Graphical Normalizing Flows

Antoine Wehenkel & Gilles Louppe
antoine.wehenkel@uliege.be

LIÈGE université

fnrs FREEDOM TO RESEARCH

AISTATS

What? We improve continuous Bayesian networks with normalizing flows.

How? With a new conditioner that generalizes coupling and autoregressive conditioners.

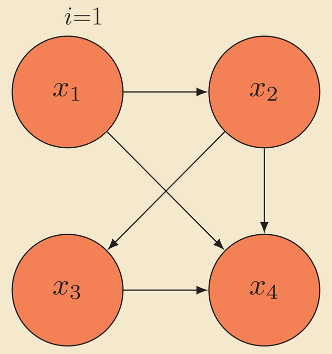
Applications? Learning the topology of Bayesian networks, Make normalizing flows more interpretable.

Code

Arxiv 2006.02548

Bayesian Networks

A Bayesian network is a directed acyclic graph that factorizes the model distribution as $p(\mathbf{x}) = \prod_{i=1}^{D} p(x_i|\mathcal{P}_i)$.



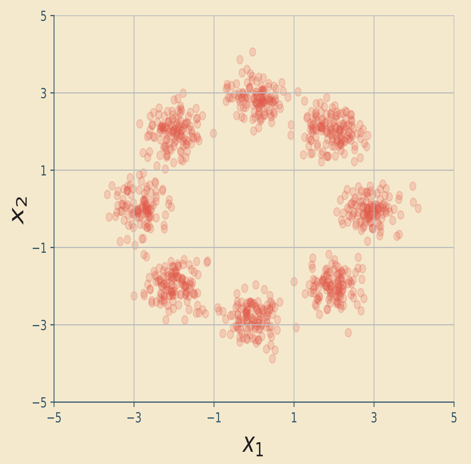
$$\iff p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_1,x_2,x_3)$$

Pros
- Good for modeling independencies and checking their impact on the modeled density.
- Applications across science and technology.

Cons
- Often used with discrete or discretized data.
- Outdated with respect to the deep learning revolution.

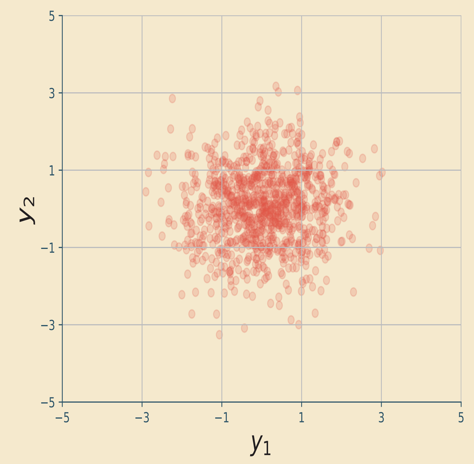
Normalizing Flows

A normalizing flow is a sequence of K invertible transformation steps $\mathbf{g}_k : \mathbb{R}^d \rightarrow \mathbb{R}^d$ composed together to create an expressive invertible mapping $\mathbf{g}(\mathbf{x})$. Density estimation is performed via the change of variables theorem: $p(\mathbf{x}) = p_{\mathbf{y}}(\mathbf{g}(\mathbf{x})) \left|\det J_{\mathbf{g}(\mathbf{x})}\right|$.



Density Estimation →

← Sampling



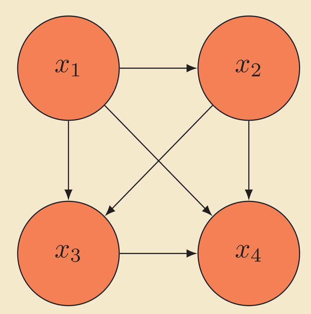
A transformation step can be written as: $\mathbf{g}(\mathbf{x}) = \left[g^1(x_1; \mathbf{c}^1(\mathbf{x})) \;\ldots\; g^d(x_d; \mathbf{c}^d(\mathbf{x}))\right]^T$.

