Transport-Regularized Normalizing Flows[1234]

"Learning by Forgetting"

J. Setpal

April 30, 2025

[1] Peyré, Cuturi. [Arxiv 2020]
[2] Kobyzev, Prince, Brubaker. [IEEE 2021]
[3] Papamakarios, et. al. [Arxiv 2021]
[4] Lai, et. al. [Journal of Computational Physics 2023]

Outline

1. Dynamic Optimal Transport
2. Normalizing Flows
3. Transport-Regularized Normalizing Flows

Outline

1. Dynamic Optimal Transport
2. Normalizing Flows
3. Transport-Regularized Normalizing Flows

Dynamic Optimal Transport (1/2)

We have explored the following Optimal Transport problem:

$$L_{\boldsymbol{C}}(a, b) := \min_{\boldsymbol{P} \in \mathcal{U}(a,b)} \langle \boldsymbol{C}, \boldsymbol{P} \rangle_F = \sum_{i,j} \boldsymbol{C}_{i,j} \boldsymbol{P}_{i,j} \tag{1}$$

Dynamic Optimal Transport (1/2)

We have explored the following Optimal Transport problem:

$$L_{\boldsymbol{C}}(a, b) := \min_{\boldsymbol{P} \in \mathcal{U}(a,b)} \langle \boldsymbol{C}, \boldsymbol{P} \rangle_F = \sum_{i,j} \boldsymbol{C}_{i,j} \boldsymbol{P}_{i,j} \tag{1}$$

This notion has a couple of properties / constraints:

1. $\mathcal{U}$ represents the set of valid couplings, which encapsulates criteria:
    a. Mass is conserved.
    b. Applying the coupling gets us the target mesures: $\beta = \boldsymbol{P}_\sharp \alpha$
2. The optimization problem is *convex*.
3. If $\boldsymbol{C}$ is a distance in element space, $L_{\boldsymbol{C}}$ is a distance in measure space.

Dynamic Optimal Transport (1/2)

We have explored the following Optimal Transport problem:

$$L_{\boldsymbol{C}}(a, b) := \min_{\boldsymbol{P} \in \mathcal{U}(a,b)} \langle \boldsymbol{C}, \boldsymbol{P} \rangle_F = \sum_{i,j} \boldsymbol{C}_{i,j} \boldsymbol{P}_{i,j} \tag{1}$$

This notion has a couple of properties / constraints:

1. $\mathcal{U}$ represents the set of valid couplings, which encapsulates criteria:
    a. Mass is conserved.
    b. Applying the coupling gets us the target mesures: $\beta = \boldsymbol{P}_\sharp \alpha$
2. The optimization problem is *convex*.
3. If $\boldsymbol{C}$ is a distance in element space, $L_{\boldsymbol{C}}$ is a distance in measure space.

Observation: $\boldsymbol{C} = \| \cdot \|_2^2 \implies L_{\boldsymbol{C}}$ is **squared geodesic distance**.

Dynamic Optimal Transport (1/2)

We have explored the following Optimal Transport problem:

$$L_{\boldsymbol{C}}(a,b) := \min_{\boldsymbol{P}\in\mathcal{U}(a,b)} \langle \boldsymbol{C}, \boldsymbol{P} \rangle_F = \sum_{i,j} \boldsymbol{C}_{i,j}\boldsymbol{P}_{i,j} \tag{1}$$

This notion has a couple of properties / constraints:

1. $\mathcal{U}$ represents the set of valid couplings, which encapsulates criteria:
    a. Mass is conserved.
    b. Applying the coupling gets us the target mesures: $\beta = \boldsymbol{P}_\sharp \alpha$
2. The optimization problem is *convex*.
3. If $\boldsymbol{C}$ is a distance in element space, $L_{\boldsymbol{C}}$ is a distance in measure space.

Observation: $\boldsymbol{C} = \|\cdot\|_2^2 \implies L_{\boldsymbol{C}}$ is **squared geodesic distance**.
Implication: Solving for Endpoints → Interpolatable Transport.

