Neural Causal Discovery and Graph Signal Processing

Shiuli Subhra Ghosh

Rensselaer Polytechnic Institute

March 20, 2022

Overview

1. Introduction (Graphs)
2. Signal Processing on Graphs
3. Notion of Causality in Graphical Models
4. Learning Causal Models From Data
5. Continuous Optimization for Causal Discovery
6. Discussion and Conclusion

Introduction

Network and Graphs

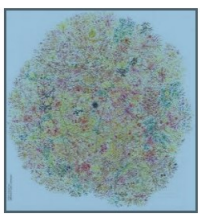
Why Graphs?

Graphs are a general language for describing and analyzing entities with relations/interactions.

Graphs/Networks can be of two types, Physical Networks and Logical Networks.

Online social media

Internet

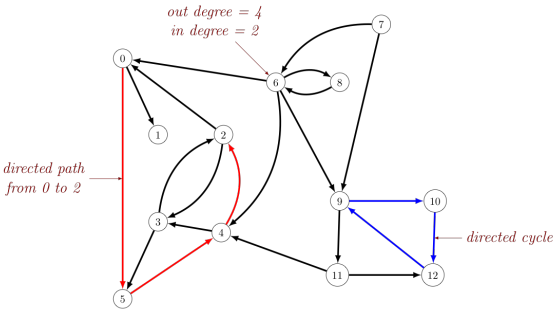


Clean energy and grid analytics

[1]

[1] A. G. Marques, Connecting the dots: learning graphs from nodal signals, Nov. 10, 2022

Some Definitions



Graph

- A graph $\mathcal{G} = (\mathbf{V}, \mathcal{E})$ - a collection of nodes ($\mathbf{V}$) and a set of edges ($\mathcal{E}$).
- Directed Graph - all the edges are directed.
- Directed Path - a finite or infinite sequence of directed edges.
- Directed Cycle - a directed path whose first and last vertices are the same.
- DAG - no directed cycles in a directed graph

Graph from Two Perspective

We discuss graphical models from two perspectives.

- Signal Processing
- Causality

Signal Processing on Graphs

Signal Processing on Graphs - Introduction

Graph Signal Processing - (GSP)

- Extended version of the Digital Signal Processing (DSP) approach for modelling graph signals.
- GSP captures the spacial relationship among the signals.

- The underlying graph can be represented as a matrix (e.g. Adjacency, Laplacian etc.).
- GSP considers frequency representation of the matrix.

Signal Processing on Graphs - Algebraic Representation

Adjacency Matrix

The *adjacency matrix* ($\mathbf{A}$) represents a graph, such that $(\mathbf{A})_{ij}$ is either 0 or 1 based on if there is an edge from node i to node j.

Degree Matrix

$$(\mathbf{D})_{ii} = \sum_{j=1}^{N} (A)_{ij} \quad \text{and} \quad (\mathbf{D})_{ij} = 0, \forall i \neq j \tag{1}$$

Laplacian

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{2}$$

Graph Shift Operator and Graph Filters

Graph Shift Operator (S)

A matrix S is called Graph Shift Operator if it satisfies, $(\mathbf{S})_{ij} = 0$ for $i \neq j$ and $(i,j) \notin \mathcal{E}$

$$\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} \tag{3}$$

For example -

1. Adjacency Matrix
2. Laplacian Matrix

Graph Filters

Matrix polynomials of the graph shift operator ($\mathbf{S}$).

$$\mathbf{H} := \sum_{\ell=0}^{N-1} h_\ell \mathbf{S}^\ell = \mathbf{V}\text{diag}(\tilde{\mathbf{h}})\mathbf{V}^{-1} \ , \tag{4}$$

Graph Frequencies and Graph Fourier Transform

Graph Frequencies

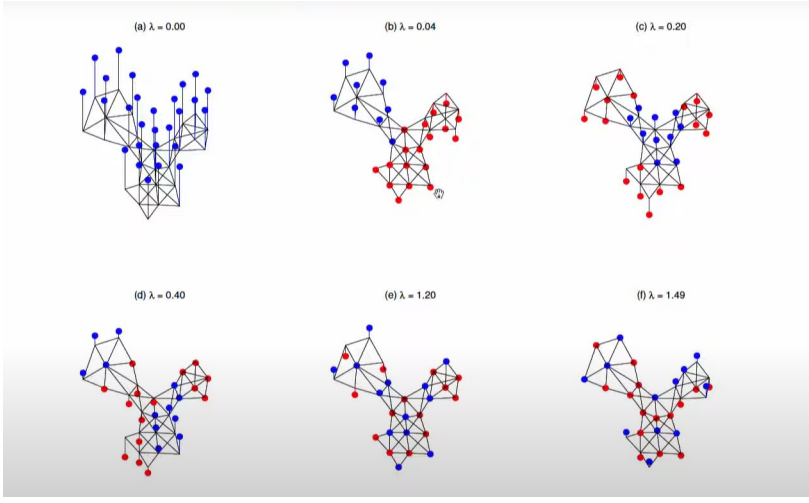
Graph frequencies are defined by the function of eigenvalues of the shift.