Normalizer (e.g. affine or monotonic)

Conditioner (ensure a simple Jacobian)

Graphical Conditioners

Autoregressive conditioner

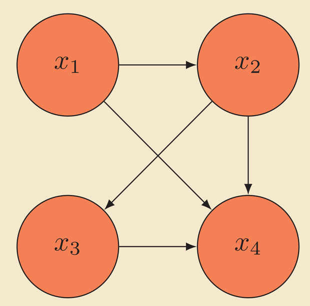


$$\mathbf{c}^i(\mathbf{x}) = \mathbf{h}^i\left(\begin{bmatrix} x_1 & \ldots & x_{i-1}\end{bmatrix}^T\right)$$

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 3 & 1 & 1 & 0 & 0 \\ 4 & 1 & 1 & 1 & 0 \end{array}$$

Generalization →

Graphical conditioner



$$\mathbf{c}^i(\mathbf{x}) = \mathbf{h}^i(\mathbf{x} \odot A_{i,:})$$

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 \\ 4 & 1 & 1 & 1 & 0 \end{array}$$

These conditioners can be generalized to any Bayesian network topology. Formally, let $A \in \{0,1\}^{d\times d}$ be the adjacency of a Bayesian network, the graphical conditioner is defined as $\mathbf{c}^i(\mathbf{x}) = \mathbf{h}^i(\mathbf{x} \odot A_{i,:})$.

Topology learning

Choosing the network topology is not always easy!

But learning a good topology can be cast as a continuous optimization problem:

$$\max_{A \in \mathbb{R}^{d\times d}} F(A) \quad \text{s.t.} \quad \mathcal{G}(A) \in \text{DAGs} \iff \max_{A \in \mathbb{R}^{d\times d}} F(A) \quad \text{s.t.} \quad w(A) = 0$$

where $F(A) = \sum_{j=1}^{N} \log\left(p(\mathbf{x}^j)\right) + \lambda_{\ell_1}||A||_1$.

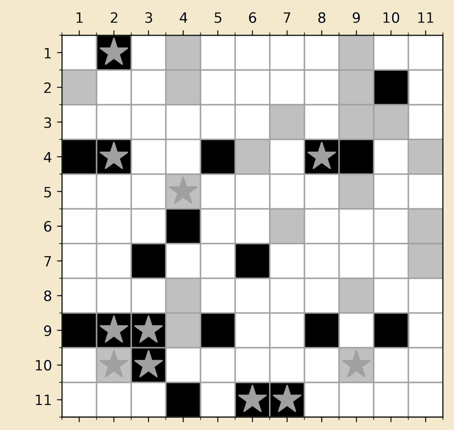
The log-density is evaluated with a graphical normalizing flow, introducing the neural networks parameters it can be written as $p(\mathbf{x};\theta) = p_{\mathbf{z}}(\mathbf{g}(\mathbf{x};\theta)) \prod_{i=1}^{d} \left|\frac{\partial g^i(x_i; \mathbf{h}^i(\mathbf{x} \odot A_{i,:}), \theta)}{\partial x_i}\right|$.

The acyclicity constraint is expressed as $w(A) := \text{tr}\left((I + \alpha A)^d\right) - d \propto \text{tr}\left(\sum_{k=1}^{d} \alpha^k A^k\right)$.

Lagrangian formulation: $\max_A \mathbb{E}_{\gamma_1,\gamma_2}[F(A)] - \lambda_t w(A) - \frac{\mu_t}{2} w(A)^2$.