Dynamic Optimal Transport (1/2)

We have explored the following Optimal Transport problem:

$$L_{\boldsymbol{C}}(a, b) := \min_{\boldsymbol{P} \in \mathcal{U}(a,b)} \langle \boldsymbol{C}, \boldsymbol{P} \rangle_F = \sum_{i,j} \boldsymbol{C}_{i,j} \boldsymbol{P}_{i,j} \tag{1}$$

This notion has a couple of properties / constraints:

1. $\mathcal{U}$ represents the set of valid couplings, which encapsulates criteria:
   a. Mass is conserved.
   b. Applying the coupling gets us the target mesures: $\beta = \boldsymbol{P}_\sharp \alpha$
2. The optimization problem is *convex*.
3. If $\boldsymbol{C}$ is a distance in element space, $L_{\boldsymbol{C}}$ is a distance in measure space.

Observation: $\boldsymbol{C} = \| \cdot \|_2^2 \implies L_{\boldsymbol{C}}$ is **squared geodesic distance**.
Implication: Solving for Endpoints $\rightarrow$ Interpolatable Transport.

How? Using fluid dynamics!

Dynamic Optimal Transport (2/2)

The Dynamic[5] Optimal Transport enables us to borrow fluid dynamics literature, and understand how the measure evolves as *time* progresses:

$$\text{Let} \quad \mathcal{X}, \mathcal{Y} \in \mathbb{R}^d, \quad \boldsymbol{C}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 \tag{2}$$

$$\text{With measures } \alpha_t \quad \text{s.t.} \quad T_\sharp \alpha_0 = \alpha_1 \quad \forall t \in [0, 1] \tag{3}$$

[5] Adjective AND Noun.

Dynamic Optimal Transport (2/2)

The Dynamic[5] Optimal Transport enables us to borrow fluid dynamics literature, and understand how the measure evolves as *time* progresses:

$$\text{Let} \quad \mathcal{X}, \mathcal{Y} \in \mathbb{R}^d, \quad \boldsymbol{C}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 \tag{2}$$

$$\text{With measures } \alpha_t \quad \text{s.t.} \quad T_\sharp \alpha_0 = \alpha_1 \quad \forall t \in [0, 1] \tag{3}$$

We can describe the path in continuous time using by moving α_t along a vector field v_t.

[5] Adjective AND Noun.

Dynamic Optimal Transport (2/2)

The Dynamic[5] Optimal Transport enables us to borrow fluid dynamics literature, and understand how the measure evolves as *time* progresses:

$$\text{Let} \quad \mathcal{X}, \mathcal{Y} \in \mathbb{R}^d, \quad \boldsymbol{C}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 \tag{2}$$

$$\text{With measures } \alpha_t \quad \text{s.t.} \quad T_\sharp \alpha_0 = \alpha_1 \quad \forall t \in [0, 1] \tag{3}$$

We can describe the path in continuous time using by moving α_t along a vector field v_t. Infinitesimally, our transport cost can be computed as:

$$\|v_t\|_{L^2(\alpha_t)} = \left(\int_{\mathbb{R}^d} \|v_t(\boldsymbol{x})\|^2 d\alpha_t(\boldsymbol{x}) \right)^{1/2} \tag{4}$$

[5] Adjective AND Noun.

Dynamic Optimal Transport (2/2)

The Dynamic[5] Optimal Transport enables us to borrow fluid dynamics literature, and understand how the measure evolves as *time* progresses:

$$\text{Let} \quad \mathcal{X}, \mathcal{Y} \in \mathbb{R}^d, \quad \boldsymbol{C}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 \tag{2}$$

$$\text{With measures } \alpha_t \quad \text{s.t.} \quad T_\sharp \alpha_0 = \alpha_1 \quad \forall t \in [0, 1] \tag{3}$$

We can describe the path in continuous time using by moving α_t along a vector field v_t. Infinitesimally, our transport cost can be computed as:

$$\|v_t\|_{L^2(\alpha_t)} = \left(\int_{\mathbb{R}^d} \|v_t(\boldsymbol{x})\|^2 d\alpha_t(\boldsymbol{x}) \right)^{1/2} \tag{4}$$

Across time t, our net transport cost is:

$$W_2^2(\alpha_0, \alpha_1) = \min_{\alpha_t, v_t} \int_0^1 \int_{\mathbb{R}^d} \|v_t(\boldsymbol{x})\|^2 \; d\alpha_t(\boldsymbol{x}) \; dt \tag{5}$$

[5] Adjective AND Noun.