- Low frequency - a very low variation on the graph
- High frequency - a very high variation on the graph.

Graph Fourier Transform

If we do the eigendecomposition of the graph filters, the inverse of the orthogonal eigenvectors of the shift $\mathbf{S}$ is called the *Graph Fourier Transform (GFT)*.

$$\mathbf{F} = \mathbf{V}^{-1}$$



[2]

[2] Antonio Ortega, 2017-2018 ECE Distinguished Lecture Series.

Graph Signal Processing Applications - Clustering and Outlier Detection

Graph signal processing has multiple applications in engineering domains where leveraging the knowledge of network structures can extract useful insights.

1. GSP is built upon the graph spectra that can be perceived as a function of the structure of the graph.
2. The smooth graph signal model helps in **clustering** by using appropriate Fourier basis.
3. **detect outliers** by using high-pass filtering and thresholding in the graph.

Graph Signal Processing Applications - Network Propagation and Graph Sampling

1. GSP can provide insights into the **propagation** process of information and signals over networks.
2. **Graph sampling** is a very hard problem. If the graph is smooth, i.e., frequency is low enough, we can sample and reconstruct.

Graph Signal Processing Applications - Learning Graph

1. GSP can also be used for **learning graphs**.
2. With an assumption of smoothness and stationarity in the graph signals, the graph discovery problem can be formulated as an optimization problem.

Notion of Causality in Graphical Models

Causal Assumption

What is a Cause?

A variable X is said to be the cause of a variable Y if Y can change in response to changes in X.

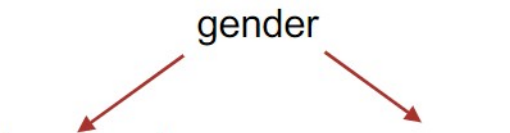
DAGs are used to capture the causal relations among the variables.

Causal Edge Assumption

In a directed graph, every parent is a direct cause of all its children.

What is special in Causal Graph?

1. The edges in a graph have a causal meaning.
2. X is a cause of Y, which does not imply the converse is true (asymmetric relation).



[3]

[3] Causality - Seminar for Statistics, ETH Zurich

Learning Causal Models from Data

Causal Graph Learning - Overview

What if we don't know the causal graph?

- We need to learn the model from data.

Well known methods for causal discovery

1. Independence-based causal discovery
2. Semi-Parametric causal discovery
3. Score-based causal discovery

Independence-Based Causal Discovery

- We can read the independencies from the empirical distribution.
- Assumptions - Global Markov Assumption and Faithfulness
- All the variables are observed (No hidden confounding).
- Example - PC algorithm

Limitations

- Identifiability upto Markov equivalence class. (skeleton and immoralities)
- Because fork and chain encode the same independencies.

Semi-Parametric Causal Discovery

- We can encode the causal interactions by some functional class.

Structural Equation Models

$$Y := f_y(X, N_Y), X \perp\!\!\!\perp N_Y \tag{5}$$

- But this functional class is too generic, and we don't have full structural identifiability.

- So, we use restricted functional class models. E.g. Additive noise models.

$$Y := f(X) + U, \quad U \perp\!\!\!\perp X. \tag{6}$$

- If the exogenous noise is non-gaussian and the functions are linear the model is identifiable.
- If the exogenous noise is gaussian and functions are non-linear then also the model is identifiable.

Score Based Causal Discovery

- Given i.i.d samples from observations, the idea is to design a score function.
- Find the best suitable graph which maximizes the score.

$$\hat{\mathcal{G}} = \arg\max_{\mathcal{G}} \mathcal{S}(\mathcal{D}, \mathcal{G}). \tag{7}$$

Problems in Causal Discovery from Observational Data

1. Problem of scalability
2. Space of all DAGs grows super exponentially with the number of nodes
3. Greedy approach is not the solution

Continuous Optimization for Causal Discovery

DAG - NOTEARS (2018)

- Score-based approach
- constructs smooth score function which enforces acyclicity [4]
- $\mathbf{X} \in \mathbb{R}^{n \times d}$ of n i.i.d. observations of the random vector $X = (X_1, \ldots, X_d)$.

$$X_j := w_j^T X + z_j. \tag{8}$$

- X_j can be modelled, by the expectation of the conditional distribution $\mathbb{E}(X_j \mid X_{\mathrm{Pa}(X_j)}) = f(w_j^T X)$ via logistic regression
- Score function - $\frac{1}{2n}||\mathbf{X} - \mathbf{X}W||_f^2 + \lambda||W||_1$, subject to, $h(W) = \mathrm{tr}(e^{W \odot W}) - d = 0$.
- Solve this optimization problem by Augmented Lagrangian Method.

