization Algorithm: Augmented Lagrangian

- Solve an equivalent augmented form

$$\begin{aligned} \min_{W \in \mathbb{R}^{d \times d}} \quad & \ell(W; X) + \frac{\rho}{2} h^2(W) \\ s.t. \quad & h(W) = 0 \end{aligned}$$

- Lagrangian:

$$L(W, \alpha) = \ell(W; X) + \frac{\rho}{2} h^2(W) + \alpha h(W)$$

- Solve the dual:

$$\min_W \max_\alpha L(W, \alpha) = \underbrace{\max_\alpha \overbrace{\min_W L(W, \alpha)}^{\text{smooth, unconstrained}}}_{\text{1d linear maximization}}$$

NO TEARS

Algorithm 1 Augmented Lagrangian

- Input: $\ell, \nabla\ell, h, \nabla h$
- For $t = 1, 2, 3, \ldots$
  - Solve primal $W_{t+1} \leftarrow \text{argmin}_W L(W, \alpha_t)$.
  - Dual ascent $\alpha_{t+1} \leftarrow \alpha_t + \rho h(W_{t+1})$.

NOTEARS = Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian for Structure learning

NO TEARS

Algorithm 1 Augmented Lagrangian

- Input: $\ell, \nabla\ell, h, \nabla h$
- For $t = 1, 2, 3, \ldots$
  - Solve primal $W_{t+1} \leftarrow \text{argmin}_W L(W, \alpha_t)$.
  - Dual ascent $\alpha_{t+1} \leftarrow \alpha_t + \rho h(W_{t+1})$.

NOTEARS = Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian for Structure learning

(Corollary 11.2.1, Nemirovski, 1999) (loosely)
For ρ large enough, and with starting point α_0 near an optimum $\alpha^\star$, the updates converge to $\alpha^\star$ linearly.

NO TEARS

```python
def notears_simple(X, max_iter=100, h_tol=1e-8, w_threshold=0.3):
    n, d = X.shape
    w_est, w_new = np.zeros(d * d), np.zeros(d * d)
    rho, alpha, h, h_new = 1.0, 0.0, np.inf, np.inf
    bnds = [(0, 0) if i == j else (None, None) for i in range(d) for j in range(d)]
    for _ in range(max_iter):
        while rho < 1e+20:
            sol = sopt.minimize(_func, w_est, method='L-BFGS-B', jac=_grad, bounds=bnds)
            w_new = sol.x
            h_new = _h(w_new)
            if h_new > 0.25 * h:
                rho *= 10
            else:
                break
        w_est, h = w_new, h_new
        alpha += rho * h
        if h <= h_tol:
            break
    w_est[np.abs(w_est) < w_threshold] = 0
    return w_est.reshape([d, d])
```

30 lines (function, gradient) + 20 lines (optimize) $\approx$ 50 lines in total

In contrast to 1000s of lines of combinatorial optimization code
(with model-specific heuristics)

Code available at: https://github.com/xunzheng/notears

Also amenable to structured sparsity constraints on G

Prefer sparse graphs:

$$\begin{aligned} \min_{W \in \mathbb{R}^{d \times d}} \quad & \ell(W; X) + \lambda \|W\|_1 \\ s.t. \quad & h(W) = 0 \end{aligned}$$

No longer smooth!

Proximal Quasi-Newton for augmented Lagrangian:

$$\begin{aligned} & \min_W \; L(W, \alpha) \\ = & \min_W \; \underbrace{\ell(W; X) + \frac{\rho}{2} h^2(W) + \alpha h(W)}_{smooth} + \lambda \|W\|_1 \end{aligned}$$