Dynamic Optimal Transport (2/2)

The Dynamic[5] Optimal Transport enables us to borrow fluid dynamics literature, and understand how the measure evolves as *time* progresses:

$$\text{Let} \quad \mathcal{X}, \mathcal{Y} \in \mathbb{R}^d, \quad \boldsymbol{C}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 \tag{2}$$

$$\text{With measures } \alpha_t \quad \text{s.t.} \quad T_\sharp \alpha_0 = \alpha_1 \quad \forall t \in [0, 1] \tag{3}$$

We can describe the path in continuous time using by moving α_t along a vector field v_t. Infinitesimally, our transport cost can be computed as:

$$\|v_t\|_{L^2(\alpha_t)} = \left(\int_{\mathbb{R}^d} \|v_t(\boldsymbol{x})\|^2 d\alpha_t(\boldsymbol{x}) \right)^{1/2} \tag{4}$$

Across time t, our net transport cost is:

$$W_2^2(\alpha_0, \alpha_1) = \min_{\alpha_t, v_t} \int_0^1 \int_{\mathbb{R}^d} \|v_t(\boldsymbol{x})\|^2 \; d\alpha_t(\boldsymbol{x}) \; dt \tag{5}$$

To satisfy mass conservation, we also enforce the following constraint:

$$\partial_t \alpha_t + \operatorname{div}(\alpha_t v_t) = 0 \tag{6}$$

[5] Adjective AND Noun.

Mean-Field Games

Problem: The mass-conservation constraint, requiring computing $\alpha_t v_t$ makes the problem <u>non-convex</u>.

Mean-Field Games

Problem: The mass-conservation constraint, requiring computing $\alpha_t v_t$ makes the problem non-convex.

Solution: We can reparameterize the problem using Mean-Field Games:

1. MFGs are ∞-agent games with each agent in spatial domain trying to minimize individual cost. We assume a density function at time t.

$$\min_{\alpha_t, v_t} T(\alpha_0, \alpha_1) + \int_0^1 \int_{\mathbb{R}^d} L(\boldsymbol{x}, v_t(\boldsymbol{x}), \alpha_t(\boldsymbol{x}))\, dx\, dt \tag{7}$$

Mean-Field Games

Problem: The mass-conservation constraint, requiring computing $\alpha_t v_t$ makes the problem non-convex.

Solution: We can reparameterize the problem using Mean-Field Games:

1. MFGs are ∞-agent games with each agent in spatial domain trying to minimize individual cost. We assume a density function at time t.

$$\min_{\alpha_t, v_t} T(\alpha_0, \alpha_1) + \int_0^1 \int_{\mathbb{R}^d} L(\boldsymbol{x}, v_t(\boldsymbol{x}), \alpha_t(\boldsymbol{x}))\, dx\, dt \tag{7}$$

2. We define agent trajectories $F : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ satisfying:

$$\begin{cases} \partial_t F(\boldsymbol{x}, t) = v_t(F(\boldsymbol{x}, t)) & \forall \boldsymbol{x} \in \mathbb{R}^d,\ t \in [0,1] \\ F(\boldsymbol{x}, 0) = \boldsymbol{x} \end{cases} \tag{8}$$

Mean-Field Games

Problem: The mass-conservation constraint, requiring computing $\alpha_t v_t$ makes the problem non-convex.

Solution: We can reparameterize the problem using Mean-Field Games:

1. MFGs are ∞-agent games with each agent in spatial domain trying to minimize individual cost. We assume a density function at time t.

$$\min_{\alpha_t, v_t} T(\alpha_0, \alpha_1) + \int_0^1 \int_{\mathbb{R}^d} L(\boldsymbol{x}, v_t(\boldsymbol{x}), \alpha_t(\boldsymbol{x}))\, dx\, dt \tag{7}$$

2. We define agent trajectories $F : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$ satisfying:

$$\begin{cases} \partial_t F(\boldsymbol{x}, t) = v_t(F(\boldsymbol{x}, t)) & \forall \boldsymbol{x} \in \mathbb{R}^d,\ t \in [0, 1] \\ F(\boldsymbol{x}, 0) = \boldsymbol{x} \end{cases} \tag{8}$$

3. Which under the reparameterization is *convex*:

$$\min_{\alpha_t, v_t} \int_0^1 \int_{\mathbb{R}^d} \|v_t(x)\|^2\, d\alpha_t(x)\, dt = \min_F \int_0^1 \int_{\mathbb{R}^d} \|\partial_t F(\boldsymbol{x}, t)\|^2 \alpha_0(\boldsymbol{x})\, dx\, dt \tag{9}$$

Mean-Field Games

Problem: The mass-conservation constraint, requiring computing $\alpha_t v_t$ makes the problem non-convex.

