Tucker-Dynotears: Non-linear Causal Discovery via Tensor Decomposition for Anomaly Detection

Nicolas Bigeard

November, 2025

Abstract

Causal structure learning in high-dimensional time series is a critical component of root cause analysis for industrial systems. Existing methods typically rely on linear Structural Vector Autoregression (SVAR), which fails to capture non-linear interactions such as saturation or hysteresis. While non-linear extensions exist, they are often computationally intractable due to the exponential growth of parameters in interaction terms. This paper introduces a unified framework, **Tucker-Dynotears**, which integrates the Dynotears continuous optimization strategy with a Tensor-Decomposed Causal Additive Model (Tucker-CAM). We formulate the structure learning problem as a constrained optimization where the acyclicity constraint is applied to a weight matrix derived from Tucker factors, while the loss function minimizes the reconstruction error of a penalized B-spline model. We further detail the application of this learned structure to window-based anomaly detection using spectral graph metrics.

1 Introduction

Multivariate time series data from cyber-physical systems often exhibit complex, non-linear dependencies. Traditional anomaly detection methods that rely on correlation matrices or marginal distributions are insufficient for identifying the source of a fault (Root Cause Analysis), as they do not model the directional propagation of errors. To address this, causal discovery algorithms attempt to recover the underlying Directed Acyclic Graph (DAG) governing the system.

The *Dynotears* framework [1, 2] represented a significant advance by formulating DAG learning as a continuous optimization problem using a trace exponential constraint. However, the standard implementation assumes a linear relationship between variables. Extending this to non-linear cases usually involves kernel methods or neural networks, which lack interpretability and parameter efficiency.

We propose a method that links the Dynotears optimization framework with a Tucker-decomposed P-spline model (Tucker-CAM). By representing non-linear interaction surfaces as low-rank tensors, we maintain the interpretability of additive models while reducing memory complexity from exponential to linear with respect to the number of parents. This paper details the mathematical derivation of inserting the Tucker-CAM loss function into the Dynotears constraints.

2 Methodology

Let $\mathbf{X} \in \mathbb{R}^{T \times d}$ be a time series dataset consisting of T observations and d variables. We assume the data is stationary (verified via ADF and KPSS tests).

2.1 The Generalized Dynotears Framework

The core premise of Dynotears is to find a weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ and a lag matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ (for lag $p = 1$) that minimize a prediction loss $L(\theta)$ subject to an acyclicity constraint $h(\mathbf{W}) = 0$.

The general optimization problem is:

$$\min_\theta L(\mathbf{X};\theta) + \lambda_W \|\mathbf{W}(\theta)\|_1 + \lambda_A \|\mathbf{A}(\theta)\|_1 \tag{1}$$

$$\text{subject to } h(\mathbf{W}(\theta)) = \mathrm{Tr}(e^{\mathbf{W}(\theta)\circ\mathbf{W}(\theta)}) - d = 0 \tag{2}$$

In the standard linear formulation, the parameters θ are simply the matrices $\mathbf{W}$ and $\mathbf{A}$ themselves, and the loss L is the Frobenius norm of the residuals. In our proposed **Tucker-Dynotears**, θ represents the factors of a tensor decomposition, and $\mathbf{W}$ is a derived variable representing the aggregate interaction strength.

2.2 Tucker-CAM: Tensor-Based Non-Linear Modeling

We assume that each variable $X_{t,j}$ is a non-linear function of all other variables at time t and $t-1$:

$$X_{t,j} = f_j(\mathbf{X}_t, \mathbf{X}_{t-1}) + \epsilon_{t,j} \tag{3}$$

To model f_j non-linearly while capturing interactions, we use B-splines. A full tensor-product B-spline basis for d variables results in K^d coefficients, which is intractable. We approximate the coefficient tensor using Tucker Decomposition.

For a specific target variable j (omitted for brevity), the predicted value $\hat{x}_t$ is given by:

$$\hat{x}_t = \mathcal{G} \times_1 (\mathbf{B}(\mathbf{X}_t)\mathbf{U}^{(1)}) \times_2 \cdots \times_K (\mathbf{B}(\mathbf{X}_{t-1})\mathbf{U}^{(K)}) \tag{4}$$

where:

- $\mathcal{G}$ is the core tensor of small rank $(r, \ldots, r)$.
- $\mathbf{U}^{(k)}$ are factor matrices mapping the high-dimensional B-spline basis space to the low-rank core space.
- $\mathbf{B}(\cdot)$ represents the B-spline basis expansion of the input data.

2.3 Linking Tucker-CAM to the Dynotears Formula

To insert this model into the Dynotears framework, we must define two components: the Loss Function L and the Effective Adjacency Matrix $\mathbf{W}$.

