Yale University, Department of Statistics

Graphical component analysis via score matching

Natalie Doss

December 6, 2019

Contents

1 Introduction

This sheet summarizes the work we have done so far for graphical component analysis, a new method we are proposing for latent signal detection.

Uncovering hidden structure in data is a central objective of machine learning. Observed data are commonly a transformation of an underlying signal that twe wish to detect. Independent component analysis (ICA) addresses this setting when the signal is generated with independent components, followed by a linear transformation. The key assumption of ICA that the latent signals are independent is often unrealistic. We propose *graphical component analysis* (GCA), which allows for the latent source components to be dependent, with the dependence modeled by a sparse graphical model.

Now fitting a high-dimensional graphical model is usually intractable. We circumvent this issue via score matching, which allows for the estimation of a graphical model without computation of the normalizing constant. We present algorithms that jointly estimate the graphical model and mixing transformation, where the graphical model is represented in terms of an orthogonal polynomial basis, and the mixing matrix is estimated using a coordinate descent procedure based on the use of Givens rotations. GCA generalizes both ICA where the components are independent, and tree component analysis Bach & Jordan (2003) (TCA) where they are governed by a graph with no cycles. Our experiments demonstrate how the presence of cycles in the latent graph can result in failures by both ICA and TCA to recover accurate models of the marginal components.

2 Approach

Let $A \in \mathbb{R}^{d\times d}$ be invertible, and define $W = A^{-1}$. Suppose we observe data points $x_1, \ldots, x_n$, where each x_i is in $\mathbb{R}^d$, and each has the form

$$x = As. \tag{1}$$

Let the true density of the observed data x be p. Suppose we have a model with density $p_\theta(x)$ and we wish to estimate θ. The classical score matching objective, see Hyvarinen (2005), is

$$\mathcal{L}(\theta) = \frac{1}{2}\mathbb{E}\|\nabla_x \log p(x) - \nabla_x \log p_\theta(x)\|_2^2$$

As shown in Hyvarinen (2005), using integration by parts, it turns out that the score-matching objective simplifies to something we can optimize in θ:

$$\mathcal{L}(\theta) =^c \frac{1}{2}\mathbb{E}\|\nabla_x \log p_\theta(x)\|_2^2 + \mathbb{E}\,\mathrm{tr}\left(\nabla_x^2 \log p_\theta(x)\right). \tag{2}$$

This has a further simplification if p_θ is an exponential family, as we discuss below and as shown in Hyvarinen (2005).

The model (1) is equivalent to $x = DA\tilde{s}$ where D is a diagonal matrix with non-zero entries and $\tilde{s}$ is the correspondingly-scaled version of s. That is, the model is only identifiable up to scaling. We assume $W^TW = WW^T = I_d$ to handle this problem. Note also for any permutation matrix P, $x = AP^{-1}Ps$, i.e., the model is identifiable up to permutation. We estimate the graphical model up to the equivalence class under permutation of the nodes.

Let $G = (V, E)$ be a graph with $|V| = d$. We assume that we observe $x_1, \ldots, x_n$ transformed as in (1), where $s_1, \ldots, s_n$ are d-dimensional vectors. In the GCA model, s has distribution $\mathbb{P}_\theta$ with density

$$p_\theta(s) = \exp(\theta^T\phi(s) - \Psi), \tag{3}$$

where $\phi : \mathbb{R}^d \to \mathbb{R}^k, \theta \in \mathbb{R}^k$, and $\Psi = \int_{s\in\mathbb{R}^d} \exp(\theta^T\phi(s))\,ds$. We also write $g_\theta(s) = \theta^T\phi(s)$. The distribution for x therefore has density:

$$p_{W,\theta}(x) = |W|p_\theta(Wx) \tag{4}$$

In our GCA approach, we optimize the score matching objective (2) in W and θ. Using the transformation (4) and the chain rule,

$$\nabla_x \log p_{\theta,W}(x) = \nabla_x\left(\log p_\theta(Wx) + \log|W|\right) = W^T\left(\nabla \log p_\theta(Wx)\right).$$

And

$$\nabla_x^2 \log p_{\theta,W}(x) = W^T \left(\nabla^2 \log p_\theta(Wx)\right) W.$$