Solution: We can reparameterize the problem using Mean-Field Games:

1. MFGs are ∞-agent games with each agent in spatial domain trying to minimize individual cost. We assume a density function at time t.

$$\min_{\alpha_t, v_t} T(\alpha_0, \alpha_1) + \int_0^1 \int_{\mathbb{R}^d} L(\boldsymbol{x}, v_t(\boldsymbol{x}), \alpha_t(\boldsymbol{x}))\, dx\, dt \tag{7}$$

2. We define agent trajectories $F : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ satisfying:

$$\begin{cases} \partial_t F(\boldsymbol{x}, t) = v_t(F(\boldsymbol{x}, t)) & \forall \boldsymbol{x} \in \mathbb{R}^d,\ t \in [0,1] \\ F(\boldsymbol{x}, 0) = \boldsymbol{x} \end{cases} \tag{8}$$

3. Which under the reparameterization is *convex*:

$$\min_{\alpha_t, v_t} \int_0^1 \int_{\mathbb{R}^d} \|v_t(x)\|^2\, d\alpha_t(x)\, dt = \min_F \int_0^1 \int_{\mathbb{R}^d} \|\partial_t F(\boldsymbol{x}, t)\|^2 \alpha_0(\boldsymbol{x})\, dx\, dt \tag{9}$$

Next, we need to find a way to learn F. Let's talk normalizing flows.

Outline

Generative Modelling Framework

GAN: minimax the classification error loss.

VAE: maximize ELBO.

Flow-based generative models: minimize the negative log-likelihood

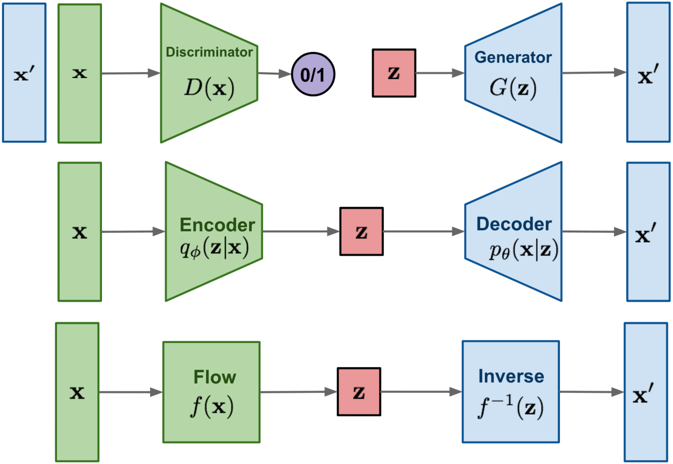


Image Credit: `https://lilianweng.github.io/posts/2018-10-13-flow-models/`

Change of Variables

Since mass vectors are sampled from Σ, α_t are *probabibility measures*.

Change of Variables

Since mass vectors are sampled from Σ, α_t are *probabibility measures*.

Setting $\boldsymbol{y} = T(\boldsymbol{x})$, we can apply the change of variables formula:

$$\alpha_1(\boldsymbol{y}) = \alpha_0(T_\sharp^{-1}\alpha_1(\boldsymbol{y}))|\det \nabla_{\boldsymbol{y}} T_\sharp^{-1}\alpha_1(\boldsymbol{y})| = \frac{\alpha_0(T_\sharp^{-1}\alpha_1(\boldsymbol{y}))}{|\det \nabla_{\boldsymbol{x}} T_\sharp \circ T_\sharp^{-1}\alpha_1(\boldsymbol{y})|} \quad (10)$$

To compute the PDF of α_1.

Change of Variables

Since mass vectors are sampled from Σ, α_t are *probabibility measures*.

Setting $\boldsymbol{y} = T(\boldsymbol{x})$, we can apply the change of variables formula:

$$\alpha_1(\boldsymbol{y}) = \alpha_0(T_\sharp^{-1}\alpha_1(\boldsymbol{y}))|\det \nabla_{\boldsymbol{y}} T_\sharp^{-1}\alpha_1(\boldsymbol{y})| = \frac{\alpha_0(T_\sharp^{-1}\alpha_1(\boldsymbol{y}))}{|\det \nabla_{\boldsymbol{x}} T_\sharp \circ T_\sharp^{-1}\alpha_1(\boldsymbol{y})|} \quad (10)$$

To compute the PDF of α_1.