Caveats with M-estimator

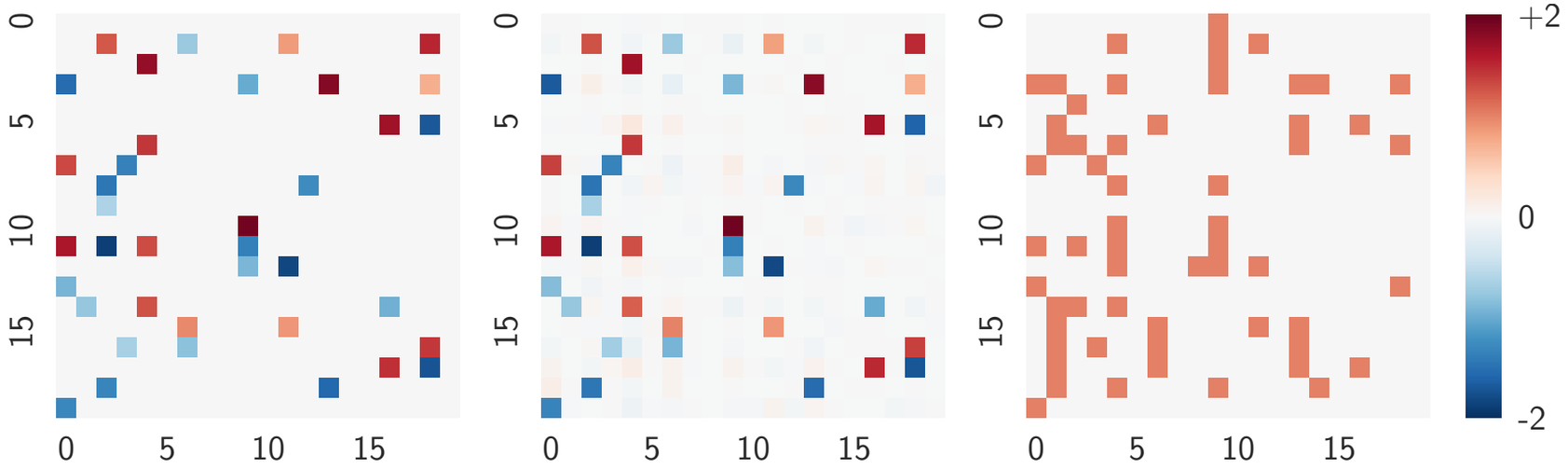
- Constraint set is non-convex
- Computing matrix exponential is $O(d^3)$
- With typical continuous optimization algorithms, constraints are only satisfied up to some tolerance
    - due to which we add a post-processing thresholding step to weighted adjacency matrix

Experiments

- Random graphs: Erdos-Renyi (ER), Scale Free (SF)
- Samples n = {20, 1000}, variables d = {10, 20, 100, 200}
- Baseline: FGS (Fast Greedy Search; Ramsey et al 2016); state of art implementation of greedy equivalent search (GES; Chickering 2008); has been known to outperform other local search techniques
- Metrics:
  - False Discovery Rate (FDR): #false edges/#predicted edges
  - Structural Hamming Distance (SHD): #false edges + #reversed edges + #missed edges