Now $\log p_\theta(x) = g_\theta(x) - \Psi$, and as in the usual score matching, the normalizer disappears when we take the derivative. Plugging into (2) and using the assumption that $WW^T = I_d$, our objective becomes

$$\mathcal{L}(\theta, W) = \frac{1}{2}\mathbb{E}\left(\nabla g_\theta(Wx)^T \nabla g_\theta(Wx)\right) + \mathbb{E}\,\mathrm{tr}\left(\nabla^2 g_\theta(Wx)\right). \tag{5}$$

For a function $g : \mathbb{R}^d \to \mathbb{R}$ and for $u, v \in V$, let $\partial_v g = \partial g/\partial s_v$, and $\partial_{uv} g = \partial^2 g/\partial s_u \partial s_v$. And for a vector-valued function $\psi : \mathbb{R}^d \to \mathbb{R}^k$, let $\partial_v \psi = (\partial_v \psi_1, \ldots, \partial_v \psi_m)^T$, and so on. Define $\Delta g(s) = \sum_{v\in V} \partial_{vv}^2 g(s)$. Now $\nabla g = \left(\theta^T \partial_1 \phi, \ldots, \theta^T \partial_p \phi\right)^T$ and $(\nabla^2 g)_{vv} = \langle \theta,\, \partial_{vv}^2 \phi\rangle$. Define

$$\begin{aligned} C(Wx) &= \sum_{v\in V} \partial_v \phi(Wx) \partial_v \phi(Wx)^T. \\ \xi(Wx) &= \sum_{v\in V} \partial_{vv}^2 \phi(Wx). \end{aligned} \tag{6}$$

Then we see that our population score matching objective becomes just similar to the usual one discussed in Section **??**, except that now it contains W:

$$\mathcal{L}(\theta, W) = \frac{1}{2}\theta^T \mathbb{E} C(Wx)\theta + \theta^T \mathbb{E}\xi(Wx).$$

In practice, we use the empirical expectation. Let $\hat{\mathbb{E}} f(x) = \sum_{i\in[n]} f(x_i)/n$. Let $\hat{C}_W = \hat{\mathbb{E}} C(Wx)$ and $\hat{\xi}_W = \hat{\mathbb{E}}\xi(Wx)$. We add in the penalty $\|\theta\|_2^2$ to discourage too many edges. Now define

$$\hat{\mathcal{L}}(\theta, W) = \frac{1}{2}\theta^T \hat{C}_W \theta + \theta^T \hat{\xi}_W + \frac{\lambda}{2}\|\theta\|_2^2$$

Since we have the added constraint that W must be orthogonal, the score-matching objective becomes:

$$\min_{\theta, W} \hat{\mathcal{L}}(\theta, W) \text{ s.t. } WW^T = I_d$$

Fix W. The objective is quadratic in θ, and there is a closed-form solution:

$$\hat{\theta} = \left(\hat{C}_W + \lambda I_k\right)^{-1}(-\hat{\xi}_W). \tag{7}$$

Alternatively, as in Janofsky (2015), we may impose $\theta_{vu} = \theta_{uv}$, as well as the group sparsity penalty:

$$\hat{\mathcal{L}}(\theta, W) = \frac{1}{2}\theta^T \hat{C}_W \theta + \theta^T \hat{\xi}_W + \sum_{(u,v)\in E} \|\theta_{u,v}\|_2$$

In the simulations, we use the latter. When we have a graphical model on the components of s, g_θ can be expressed as an undirected pairwise graphical model and that we expand the potential functions with a truncated orthogonal basis. Let us have m_1 basis elements for the individual node potential functions and m_2 basis elements for the pairwise potential functions. Then

$$g_\theta(s) = \sum_{v \in V} \sum_{j \in [m_1]} \theta_v^{(j)} \phi_v^{(j)}(s_v) + \sum_{(u,v) \in E} \sum_{j \in [m_2]} \theta_{u,v}^{(j)} \phi_{uv}^{(j)}(s_u, s_v).$$

Thus, for us, $k = m_1 d + m_2 d(d-1)$.