We still need a known α_0, so we use the multivariate normal $\mathcal{N}(0, \boldsymbol{I})$:

$$\alpha_0(x) = \left(\frac{1}{2\pi}\right)^{d/2} \exp\left(-\frac{1}{2}\|\boldsymbol{x}\|^2\right) \quad (11)$$

Change of Variables

Since mass vectors are sampled from Σ, α_t are *probabibility measures*.

Setting $\boldsymbol{y} = T(\boldsymbol{x})$, we can apply the change of variables formula:

$$\alpha_1(\boldsymbol{y}) = \alpha_0(T_\sharp^{-1}\alpha_1(\boldsymbol{y}))|\det \nabla_{\boldsymbol{y}} T_\sharp^{-1}\alpha_1(\boldsymbol{y})| = \frac{\alpha_0(T_\sharp^{-1}\alpha_1(\boldsymbol{y}))}{|\det \nabla_{\boldsymbol{x}} T_\sharp \circ T_\sharp^{-1}\alpha_1(\boldsymbol{y})|} \quad (10)$$

To compute the PDF of α_1.

We still need a known α_0, so we use the multivariate normal $\mathcal{N}(0, \boldsymbol{I})$:

$$\alpha_0(x) = \left(\frac{1}{2\pi}\right)^{d/2} \exp\left(-\frac{1}{2}\|\boldsymbol{x}\|^2\right) \quad (11)$$

Our goal is to learn the *inverse direction* – a function from target measure α_1 to source measure α_0. Next, we can discuss approches to model T.

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can learn the normalizing direction.

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can <u>learn the normalizing direction</u>.
2. Must be expressive enough to model the target distribution.

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can learn the normalizing direction.
2. Must be expressive enough to model the target distribution.
3. Must be efficient to compute.

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can <u>learn the normalizing direction</u>.
2. Must be expressive enough to model the target distribution.
3. Must be efficient to compute.

The class of functions that satisfies these properties are **diffeomorphisms**.

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can learn the normalizing direction.
2. Must be expressive enough to model the target distribution.
3. Must be efficient to compute.

The class of functions that satisfies these properties are **diffeomorphisms**.

Diffeomorphisms are arbitrarily composable. Let $y_\ell := f_\ell$ be diffemorphisms with inverses g_ℓ for $\ell \in \{1, \dots, L\}$:

$$F := f_L \circ f_{L-1} \circ \cdots \circ f_2 \circ f_1 \tag{12}$$

$$G := g_1 \circ g_2 \circ \cdots \circ g_{L-1} \circ g_L \tag{13}$$

Then we have $F = G^{-1}$.

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can learn the normalizing direction.
2. Must be expressive enough to model the target distribution.
3. Must be efficient to compute.

The class of functions that satisfies these properties are **diffeomorphisms**.

Diffeomorphisms are arbitrarily composable. Let $y_\ell := f_\ell$ be diffemorphisms with inverses g_ℓ for $\ell \in \{1, \ldots, L\}$:

$$F := f_L \circ f_{L-1} \circ \cdots \circ f_2 \circ f_1 \tag{12}$$

$$G := g_1 \circ g_2 \circ \cdots \circ g_{L-1} \circ g_L \tag{13}$$

Then we have $F = G^{-1}$. Additionally, we can also compose determinants:

$$\det \nabla_{\boldsymbol{y}} F = \prod_{\ell=1}^{L} \det \nabla_{\boldsymbol{y}_\ell} f_\ell \tag{14}$$

Invertible Functions

We need three major properties from our choice of model:

1. Must be invertible, so that we can learn the normalizing direction.
2. Must be expressive enough to model the target distribution.
3. Must be efficient to compute.

The class of functions that satisfies these properties are **diffeomorphisms**.