2.3.1 1. The Non-Linear Loss Function

The loss function replaces the standard Least Squares term. It is the reconstruction error of the Tucker model plus a smoothness penalty (P-splines):

$$L(\mathcal{G}, \mathbf{U}) = \frac{1}{2T}\sum_{t=1}^{T} \|\mathbf{X}_t - \hat{\mathbf{X}}_{Tucker}(\mathbf{X}_t, \mathbf{X}_{t-1})\|^2 + \lambda_{smooth}\sum_k \mathrm{Tr}(\mathbf{U}^{(k)T}\mathbf{D}^T\mathbf{D}\mathbf{U}^{(k)}) \tag{5}$$

where $\mathbf{D}$ is the second-order difference matrix used in P-splines to penalize high-frequency oscillations (roughness).

2.3.2 2. The Effective Adjacency Matrix

The standard Dynotears constraint $h(\mathbf{W})$ requires a $d \times d$ matrix $\mathbf{W}$ where W_{ij} represents the dependence of variable i on variable j. In the Tucker model, this dependence is distributed across the core tensor $\mathcal{G}$ and the factor matrices $\mathbf{U}$.

We define the effective weight W_{ij} as the Frobenius norm of the partial derivative of the function f_i with respect to input j, averaged over the domain. Alternatively, in the B-spline domain, we aggregate the coefficients. A simplified effective weight for the optimization constraint is defined by the norm of the slices of the reconstructed coefficient tensor.

Let $\mathcal{C}_i$ be the reconstructed coefficient tensor for target variable i. We define W_{ji} (influence of j on i) as:

$$W_{ji} = \|\mathcal{C}_i \times_j \mathbf{1}\|_F \tag{6}$$

In practice, to maintain differentiability for backpropagation, we compute $\mathbf{W}$ directly from the factor matrices $\mathbf{U}$. Since the factor matrix $\mathbf{U}^{(j)}$ scales the contribution of input variable j, we define:

$$W_{ji} \approx \|\mathbf{U}^{(j)}_{target=i}\|_F \tag{7}$$

2.3.3 3. The Unified Optimization Problem

We now substitute these definitions back into the Augmented Lagrangian formulation. We minimize:

$$\mathcal{L}_{total} = L(\mathcal{G}, \mathbf{U}) + \alpha h(\mathbf{W}(\mathbf{U})) + \frac{\rho}{2}|h(\mathbf{W}(\mathbf{U}))|^2 + \lambda_{sparsity}\|\mathbf{W}(\mathbf{U})\|_1 \tag{8}$$

Gradient descent is performed on the parameters $\mathcal{G}$ and $\mathbf{U}$. Crucially, via the chain rule, the gradient of the acyclicity constraint $\nabla h(\mathbf{W})$ propagates back to the factor matrices $\mathbf{U}$:

$$\frac{\partial h}{\partial \mathbf{U}} = \frac{\partial h}{\partial \mathbf{W}} \cdot \frac{\partial \mathbf{W}}{\partial \mathbf{U}} \tag{9}$$

This allows the solver to adjust the Tucker factors such that the effective interaction graph remains acyclic.

3 Anomaly Detection Strategy

Once the structure ($\mathbf{W}$) and parameters ($\mathcal{G}, \mathbf{U}$) are learned from a reference period, we detect anomalies using a sliding window approach.

3.1 Topological Anomaly Detection

For a window t, we estimate a local adjacency matrix $\mathbf{W}_t$. We compare this to the reference $\mathbf{W}_{ref}$ using the Spectral Distance of the Normalized Laplacian $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$:

$$D_{spec}(t) = \sqrt{\sum_{k=1}^{d} (\lambda_k(\mathcal{L}_t) - \lambda_k(\mathcal{L}_{ref}))^2} \tag{10}$$

This metric is robust to minor noise but sensitive to structural changes, such as the breaking of a causal link.

3.2 Root Cause Localization

Upon detecting an anomaly, we compute the Root Cause Score (RCS) for each node i. The score aggregates the node's local reconstruction error $e_{t,i}$ with the errors of its children, weighted by the causal strength:

$$RCS_i = e_{t,i}^2 + \sum_{j \in Children(i)} W_{ji} \cdot e_{t,j}^2 \tag{11}$$

This formulation leverages the causal graph to distinguish between source errors and propagated symptoms.

4 Conclusion

The Tucker-Dynotears framework provides a mathematically rigorous method for learning non-linear causal structures. By embedding the Tucker decomposition within the Dynotears constraint formulation, we achieve a model that is both expressive enough to capture complex industrial dynamics and constrained enough to yield interpretable, acyclic causal graphs.

References

[1] X. Zheng et al., "DAGs with NO TEARS: Continuous optimization for structure learning," *NeurIPS*, 2018.

[2] R. Pamfil et al., "DYNOTEARS: Structure Learning from Time-Series Data," *AISTATS*, 2020.