3 Algorithm

We iterate over optimizing $\hat{\mathcal{L}}(\theta, W)$ in θ and in W. The optimization in θ has the simple solution given in (7). To optimize the orthogonal matrix W, we use the Givens rotation method of Shalit & Chechik (2014). This method reduces the problem of optimizing the matrix to $O(d^2)$ sub-problems of optimizing rotations of the matrix. In each rotation, a pair of nodes is selected and the angle of rotation is optimized. This is a one-dimensional optimization. For $u, v \in V$, define

$$G_{u,v}(\eta) = \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & \cos\eta & \dots & -\sin\eta & \dots & 0 \\ \vdots & & & & & \vdots & \\ 0 & \dots & \sin\eta & \dots & \cos\eta & \dots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{pmatrix} \tag{8}$$

That is, it has 1's down the diagonal except for columns and zeros everywhere else except for the (u,u), (v,v), (u,v), and (v,u) entries, where it has $\cos\eta, \cos\eta, -\sin\eta, \sin\eta$, respectively. Fix a pair (u,v). Write G_η for simplicity.

Algorithm 1: GCA. Define $\hat{C}_W, \hat{\xi}_W, \hat{\mathcal{L}}(\theta, W)$ as in (6). Let $G_{u,v}(\eta)$ be as defined in (11). See 6.2 for how to find $\text{argmin}_\eta \hat{\mathcal{L}}(\theta, W_t G_{u,v}(\eta))$.

```
Input: Dataset $\{x_i\}_{i\in[n]}$, each point in $\mathbb{R}^d$, penalty coefficient $\lambda \in \mathbb{R}$, cutoff $\epsilon \in \mathbb{R}^+$,
       number of steps $T$
Output: $\theta \in \mathbb{R}^k$, $W \in \mathbb{R}^{d\times d}$.
Initialize $W_0$ satisfying $WW^T = I_d$ ;
While $\hat{\mathcal{L}}(\theta, W) > \epsilon$ ;
  $\theta = \left(\hat{C}_W + \lambda I_k\right)^{-1}\left(-\hat{\xi}_W\right)$ ;
  While not converged and $t \in [T]$, do ;
    Select pair of indices $(u(t), v(t))$ satisfying $1 \leq u(t) < v(t) \leq p$ ;
    $\eta_{t+1} = \text{argmin}_\eta \hat{\mathcal{L}}(\theta, W_t G_{u,v}(\eta))$ ;
    $W_{t+1} = W_t G_{u,v}(\eta_{t+1})$ ;
    $t = t + 1$ ;
```

4 Experiments

Our GCA algorithm produces an estimate of W and an estimate of θ. Note that by the formula (4),

$$p_{W,\theta}(x) = |W| p_\theta(Wx).$$

To compare GCA, TCA, and ICA, we generate synthetic source data s with three types of graphical model structures: independent, tree, and cycles with no more than three components. We first generate Gaussian data with the required graphical model structure, then transform it to be non-Gaussian. In all three, we compare the algorithms in terms of heldout log-likelihood, where the log-likelihood is approximated using a kernel density estimate. This is feasible since components have size no more than three. A sample result is in Figure 1 on page 6. This is for $d = 15$, and we currently only compare to ICA. We can see that we about match ICA on independent data, and surpass it on tree and cycles data, with improvement as sample size (the x-axis) increases.

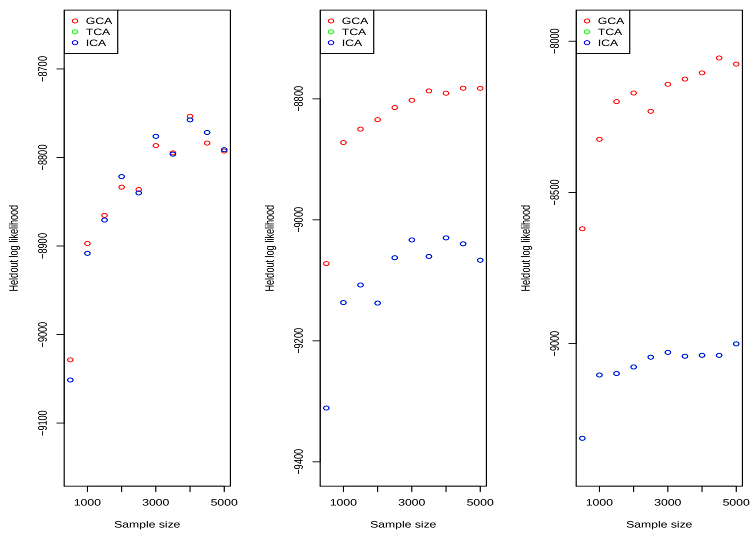
Alternatively, the Fisher divergence does not require a normalizing constant. We might consider computing the Fisher divergence on a holdout dataset x_{test},

$$\hat{\mathbb{E}}_{x_{test}} \left(\frac{1}{2} \hat{\theta}^\top C(\hat{W} x_{test}) \hat{\theta} + \hat{\theta}^\top \xi(\hat{W} x_{test}) \right).$$

5 Alternative: neural network approach

Score-matching in the form of (2) is not the only possibility for optimizing while avoiding computation of the normalizing constant. Saremi *et al.* (2018) proposes "Deep Energy Estimator Networks," (DEEN). The method, discussed below, avoids Hessian computations that are an encumbrance in ordinary score matching. We now discuss this method and how we use it to implement GCA.