[4] Zheng et.al., DAGs with NO TEARS: Continuous Optimization for Structure Learning, 2018

Gradient Based Neural DAG Learning (GraN-DAG)

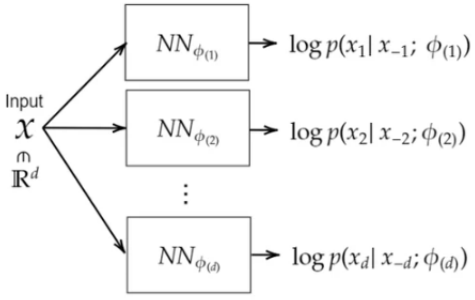
- The non-linear extension of NOTEARS was provided in GraN-DAG [5].
- Score-based approach.
- $\mathbf{X} \in \mathbb{R}^{n \times d}$ of n i.i.d. observations of the random vector $X = (X_1, \ldots, X_d)$.

$$X_j := f(\mathrm{Pa}(X_j) + N_j \quad , N_j \sim \mathcal{N}(0, \sigma^2) \quad , \forall j \tag{9}$$

- Leverages neural networks to model complex nonlinear relationships.
- Avoids the discrete nature of the graph search problem via an extension of a continuous constrained optimization formulation (same as NOTEARS).

[5] Lachapelle et.al., Gradient Based Neural DAG Learning, ICLR 2020

Learning Conditional Probability using Neural Network



$\phi_{(i)} \triangleq \{W_{(i)}^{(1)}, \ldots, W_{(i)}^{(L+1)}\}$

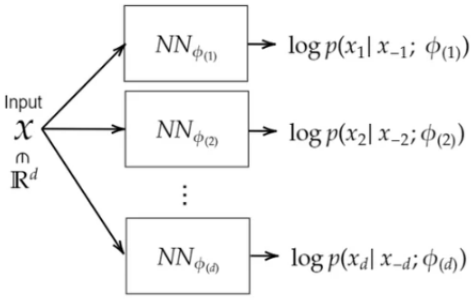
$W_{(i)}^{(\ell)} = \ell$th weight matrix of $NN_{\phi_{(i)}}$

$\phi \triangleq \{\phi_{(i)}\}_{i=1}^{d}$

6

[6] Lachapelle et.al., Gradient Based Neural DAG Learning, ICLR 2020

Learning Conditional Probability using Neural Network



$\phi_{(i)} \triangleq \{W_{(i)}^{(1)}, \ldots, W_{(i)}^{(L+1)}\}$

$W_{(i)}^{(\ell)} = \ell$th weight matrix of $NN_{\phi_{(i)}}$

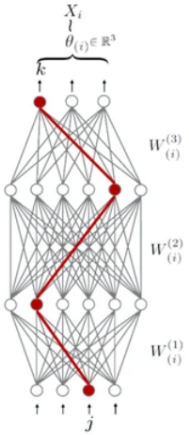
$\phi \triangleq \{\phi_{(i)}\}_{i=1}^{d}$

Need for imposing acyclicity constraint

$\prod_{i=1}^{d} P_j(x_j \mid \mathrm{Pa}(x_j), \phi_{(j)})$ does not decompose according to a DAG.

Construction of Weighted Adjacency Matrix (A_ϕ)

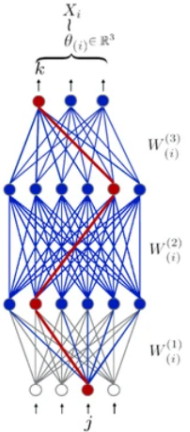
- Key idea - From neural network construction of one adjacency matrix could be used in the acyclicity constraints.



Description of A_ϕ

We want to get $(A_\phi)_{j,i}$ which measures the connection strength from $X_j \to X_i$

Construction of Weighted Adjacency Matrix (A_ϕ)



a

[a] Lachapelle et.al., Gradient Based Neural DAG Learning, ICLR 2020

Description of A_ϕ

We want to get $(A_\phi)_{j,i}$ which measures the connection strength from $X_j \to X_i$

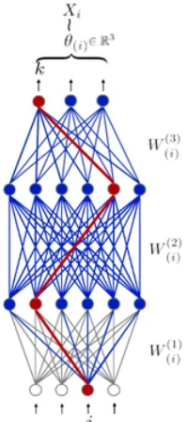
- Path product:

$$\left|W^{(1)}_{h_1 j}\right| \left|W^{(2)}_{h_2 h_1}\right| \left|W^{(3)}_{k h_2}\right| \geq 0$$

- $C \triangleq \left|W^{(3)}\right| \left|W^{(2)}\right| \left|W^{(1)}\right|$ The sum of all the path products from X_j to X_i :

$$\sum_{k=1}^{m} c_{kj} \geq 0$$

Construction of Weighted Adjacency Matrix (A_ϕ)



[a]

- $\sum_{k=1}^{m} C_{kj} = 0$ means all paths from X_j to X_i are inactive!

Final Formula

$$(A_\phi)_{ji} \triangleq \sum_{k=1}^{m} \left(C_{(i)}\right)_{kj}$$

[a] Lachapelle et.al., Gradient Based Neural DAG Learning, ICLR 2020

Explanation of Acyclity Constraints

Acyclity Constraint

$$h(\phi) = \text{tr}(e^{A_\phi}) - d = 0$$

Proof

- If B is the adjacency matrix of a graph, then the number of closed walks of length k can be obtained by, $\text{tr}(B^k)$.
- As we don't want any cycles, for our case, $\text{tr} \sum_{k=1}^{\infty} B^k = 0$.
- But just using the B can result in some numerical stability issues.
- Use of matrix exponentiation is proposed for the case of weighted adjacency matrix A_ϕ, given $(A_{phi})_{j,i} \geq 0$.