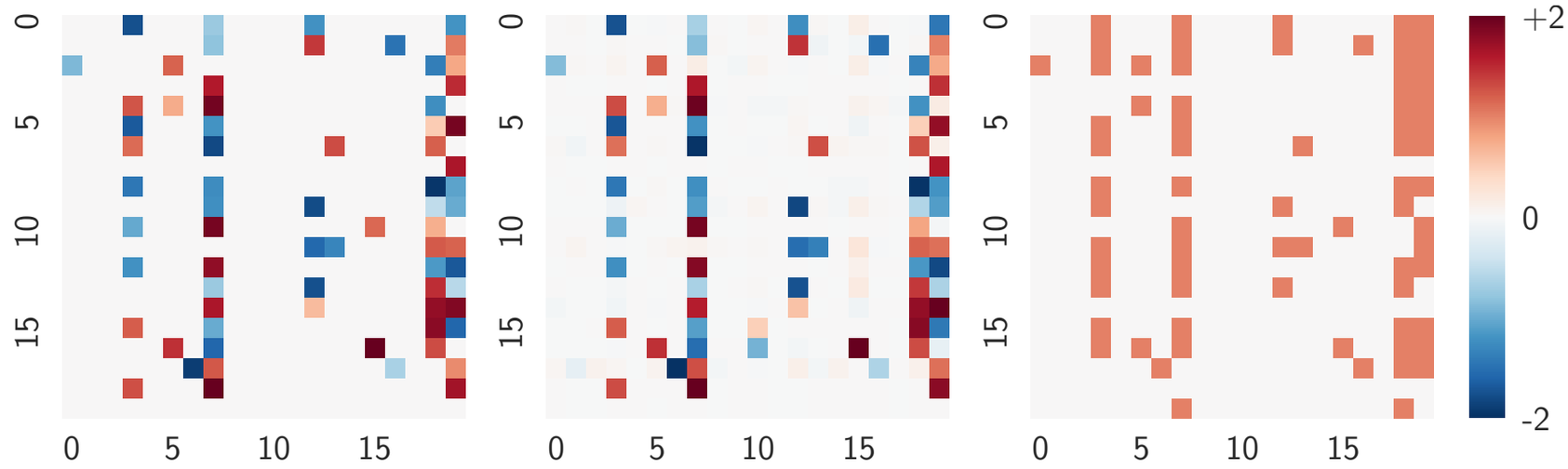
Heatmaps, n = 1000

ER Graphs:



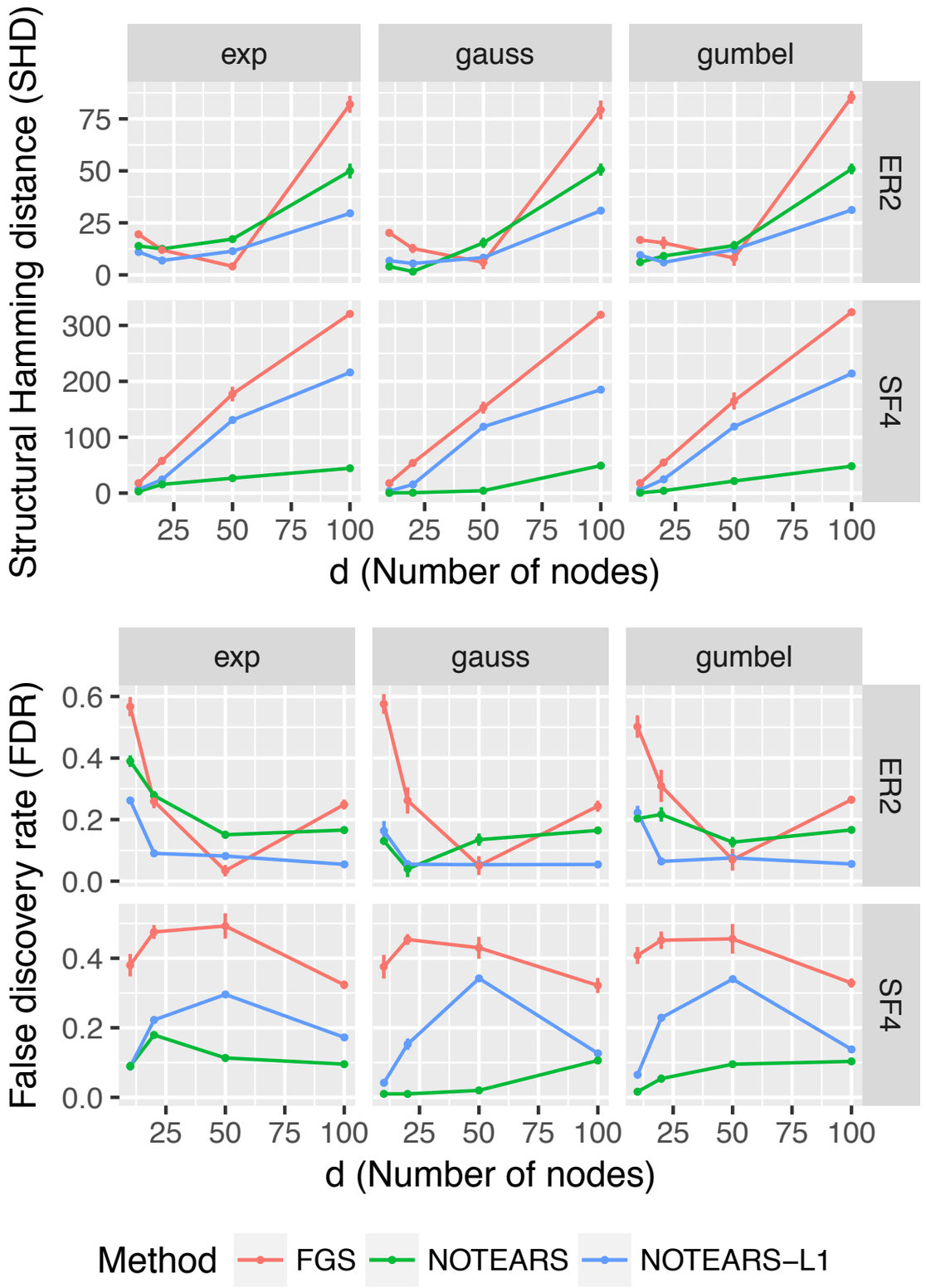
ground truth this work baseline (FGS)