Figure 1: GCA vs. ICA on various graphical models.



Let nl be the number of hidden layers and let ℓ be the number of units per hidden layer; I assume we use the same number per layer for simplicity. Let $W_1 \in \mathbb{R}^{p\times\ell}, W_a \in \mathbb{R}^{\ell\times\ell}$ for $a \in [nl]$.

$$\log p_{M,b}(s) = -M_3\sigma(M_2\sigma(M_1 s + b_1) + b_2) + b_3. \tag{9}$$

The new score matching optimizes:

$$\mathcal{L}(M,b) = \sum_{i\in[n],j\in[m]} \|x_i - \xi_{i,j} - \sigma^2\nabla_\xi \log p_{M,b}(\xi_{ij})\|_2^2.$$

There are a few interpretations of this; one is that essentially, the first term is like ∇_ξ log kernel density estimate if we use Gaussian kernel. Now for us, using the Saremi *et al.* (2018) objective but with our gca formulation. Let $m \in \mathbb{N}$. For each $i \in [n]$, $j \in [m]$,

$$s_i \sim_{i.i.d.} P_S \text{ with density } p_S.$$
$$\xi_{s,i,j} = s_i + z_j, \text{ where } z_j \sim_{i.i.d.} N(0, I_p).$$

Then we have an orthonormal matrix A, and we observe the data:

$$x_i = As_i.$$
$$\xi_{x,i,j} = A\xi_{s,i,j} = As_i + Az_j = As_i + N(0, I_p).$$

We optimize

$$\begin{aligned}\mathcal{L}(W,M,b) &= \sum_{i\in[n],j\in[m]} \|x_i - \xi_{x,i,j} - \sigma^2\nabla_\xi \log p_{M,b}(\xi_{x,i,j})\|_2^2 \\ &= \sum_{i\in[n],j\in[m]} \|x_i - \xi_{x,i,j} - \sigma^2 W^\top \nabla_{\xi_s} \log p_{M,b}(W\xi_{x,i,j})\|_2^2.\end{aligned}$$

This is optimized using stochastic gradient descent. We still used the Givens rotation representation to optimize W. We adapt the method and code of Jing *et al.* (2016) to represent W as a product of orthonormal matrices and to optimize. Our implementation is a package provided on github.

6 Supplementary

6.1 Pairwise graphical model

Suppose now that we have an undirected pairwise graphical model and that we expand the potential functions with a truncated orthogonal basis. Let us have m_1 basis elements for the individual node potential functions and m_2 basis elements for the pairwise potential functions. Then

$$\begin{aligned} g(s) &= \sum_{v \in V} \psi_v(s_v) + \sum_{(u,v) \in E} \psi_{u,v}(s_u, s_v) \\ &= \sum_{v \in V} \sum_{k \in [m]} \theta_v^k \phi_v^{(k)}(s_v) + \sum_{(u,v) \in E} \sum_{k \in [m_2]} \theta_{u,v}^k \phi_{uv}^{(k)}(s_u, s_v) \end{aligned}$$

There are a few simple ways to write J, C, ξ, θ, ϕ in such a model. Here is my preferred way. As in Janofsky (2015), we define the vectors

$$\begin{aligned} \theta_v &= \left(\theta^{(1)}, \dots, \theta^{(m_1)}\right)^T \in \mathbb{R}^{m_1} \\ \theta_{vu} &= \left(\theta_{vu}^{(1)}, \dots, \theta_{vu}^{(m_2)}\right)^T \in \mathbb{R}^{m_2} \text{ for } u \neq v \\ \theta_{v\cdot} &= \left(-\theta_{v1}^T-, \dots, -\theta_v^T-, \dots, -\theta_{vp}^T-\right)^T \in \mathbb{R}^{m_1 + m_2(p-1)} \\ \theta &= \left(-\theta_{1\cdot}-^T, \dots, -\theta_{p\cdot}^T-\right)^T \in \mathbb{R}^{m_1 p + m_2 p(p-1)} \end{aligned} \tag{10}$$