Explanation of Acyclity Constraints

Acyclity Constraint

$$h(\phi) = \text{tr}(e^{A_\phi}) - d = 0$$

Proof

$$\text{tr}(e^{A_\phi}) = \text{tr}(I + \frac{A_\phi^1}{1!} + \frac{A_\phi^2}{2!} + \cdots + \infty) \tag{10}$$

$$= d + \text{tr} \sum_{i=1}^{\infty} \frac{A_\phi^i}{i!} \tag{11}$$

$\sum_{i=1}^{\infty} \frac{A_\phi^i}{i!} = 0$, if A_ϕ is a DAG. Hence, $\text{tr}(e^{A_\phi}) - d = 0$.

Score Function

Maximum Likelihood Optimization Problem

$$\max_{\phi} \ \mathbb{E}_{X \sim P_X} \sum_{j=1}^{d} \log P_j(x_j \mid \mathrm{Pa}(x_j), \phi_{(j)}) \quad \text{, Subject to } \ \mathrm{tr}(e^{A_\phi}) - d = 0 \tag{12}$$

Note - $\sum_{j=1}^{d} \log P_j(x_j \mid \mathrm{Pa}(x_j)$ is a valid log-likelihood when the constraint $\mathrm{tr}(e^{A_\phi}) - d = 0$ is satisfied.

Optimization Procedure

- Augmented Lagrangian approach to get an approximate solution.
- Augmented Lagrangian transforms the constrained optimization problem to the unconstrained one.
- Solution to the sequence of unconstrained subproblems converge to a stationary point of the constrained optimization problem.

Lagrangian Subproblem

$$\max_{\phi} \mathcal{L}(\phi, \lambda_t, \mu_t) = \mathbb{E}_{X \sim P_X} \sum_{j=1}^{d} \log P_j(x_j \mid \mathrm{Pa}(x_j), \phi_{(j)}) - \lambda_t h(\phi) - \frac{\mu_t}{2} h(\phi)^2 \tag{13}$$

- μ_t = Penalty Coefficient and λ_t = Lagrangian Coefficient for t^{th} subproblem.

Optimization Procedure

Lagrangian Subproblem

$$\max_{\phi} \mathcal{L}(\phi, \lambda_t, \mu_t) = \mathbb{E}_{X \sim P_X} \sum_{j=1}^{d} \log P_j(x_j \mid \mathrm{Pa}(x_j), \phi_{(j)}) - \lambda_t h(\phi) - \frac{\mu_t}{2} h(\phi)^2 \tag{14}$$

- μ_t and λ_t are updated after solving each subproblem using the gradient ascent algorithm.
- B is a minibatch sampled from the data set. The gradient $\nabla_\phi \hat{\mathcal{L}}_B(\phi, \lambda_t, \mu_t)$ is computed.
- The convergence of the subproblem is tested on the validation data set ($\hat{\mathcal{L}}_H(\phi, \lambda_t, \mu_t)$).
- Early stopping criterion is enabled, when $\hat{\mathcal{L}}_H(\phi, \lambda_t, \mu_t)$ stops increasing.

λ_t and μ_t Update Rule

- Let ϕ^* be the approximate solution for the t^{th} subproblem. Then,

Update Rule

$$\lambda_{t+1} \leftarrow \lambda_t + \mu_t h\left(\phi_t^*\right)$$

$$\mu_{t+1} \leftarrow \begin{cases} \eta\mu_t, & \text{if } h\left(\phi_t^*\right) > \gamma h\left(\phi_{t-1}^*\right) \\ \mu_t, & \text{otherwise} \end{cases}$$

Hyperparameters: $\eta = 10$ and $\gamma = 0.9$ Convergence: $h(\phi) \leq 10^{-8}$

Computational Complexity

Method	Complexity
Matrix Exponential	$\mathcal{O}(d^3)$
Approximate Gradient of Log-likelihood	$\mathcal{O}(d^2)$

Table: Computational Complexity

Table 4: Total number of iterations ($\times 10^3$) before augmented Lagrangian converges on Gauss-ANM data.

	20 nodes ER1	20 nodes ER4	100 nodes ER1	100 nodes ER4
GraN-DAG	27.3 ± 3.6	30.4 ± 4.2	23.1 ± 0.7	23.1 ± 0.8
NOTEARS	67.1 ± 35.3	72.3 ± 24.3	243.6 ± 12.3	232.4 ± 12.9

7

[7] Lachapelle et.al., Gradient Based Neural DAG Learning, ICLR 2020

Results

- GraN-DAG performs better in the synthetic data than all the existing algorithms for causal discovery from observational data
  - Because GraN-DAG doesn't take any assumption regarding the functional class.
- For synthetic data, the performance measures are evaluated by Structural Hamming Distance (SHD) and Structural Interventional Distance (SID).
- All the causal discovery method performs badly for real data. Out of them, GraN-DAG performs better.
  - Real data might lack identifiability guarantees.