SF Graphs:



ground truth this work baseline (FGS)

FDR, SHD, n = 1000



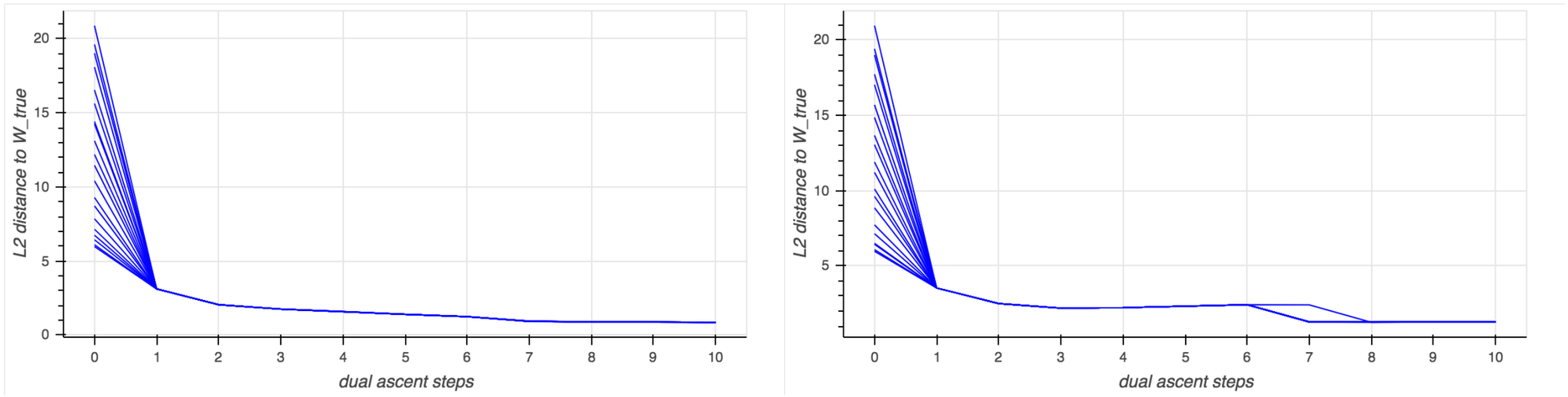
Comparison to global optimum

n	λ	Graph	$F(W)$	$F(W_{\mathsf{G}})$	$F(\widehat{W})$	$\Delta(W_{\mathsf{G}}, \widehat{W})$	$\|\widehat{W} - W_{\mathsf{G}}\|$	$\|W - W_{\mathsf{G}}\|$
20	0	ER2	5.11	3.85	5.36	-1.52	0.07	3.38
20	0.5	ER2	16.04	12.81	13.49	-0.68	0.12	3.15
1000	0	ER2	4.99	4.97	5.02	-0.05	0.02	0.40
1000	0.5	ER2	15.93	13.32	14.03	-0.71	0.12	2.95
20	0	SF4	4.99	3.77	4.70	-0.93	0.08	3.31
20	0.5	SF4	23.33	16.19	17.31	-1.12	0.15	5.08
1000	0	SF4	4.96	4.94	5.05	-0.11	0.04	0.29
1000	0.5	SF4	23.29	17.56	19.70	-2.13	0.13	4.34

- W: ground truth
- W_G: global optimum of NOTEARS M-estimator
- W_hat: Aug. Lagrangian (near) limit point of NOTEARS M-estimator

Sensitivity to Initialization

Initialization: $W_{\text{init}} \sim \text{Sphere}(r)$ uniformly.
What happens if we vary the radius $r \in [0, 20]$?