Here, we let $\theta_{uv} = \theta_{vu}$. We define the vectors $\phi_v, \phi_{vu}, \phi_{v\cdot}$, and ϕ analogously. Now $\partial_v \phi = (0, \dots, 0, \partial_v \phi_{\cdot v}, 0, \dots, 0)$. Given these definitions, we can write the terms from (6) via the following. Everything here is a function of Wx, which I drop for simplicity. Now,

$$J(Wx) = \begin{pmatrix} \begin{matrix} | \\ \partial_1 \phi_{1\cdot} \\ | \end{matrix} & 0 & & 0 \\ 0 & \begin{matrix} | \\ \partial_2 \phi_{2\cdot} \\ | \end{matrix} & 0 & 0 \\ & & \ddots & \\ 0 & 0 & 0 & \begin{matrix} | \\ \partial_p \phi_{p\cdot} \\ | \end{matrix} \end{pmatrix}$$

and

$$C(Wx) = \begin{pmatrix} \partial_1\phi_{1\cdot}\partial_1\phi_{1\cdot}^T & \dots & \dots \\ \dots & \partial_2\phi_{2\cdot}\partial_2\phi_{2\cdot}^T & \dots \\ \ddots & & \\ \dots & \dots & \partial_p\phi_{p\cdot}\partial_p\phi_{p\cdot}^T \end{pmatrix}$$

and

$$\xi(Wx) = \left(-\partial_{11}\phi_{1\cdot}^T-, \dots - \partial_{pp}\phi_{p\cdot}^T-\right)^T$$

Note that $\theta_{vu} = \theta_{uv}$, and $\theta \in \mathbb{R}^{m_1 p + m_2 p(p-1)}$. Note that we can write the objective as

$$\mathcal{L}(\theta, W, x) = \sum_{v \in V} \theta_{v\cdot}^T \partial_v \phi_{v\cdot}(Wx) \partial_v \phi_{v\cdot}(Wx)^T \theta_{v\cdot} + \theta_{v\cdot}^T \partial_{vv}^2 \phi_{v\cdot}(Wx)$$

6.2 W optimization: Givens rotation

The Givens orthogonal rotation algorithm is in Algorithm 1. The main thing to figure out is how to find $\operatorname{argmin}_\eta \hat{\mathcal{L}}\left(WG_{u,v}(\eta)\right)$; for this, we compute the first derivative of the function with respect to η. For $u, v \in V$, define

$$G_{u,v}(\eta) = \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & \cos\eta & \dots & -\sin\eta & \dots & 0 \\ \vdots & & & & & \vdots & \\ 0 & \dots & \sin\eta & \dots & \cos\eta & \dots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{pmatrix} \tag{11}$$

That is, it has 1's down the diagonal except for columns and zeros everywhere else except for the $(u, u), (v, v), (u, v)$, and (v, u) entries, where it has $\cos\eta, \cos\eta, -\sin\eta, \sin\eta$, respectively. Fix a pair (u, v). Write G_η for simplicity. To simplify notation, let $f_v = \partial_v \phi$ and $g_v = \partial_{vv}^2 \phi$. And let $y_\eta := WG_\eta x$. Our objective in η is $\mathbb{E}\mathcal{L}(\theta, WG_\eta, x)$, where

$$\mathcal{L}(\eta, x) = \mathcal{L}(\theta, WG_\eta, x) = \frac{1}{2}\theta^T \left(\sum_{v \in V} f_v(y_\eta) f_v(y_\eta)^T\right) \theta + \theta^T \left(\sum_{v \in V} g_v(y_\eta)\right)$$

Now let $h : \mathbb{R}^p \to \mathbb{R}$, let η be a scalar, and let $y_\eta := y(\eta) = (y_1(\eta), \dots, y_p(\eta))^T \in \mathbb{R}^p$ be a vector depending on η. Then $\partial_\eta h(y_\eta) = \langle \nabla h(y_\eta), \partial_\eta y_\eta \rangle$. For us, $\partial_\eta y_\eta = W \partial_\eta G_\eta x$, where

$$\partial_\eta G_{u,v}(\eta) = \begin{pmatrix} 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & -\sin\eta & \dots & -\cos\eta & \dots & 0 \\ \vdots & & & & & \vdots & \\ 0 & \dots & \cos\eta & \dots & -\sin\eta & \dots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & 0 & \dots & 0 \end{pmatrix}.$$