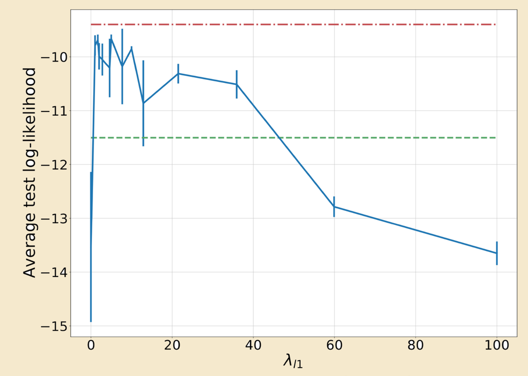
The importance of graph topology

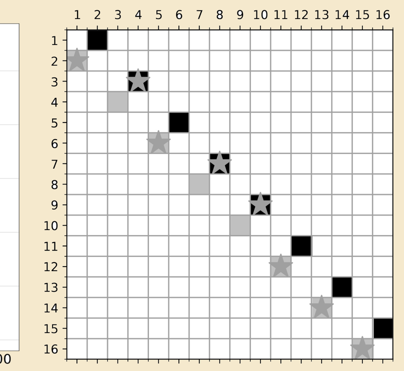
Protein interaction network



On real data, the optimization recovers (stars) a graph that is similar to the one designed by experts.

8 pairs of 2D toy problems





(left) Test log-likelihood as a function of l1-penalization. Red: Correct topology. Green: Autoregressive conditioner.
(right) Recovered topology.
Learning the right topology leads to better results than autoregressive conditioners.

Density estimation

Dataset	d	V	Train	Test
Arithmetic Circuit	8	8	10,000	5,000
8 Pairs	16	8	10,000	5,000
Tree	7	8	10,000	5,000
Protein	11	20	6,000	1,466
POWER	6	≤ 15	1,659,917	204,928
GAS	8	≤ 28	852,174	105,206
HEPMASS	21	≤ 210	315,123	174,987
MINIBOONE	43	≤ 903	29,556	3,648
BSDS300	63	$\leq 1,953$	1,000,000	250,000

We tested density estimation performance on many diverse datasets. **Graphical conditioners outperform coupling and autoregressive flows.**

Prescribed topology

Conditioner	Graphical	Autoreg.
Arithmetic Circuit	$\mathbf{3.99}_{\pm.16}$	$3.06_{\pm.38}$
8 Pairs	$\mathbf{-9.40}_{\pm.06}$	$-11.50_{\pm.27}$
Tree	$\mathbf{-6.85}_{\pm.02}$	$-6.96_{\pm.05}$
Protein	$6.46_{\pm.08}$	$\mathbf{7.52}_{\pm.10}$

Learned topology

	Dataset	POWER	GAS	HEPMASS	MINIBOONE	BSDS300
	Coup.	$-5.60_{\pm.00}$	$-3.05_{\pm.01}$	$-25.74_{\pm.01}$	$-38.34_{\pm.02}$	$57.33_{\pm.00}$
(a)	Auto.	$-3.55_{\pm.00}$	$-0.34_{\pm.01}$	$-21.66_{\pm.01}$	$\mathbf{-16.70}_{\pm.05}$	$\mathbf{63.74}_{\pm.00}$
	Graph.	$\mathbf{-2.80}_{\pm.01}$	$\mathbf{1.99}_{\pm.02}$	$\mathbf{-21.18}_{\pm.07}$	$-19.67_{\pm.06}$	$62.85_{\pm.07}$
	Coup.	$0.25_{\pm.00}$	$5.12_{\pm.03}$	$-20.55_{\pm.04}$	$-32.04_{\pm.12}$	$107.17_{\pm.46}$
(b)	Auto.	$0.58_{\pm.00}$	$9.79_{\pm.04}$	$-14.52_{\pm.16}$	$\mathbf{-11.66}_{\pm.02}$	$151.29_{\pm.31}$
	Graph.	$\mathbf{0.62}_{\pm.04}$	$\mathbf{10.15}_{\pm.15}$	$\mathbf{-14.17}_{\pm.13}$	$-16.23_{\pm.52}$	$\mathbf{155.22}_{\pm.11}$

Take home messages

- Continuous Bayesian networks can be combined with deep generative models.
- A correct prescribed topology improves the performance of normalizing flows.
- It is possible to discover relevant Bayesian network topology with graphical normalizing flows.