(Left) Population risk. (Right) Finite samples, $n = 100$.

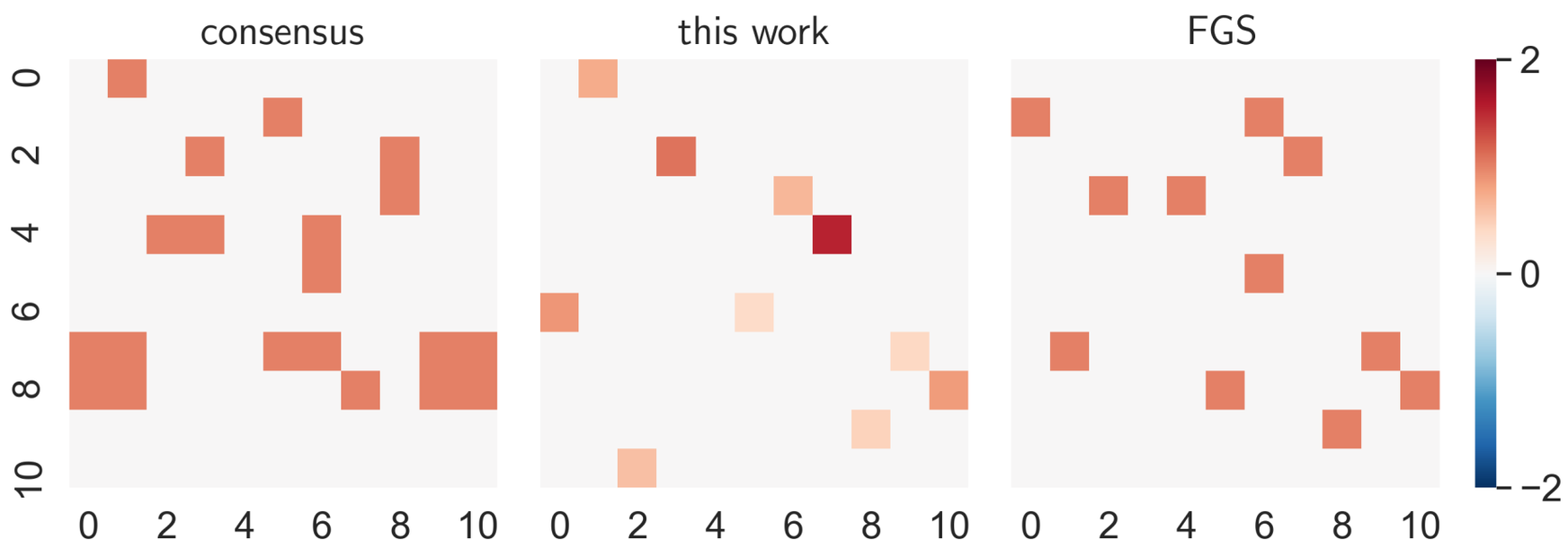
(Surprisingly) robust to initialization.

- Erdos-Renyi graph, Linear Gaussian SEM, d = 20, num edge = 20, n = {population, 200}, L1 regularization parameter = 0.1

- Each line plot is a random initialization with different r.

Real data example: Cytometry Data

Raw measurement data from Sachs et al. (2005).
Expression levels of proteins and phospholipids in human immune system cells (n = 7466 d = 11, 20 edges).



Summary

- We reduce learning DAGs to continuous optimization via a novel M-estimator
    - in contrast to conditional independence test based, and local search based methods
- Bridges gap between Markov and Bayesian networks with respect to scalable estimation
- Ongoing Work:
    - Analyze landscape of non-convex DAG regularization function: conditions under which it is feasible to get to global optimum
    - Approximate fast solvers for matrix exponential