This is the matrix whose entries are the derivatives of the entries of G_η with respect to η. Now f_v, g_v are functions from $\mathbb{R}^p \to \mathbb{R}^K$ (so they're vectors in $\mathbb{R}^K$). And

$$\begin{aligned}\partial_\eta f_v(y_\eta) &\triangleq (\partial_\eta f_{v1}(y_\eta), \ldots, \partial_\eta f_{vK}(y_\eta))^T \\ &= (\langle \nabla f_{v1}(y_\eta),\, \partial_\eta y_\eta \rangle, \ldots, \langle \nabla f_{vK}(y_\eta),\, \partial_\eta y_\eta \rangle)^T \\ &= J_{f_v}(y_\eta)\partial_\eta y_\eta\end{aligned}$$

Note that by the product rule applied to each element of the matrix $f_v(y_\eta)f_v(y_\eta)^T$, $\partial_\eta\left(\theta^T\left(f_v f_v)^T\right)\theta\right) = \theta^T\left(\left(\partial_\eta f_v\right)f_v^T + f_v\left(\partial_\eta f_v\right)^T\right)\theta$. Thus

$$\partial_\eta \mathcal{L}(\theta, WG_\eta, x) = \frac{1}{2}\theta^T\left(\sum_{v\in V} J_{f_v}(y_\eta)\partial_\eta y_\eta f_v(y_\eta)^T + f_v(y_\eta)\partial_\eta y_\eta^T J_{f_v}(y_\eta)^T\right)\theta + \theta^T\left(\sum_{v\in V} J_{g_v}(y_\eta)\partial_\eta y_\eta\right)$$

So now assuming we can bring the derivative inside the expectation, $\partial_\eta \mathcal{L}(\theta, WG_\eta) = \mathbb{E}\partial_\eta \mathcal{L}(\theta, WG_\eta, x)$.

6.2.1 Graphical model form for Givens rotation

Let $f_v = \partial_v \phi_{v\cdot}$ and $g_v = \partial_{vv}^2 \phi_{v\cdot}$. We can now write our objective in η in the form

$$\mathcal{L}(\theta, WG_\eta, x) = \frac{1}{2}\sum_{v\in V}\theta_{v\cdot}^T f_v(y_\eta)f_v(y_\eta)^T\theta_{v\cdot} + \sum_{v\in V}\theta_{v\cdot}^T g_v(y_\eta)$$

And

$$\partial_\eta \mathcal{L}(\theta, WG_\eta, x) = \frac{1}{2}\sum_{v\in V}\theta_{v\cdot}^T\left(J_{f_v}(y_\eta)\partial_\eta y_\eta f_v(y_\eta)^T + f_v(y_\eta)\partial_\eta y_\eta^T J_{f_v}(y_\eta)^T\right)\theta_{v\cdot} + \sum_{v\in V}\theta_{v\cdot}^T J_{g_v}(y_\eta)\partial_\eta y_\eta$$

References

Bach, Francis R., & Jordan, Michael I. 2003. Beyond independent components: trees and clusters. *Journal of Machine Learning Research*, **4**, 1205–1233.

Hyvarinen, Aapo. 2005. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, **6**, 695–709.

Janofsky, Eric. 2015. *Exponential series approaches for nonparametric graphical models.* Ph.D. thesis, University of Chicago.

Jing, Li, Shen, Yichen, Dubcek, Tena, Peurifoy, John, Skirlo, Scott A., Tegmark, Max, & Soljacic, Marin. 2016. Tunable Efficient Unitary Neural Networks (EUNN) and their application to RNN. *CoRR*, **abs/1612.05231**.

Saremi, Saeed, Mehrjou, Arash, Schölkopf, Bernhard, & Hyvärinen, Aapo. 2018. Deep Energy Estimator Networks. *ArXiv*, **abs/1805.08306**.

Shalit, Uri, & Chechik, Gal. 2014. Coordinate-descent for Learning Orthogonal Matrices Through Givens Rotations. *Pages I–548–I–556 of: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32.* ICML'14. JMLR.org.