Discussion and Conclusion

Comparison of GSP and Causality - Graph Learning Aspect

- IN GSP literature, leveraging the property of smoothness and stationarity, the graph structure can be learned.
- But GSP is concerned with the spatial representation of the graph signals. The edges in the graphs don't have any causal meaning.
- Active research area - Causal graph learning using GSP techniques.

Future Scope of Research

- GraN-DAG can be extended for equilibrium causal models, by incorporating the notion of stability but not acyclicity.
- Incorporating prior knowledge for GraN-DAG for better estimates.
- Leveraging Spectral representation of graph signals for causal inference (building a strong connection between GSP and Causality).

Thank You

Appendix Slides

Probabilistic Graphical Models (Brief Introduction)

Definition

A graphical model or probabilistic graphical model (PGM) or structured probabilistic model is a probabilistic model for which a graph expresses the conditional dependence structure between random variables.

PGM's are modelled by graphical networks such as **Bayesian Network**.

Bayesian Network

(Bayesian Network) A Bayesian network (BN) is a DAG that models the dependencies among a set of random variables.

BN Chain Rule

Given a set of random variables $\mathbf{X} = (X_1, \ldots, X_n)$ with index set $\mathbf{V} = \{1, \ldots, n\}$, we denote their joint distribution by $P(X_1, X_2, \ldots, X_n)$. Using the BN chain rule of probability, we can factorize the joint distribution as,

$$P(X_1, X_2, \ldots, X_n) = P(X_1) \cdot \prod_i P(X_i | X_{i-1}, \ldots X_1). \tag{15}$$

Example - $P(A, B, C) = P(A)P(B \mid A)P(C \mid A, B)$

Local Markov Assumption

Now, if we have a DAG representation of a probability distribution, we can make the Local Markov Assumption to formalize the specification.

Local Markov Assumption

In a DAG, a node X is independent of all its non-descendants given its parents.

This assumption helps us to factorize the (15) as,

$$P(X_1, X_2, \ldots, X_n) = P(X_1) \cdot \prod_i P(X_i | \text{Pa}(X_i)). \tag{16}$$

Causal Assumption

What is a Cause?

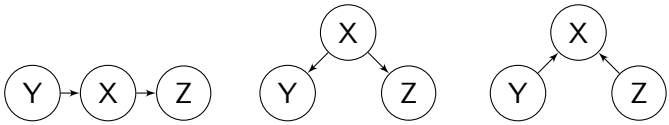
A variable X is said to be the cause of a variable Y if Y can change in response to changes in X.

For transforming a directed graph to a causal graph, we need the following assumption.

Causal Edge Assumption

In a directed graph, every parent is a direct cause of all its children.

Chain, Fork and Collider



(a) Chain (b) Fork (c) Collider

Figure: Basic graph building blocks

Graph	Independence
Chain	$Y \perp\!\!\!\perp Z \mid X$
Fork	$Y \perp\!\!\!\perp Z \mid X$
Collider	$Y \perp\!\!\!\perp Z$

d-Separation

A path between X to Y is blocked by a set S if at least one of the following holds:

1. There is a non-collider on the path that is in S
2. There is a collider on the path such that neither this collider nor any descendants are in S.

d-Separation

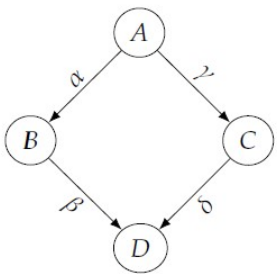
If all paths between X and Y are blocked sy S, then $X \perp Y \mid S$

Global Markov Property

Given that P is Markov with respect to G (satisfies the local Markov assumption), if X and Y are d-separated in G conditioned on Z, then X and Y are independent in P conditioned on Z.

$$X \perp\!\!\!\perp_G Y \mid Z \implies X \perp\!\!\!\perp_P Y \mid Z \tag{17}$$

Violation of Faithfulness



faithfulness

$$X \perp\!\!\!\perp_G Y \,|Z \Longleftarrow X \perp\!\!\!\perp_P Y|\, Z$$

From SCM, $A \perp\!\!\!\perp D$, but A and D aren't d-separated in the graph.

$$B := \alpha A$$
$$C := \gamma A$$
$$D := \beta B + \delta C$$
$$D = (\alpha\beta + \gamma\delta)A$$

Paths cancel if $\alpha\beta = -\gamma\delta$

Independence-Based Causal Discovery

- We can read the independencies from the empirical distribution.
- Just global Markov assumption is not sufficient.

Faithfulness Assumption

A graph G is said to be faithful according to a distribution P if,

$$X \perp\!\!\!\perp_P Y \mid Z \implies X \perp\!\!\!\perp_G Y \mid Z \tag{18}$$

- All the variables are observed (No hidden confounding).
- Example - PC algorithm

Limitations

- Identifiability upto Markov equivalence class. (skeleton and immoralities)
- Because fork and chain encode the same independencies.

