Problem Set 2

6.S091: Topics in Causality

IAP 2024

Due: Thursday, January 25th at 3pm EST

- Problem sets **must** be done in LaTeX.
- Printed problem sets must be turned in at the beginning of lecture. If you cannot attend, please find a classmate to turn the problem set in for you.
- You may use any programming language for your solutions (we recommend `python`). You are **required** to attach your code at the end of this PDF, which constitutes 50% of the points for each corresponding subproblem.

Problem 1: Implementing DAGs with NO TEARS [30 points]

In the seminar-style lecture, we have read about a continuous method for causal structure learning using NOTEARS. NOTEARS can be a useful tool in various research in causality and machine learning. This problem aims to consolidate your understanding of this method by implementing NOTEARS in linear causal models with additive noises.

Suppose we are given a data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ of n i.i.d. observations of the random vector $X = (X_1, ..., X_d)$, where X follows a linear structural causal models with additive noises:

$$X_j = w_j^\top X + z_j, \qquad j = 1, ...d.$$

Here $w_j \in \mathbb{R}^{d \times 1}$ are unknown and $z = (z_1, ..., z_d)$ is a random noise vector.

Note that w_j represents the incoming edge weights to node j where $w_{ji} = 0$ if and only if there is no edge from i to j. Let $W = (w_1, ..., w_d) \in \mathbb{R}^{d \times d}$. The adjacency matrix $\mathcal{A}(W) \in \{0, 1\}^{d \times d}$, defined by $\mathcal{A}(W)_{ji} = 1 \iff W_{ji} \neq 0$, represents the underlying causal graph that generates the data matrix. Denote the underlying causal graph by $\mathsf{G}(W)$. Figure 1 shows an example when $d = 4$.

(a) Edge weights W

(b) Causal graph $\mathsf{G}(W)$

Figure 1: An example of W and $\mathsf{G}(W)$.

Preliminaries [10 point]

(a) For any positive integer k, show that $\mathrm{tr}(\mathcal{A}(W)^k)$ equals to the number of length-k circles in $\mathsf{G}(W)$.

(b) NOTEARS makes use of the following trace-exponential penalty:

$$h(W) = \mathrm{tr}(e^{W \circ W}) - d,$$

where $\circ$ is the Hadamard product and e^* is the matrix exponential. Prove that $h(W) \geq 0$ for any $W \in \mathbb{R}^{d \times d}$, and $h(W) = 0$ if and only if $\mathsf{G}(W)$ is acyclic. Write out the closed-form formula for the gradient $\nabla h(W)$.

Implementing the constraint [10 point]

(c) In your preferred programming language, write a function that takes in W and outputs $h(W)$.

(d) Evaluate your implemented function on the following W_1, W_2 and provide the outputs $h(W_1)$, $h(W_2)$.

$$W_1 = \begin{pmatrix} 0 & -2 & 0.5 & 0 \\ 0 & 0 & 1.5 & 0.8 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$W_2 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1.5 \\ 0.5 & 0 & 0 \end{pmatrix}.$$

Optimizing with augmented Lagrangian [10 point]

Assume that the the data matrix is generated by a directed acyclic graph (DAG), i.e., $\mathsf{G}(W)$ has no cycles. To learn the underlying weights W and $\mathsf{G}(W)$, NOTEARS proposes to solve the following constrained optimization problem:

$$\begin{aligned} \min_{W \in \mathbb{R}^{d \times d}} \quad & F(W), \\ \text{subject to} \quad & h(W) = 0. \end{aligned} \tag{1}$$

Here $F(W) = \frac{1}{2n}\|\mathbf{X} - \mathbf{X}W\|_F^2$,[1] which corresponds to the least-square loss. Note that $\|\cdot\|_F$ denotes the Frobenius norm.

In the preliminaries, we have proven how such constraint can be used to enforce acyclicity of $\mathsf{G}(W)$. We have also implemented how to compute this constraint. Now we combine the pieces together.

(e) Implement a solver for Eq. (1). You will be scored given your performance in **(f)**.

Hints: We will layout how to use an augmented Lagrangian method to solve Eq. (1). You can choose to either implement this procedure or your own choice of optimization method (you are encouraged to do this. please write out the detailed steps if you do).

Augmented Lagrangian are a class of methods that can be used to solve constrained optimization problems like Eq. (1). We will skip the introduction by directly providing a procedure for solving Eq. (1). For detailed deductions and explanations, one may refer to online resources (e.g., lecture notes or books).

Algorithm 1 Pseudocode for solving Eq. (1) with Augmented Lagrangian

1: **Input**: Data matrix $\mathbf{X}$.
2: **Output**: Weights W.
3: Set hyper-parameter $\beta = 10$, tolerance $\epsilon = 10^{-6}$.
4: Initialize $W_0 = 0^{d \times d}$, $\rho_0 = 1$, $\alpha_0 = 0$, $k = 0$.
5: **while** k is less than the maximum iterations **do**
6: $\quad W_{k+1} = \arg\min_{W \in \mathbb{R}^{d \times d}} F(W) + \frac{\rho_k}{2} h(W)^2 + \alpha_k h(W)$. ▷ solve the unconstrained subproblem
7: $\quad \alpha_{k+1} = \alpha_k + \rho_k \cdot h(W_{k+1})$ ▷ update the estimate for Lagrangian multiplier
8: $\quad \rho_{k+1} = \beta \cdot \rho_k$ if $h(W_{k+1}) \geq h(W_k)/2$ else $\rho_{k+1} = \beta \cdot \rho_k$. ▷ increase ρ according to a scheme
9: $\quad k \leftarrow k + 1$.
10: $\quad$ **if** ρ_k exceeds limit or $h(W_k) < \epsilon$ **do** ▷ stopping criteria
11: $\quad\quad$ **break**
12: **return** W_k

Note: you can call any off-the-shelf optimizer in existing packages to implement line 6. You can change line 3 and line 8 for alternative updating schemes for ρ. The output W is likely going to be a dense matrix. You will need to threshold it to obtain a sparse graph $\mathsf{G}(W)$ (see below).

(f) Run your implementation on the provided data matrix. For some threshold $\omega > 0$, let $\mathcal{A}(W, \omega) \in \{0, 1\}^{d \times d}$ be the adjacency matrix defined by $\mathcal{A}(W, \omega)_{ji} = 1 \iff |W_{ji}| \geq \omega$. How does your adjacency matrix look like for $\omega = 0.0$, 0.1, 0.3 and 1.0?

[1] Note that we simply here by removing the sparsity constraint.