Lagrangian Method for Optimization

Primal Problem

$$\min f(x)$$

Subject to

$$h(x) = 0$$

$$g(x) \leq 0$$

Lagrangian Formulation

The relaxed problem becomes,

$$\mathcal{L}(x, \lambda, \alpha) = f(x) + \lambda h(x) + \alpha g(x)$$

We want to minimize this lagrangian with respect to x. We want to solve this minimization problem using the dual function.

Lagrangian Method for Optimization

Dual function

$$q(\lambda, \alpha) = \min_x \mathcal{L}(x, \lambda, \alpha) \quad , \alpha >= 0$$

This dual problem is the unconstrained one.

Theorem (Weak Duality)

x^* is an optimal solution of the primal. Then,

$$q(\lambda, \alpha) \leq f(x^*)$$

Proof - $q(\lambda, \alpha) = \min_x \mathcal{L}(x, \lambda, \alpha) \leq \mathcal{L}(x^*, \lambda, \alpha) = f(x^*) + \lambda h(x^*) + \alpha g(x^*)$
If x^* is optimal solution $h(x^*) = 0$ and $g(x^*) \leq 0$. So, $q(\lambda, \alpha) \leq f(x^*)$

So, $q(\lambda, \alpha)$ is the lower bound of the primal solution. We need to maximize the lower bound. This means $\max_{\lambda,\alpha} q(\lambda, \alpha) = f(x^*)$.

Problem with Thresholding to ensure acyclicity

- It is possible to have $(A_{\phi_T})_{i,j}$ significantly higher than 0 while having $\theta_{(j)}$ almost complete independent of variable X_i.
  - NN paths from i to the output might be cancelling each other - the input i inactive.
  - Some neurons of the NNs might always be saturated for the observed range of inputs - NN paths are ineffective.

Sufficient but not necessary

Having $(A_{\phi_T})_{i,j}$ is only a sufficient condition to have $\theta_{(j)}$ independent of variable X_j and not a necessary one.

Expected Jacobian

Solution

To avoid this problem, expected Jacobian matrix is computed instead of A_{ϕ_T}

$$\mathcal{J} \triangleq \mathbb{E}_{X \sim P_X} \left| \frac{\partial \mathcal{L}}{\partial X} \right|^\top$$

- where $\left| \frac{\partial \mathcal{L}}{\partial X} \right|$ is the Jacobian matrix of $\mathcal{L}$ evaluated at X, in absolute value (element-wise).
- The entry $\mathcal{J}_{ij}$ can be loosely interpreted as the strength of edge (i, j).
- Removing edges starting from the lowest $\mathcal{J}_{ij}$ to the highest, stopping as soon as acyclicity is achieved.

Reducing Overfitting

- Maximum Likelihood Score is prone to overfitting.
- Adding edges can never reduce the maximal likelihood.

Four possible solutions

- Pruning edges by thresholding. $(A_\phi)_{i,j} \leq 10^{-4} = 0$
- Imposing early stopping.
- Final pruning by regressing each node against its parents and using a significance test to find the most suitable set of parents.
- Using preliminary neighbourhood selection for graphs with more than 50 nodes.

Regularization

- Shrinking the coefficient estimated can significantly reduce their variance.
- Two types of regularize
  1. Ridge (L2)
  2. Lasso (L1)

Ridge Regularizer

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 \tag{19}$$

- Ridge Regression seeks coefficient estimates that fit the data well, by making the residual error small.
- The term, $\lambda\sum_{j=1}^{p}\beta_j^2$ - shrinkage penalty, is small when β_j's are close to 0. It has the effects of shrinking β_j towards zero.

Verification of Causal Graph

Structural Hamming Distance

The structural hamming distance (SHD) between two partially directed acyclic graphs count how many edge types do not coincide. Estimating a non-edge or a directed edge instead of an undirected edge, contributes an error of one to the overall distance.

Limitation of SHD

The Structural Hamming Distance counts the number of incorrect edges. Although this provides an intuitive distance between graphs, it does not reflect their capacity for causal inference.

Main Idea behind SID

The idea of distance should follow the causal interpretation of a graph that enables to predict the result of interventions.

Computing Interventional Distribution: Computed using adjustment of parents. The graphs can be different in two sense,

1. Adding an edge
2. Reversing an edge

So, (2) is more severe than (1), where SHD has same importance for both 1 and 2.

Why Non-Gaussianity is important for Linear Setting?

In the linear non-Gaussian setting, if the true SCM is

$$Y := f(X) + U, \quad X \perp\!\!\!\perp U,$$

then there does not exist an SCM in the reverse direction,

$$X := g(Y) + \tilde{U}, \quad Y \perp\!\!\!\perp \tilde{U},$$

that can generate data consistent with $P(x, y)$.

Non Gaussianity and Independence

Let the underlying model is,

$$x_1 = e_1 \qquad x_2 = b_{21}x_1 + e_2 \qquad b_{21} \neq 0$$

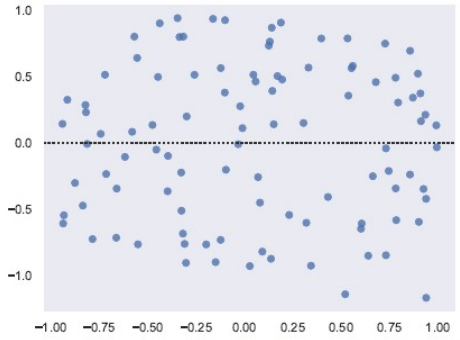
e_1 and e_2 are non-gaussian. Regressing the effect x_2 on cause x_1, we get the residual,

$$r_2^{(1)} = x_2 - \frac{cov(x_2, x_1)}{var(x_1)}x_1 = x_2 - b_{21}x_1 = e_2$$

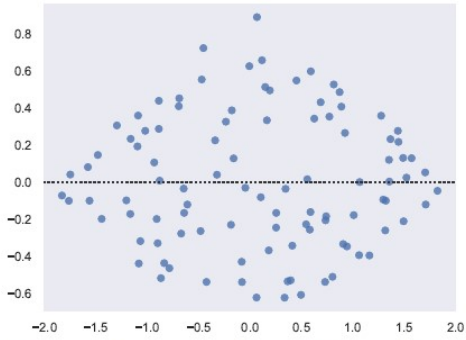
Hence x_1 and $r_2^{(1)}$ are independent.
But Regressing the cause x_1 on the effect x_2, we get the residual,

$$r_1^{(2)} = x_1 - \frac{cov(x_1, x_2)}{var(x_2)}x_2 = \left(1 - \frac{b_{21}cov(x_1, x_2)}{var(x_2)}\right)e_1 + \frac{b_{21}var(x_1)}{var(x_2)}e_2$$

Hence, x_2 and $r_1^{(2)}$ are dependent, although they are uncorrelated. So, when regressing through the opposite causal direction, the explanatory variable and the residual becomes dependent.



(a) Causal direction residuals: residuals that result from linearly regressing Y on X.



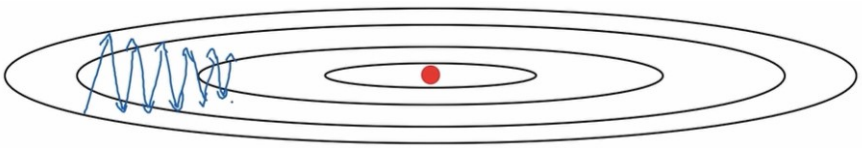
(b) Anti-causal direction residuals: residuals that result from linearly regressing X on Y.

List of Hyperparameters

Hyperparameter	Value
n_samples (Real Data)	853
n_sample (synthetic Data)	1000
Hidden Layers	2
no. nurons (per hidden layer)	10
Initialization	Xavier (Bias $= 0$, Weight $\sim \mathcal{N}(0, \frac{1}{n^l})$)
Optimizer	RMS Prop
Learning rate	1st - 10^{-2}, rest 10^{-4}
Mini batch size	64
Train test split	80%, 20%
Train iterations	100000
Activation	Leaky ReLU $(\max(0, x) + \lambda \min(0, x), \lambda < 0)$
μ_0	10^{-3}
λ_0	0

Table: Hyperparameter List

RMS Prop Optimization



On iteration t: Compute dW,db on the current minibatch.

$$S_{d\omega} = \beta S_{d\omega} + (1-\beta) d\omega^2$$
$$S_{db} = \beta S_{db} + (1-\beta) db^2$$
$$\omega := \omega - \alpha \frac{d\omega}{\sqrt{S_{d\omega}}} \quad b := b - \alpha \frac{db}{\sqrt{S_{db}}}$$

Statistical Methods for Network Topology Inference

- We can find Pearson's correlation coefficient for each pair.

$$\text{sim}(i,j) := \rho_{ij} = \frac{\text{cov}\left[x_i, x_j\right]}{\sqrt{\text{var}\left[x_i\right]\text{var}\left[x_j\right]}}, \quad i,j \in \mathcal{V} \tag{20}$$

- Zero correlation - no edges.
- Translates to hypothesis testing problem.

$$H_0 : \rho_{ij} = 0 \text{ vs. } H_1 : \rho_{ij} \neq 0 \tag{21}$$

- Find sample covariance $\hat{\mathbf{C}} = \mathbf{X}\mathbf{X}^T$, then $\hat{\rho}_{ij} = \hat{C}_{ij}/\sqrt{\hat{C}_{ij}\hat{C}_{jj}} \Rightarrow$ Edge exists if: $0.5 \log\left(\frac{1+\hat{\rho}_{ij}}{1-\hat{\rho}_{ij}}\right) > \frac{z_{\alpha/2}}{\sqrt{P-3}}$, with $P_{FA} = \alpha$
- Aim - Sparcification of the covariance matrix

Partial Correlations

1. But correlation doesn't imply causation.
2. There can be a hidden confounder which influences two variables.
3. Partial correlation better captures the direct influence among the vertices. For $i, j \in \mathcal{V}$ consider latent vertices $\mathcal{V}_{-ij} = \mathcal{V} \backslash \{i, j\}$, then partial correlation of x_i and x_j, adjusting for $\mathbf{x}_{-ij} = [x_1, .., x_{i-1}, x_{i+1}, \ldots, x_{j-1}, x_{j+1}, \ldots, x_N]^T$ is

$$\rho_{ij|\mathcal{V}_{-ij}} = \frac{\operatorname{cov}[x_i, x_j \mid \mathbf{x}_{-ij}]}{\sqrt{\operatorname{var}[x_i \mid \mathbf{x}_{-ij}] \operatorname{var}[x_j \mid \mathbf{x}_{-ij}]}}, \quad i, j \in \mathcal{V}$$

4. The precision matrix $\mathbf{\Theta} = \mathbf{C}^{-1}$
5. Under gaussianity, $\rho_{ij|\mathcal{V}_{-ij}} = 0$ iff x_i and x_j are conditionally independent.

Graph Lasso

This precision matrix can be estimated by graph lasso method.

$$\hat{\mathbf{\Theta}} = \arg\max_{\mathbf{\Theta} \geq 0} \left\{ \log\det \mathbf{\Theta} - \text{trace}(\hat{\mathbf{C}}\mathbf{\Theta}) - \lambda \|\mathbf{\Theta}\|_1 \right\} \tag{22}$$

Learning Graphs from Smooth Signals

Problem Statement

Given observations $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_P] \in \mathbb{R}^{N \times P}$, identify a graph G such that signals in $\mathbf{X}$ are smooth on G.

Criterion: Dirichlet energy on the graph G with Laplacian $\mathrm{L} \Rightarrow$ Search for the GSO $\mathbf{S} = \mathbf{L}$ such that $TV(\mathbf{x}) = \mathbf{x}^T \mathbf{L}\mathbf{x}$ small

$$\mathrm{TV}(\mathbf{X}) = \sum_{p=1}^{P} \mathbf{x}_p^T \, \mathrm{L} x_p$$

- Find a laplacian which makes the observation smooth.
- The objective function is

$$\mathbf{L}^* = \arg\min_{\mathbf{L}} \left\{ \sum_{p=1}^{P} \mathbf{x}_p^T \mathbf{L}\mathbf{x}_p + \frac{\beta}{2} \|\mathbf{L}\|_F^2 \right\}$$

subject to $\mathrm{tr}(\mathbf{L}) = N, \mathbf{L}\mathbf{1} = \mathbf{0}, L_{ij} = L_{ji} \leq 0, i \neq j$

Learning Graphs from Stationary Graph Signals

Stationarity

Def: A graph signal $\mathbf{x}$ is stationary with respect to the shift $\mathbf{S}$ if and only if $\mathbf{x} = \mathbf{H}\mathbf{w}$, where $\mathbf{H} = \sum_{l=0}^{L-1} h_l \mathbf{S}^l$ and $\mathbf{w}$ is white.

- The covariance matrix $\mathbf{C} = \mathbb{E}\left[\mathbf{x}\mathbf{x}^T\right]$ is a polynomial on $\mathbf{S}$.
- The idea is to find the polynomial mapping from the covariance matrix to the shift matrix such that the optimized shift is sparse.
- Eigenvectors of the shift = Eigenvectors of the covariance.
- Shift and the covariance matrix commutes

$$\mathbf{S}^* = \underset{\mathbf{S}}{\operatorname{argmin}} \|\mathbf{S}\|_0 \quad \text{s. to } \hat{\mathbf{C}}\mathbf{S} = \mathbf{S}\hat{\mathbf{C}}, \mathbf{S} \in \mathcal{S} \tag{23}$$

Important Points on GSP

Shift Invariance Filters

$$\mathbf{AH} = \mathbf{AH}.$$

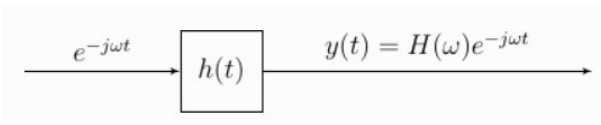
If the characteristic polynomial $p_A(z)$ and the minimum polynomial $m_A(z)$ of $\mathbf{A}$ are equal, then every filter commuting with $\mathbf{A}$ is a polynomial in $\mathbf{A}$, i.e.,

$$\mathbf{H} = h(\mathbf{A}).$$

Eigenvectors of A are the eigenfunctions of the filter

$$\begin{aligned} \mathbf{H}\boldsymbol{v}_m &= \mathbf{V}h(\mathbf{\Lambda})\mathbf{V}^{-1}\boldsymbol{v}_m \\ &= \mathbf{V}h(\mathbf{\Lambda})\boldsymbol{e}_m \\ &= h\left(\lambda_m\right)\boldsymbol{v}_m \end{aligned} \tag{24}$$

Eigenfunctions



Consider the system with impulse response h then the output is given by:

$$y(t) = \int_{-\infty}^{\infty} e^{j\omega(t-u)} h(u) du$$

We can simplify this as

$$y(t) = e^{j\omega t} \int_{-\infty}^{\infty} e^{-j\omega u} h(u) du$$

Observe that the integral only depends on ω and we denote it as $H(\omega)$, then:

$$y(t) = e^{j\omega t} H(\omega)$$