Diffeomorphisms are arbitrarily composable. Let $y_\ell := f_\ell$ be diffemorphisms with inverses g_ℓ for $\ell \in \{1, \ldots, L\}$:

$$F := f_L \circ f_{L-1} \circ \cdots \circ f_2 \circ f_1 \tag{12}$$

$$G := g_1 \circ g_2 \circ \cdots \circ g_{L-1} \circ g_L \tag{13}$$

Then we have $F = G^{-1}$. Additionally, we can also compose determinants:

$$\det \nabla_{\boldsymbol{y}} F = \prod_{\ell=1}^{L} \det \nabla_{\boldsymbol{y}_\ell} f_\ell \tag{14}$$

This is huge for satisfying property 2.

Optimization by Maximum Likelihood

Learning Normalizing Flows allows us to directly maximize log-likelihood:

$$\min_{\theta} D_{\mathrm{KL}}(\alpha_1 || T_{\sharp}\alpha_0) = -\mathbb{E}_{\boldsymbol{x} \sim \alpha_1}[\log \alpha_0(G(\boldsymbol{x})) + \log |\det \nabla_{\boldsymbol{y}} G|] + C \quad (15)$$

With some constant C, which for purpose of optimization we can ignore.

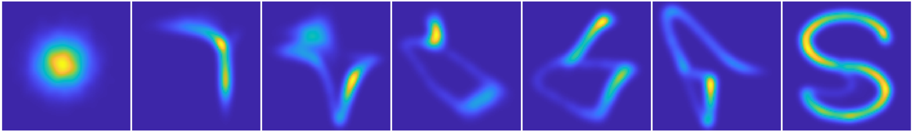
Optimization by Maximum Likelihood

Learning Normalizing Flows allows us to directly maximize log-likelihood:

$$\min_\theta D_{\mathrm{KL}}(\alpha_1 || T_\sharp \alpha_0) = -\mathbb{E}_{\boldsymbol{x}\sim\alpha_1}[\log \alpha_0(G(\boldsymbol{x})) + \log|\det \nabla_{\boldsymbol{y}} G|] + C \quad (15)$$

With some constant C, which for purpose of optimization we can ignore.

Caveat: Models trained using D_{KL} are volatile to initialization and interpolate poorly:



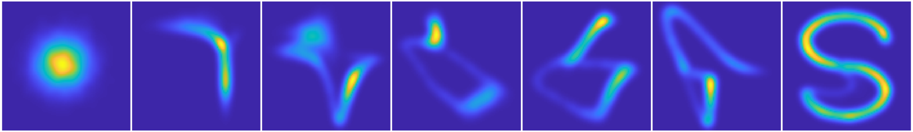
Optimization by Maximum Likelihood

Learning Normalizing Flows allows us to directly maximize log-likelihood:

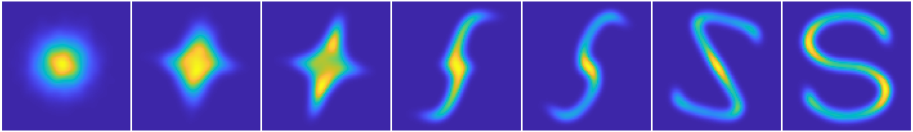
$$\min_{\theta} D_{\mathrm{KL}}(\alpha_1 || T_{\sharp}\alpha_0) = -\mathbb{E}_{\boldsymbol{x}\sim\alpha_1}[\log \alpha_0(G(\boldsymbol{x})) + \log|\det \nabla_{\boldsymbol{y}} G|] + C \quad (15)$$

With some constant C, which for purpose of optimization we can ignore.

Caveat: Models trained using D_{KL} are volatile to initialization and interpolate poorly:

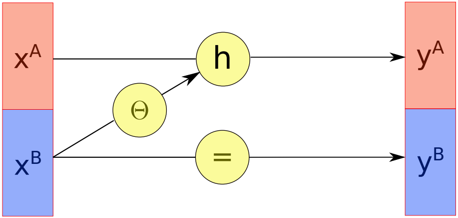


Which we can fix by regularization to transport cost:



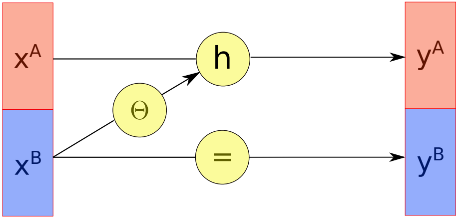
Coupling Flows

One clever diffeomorphism is a **coupling flow**:



Coupling Flows

One clever diffeomorphism is a **coupling flow**:



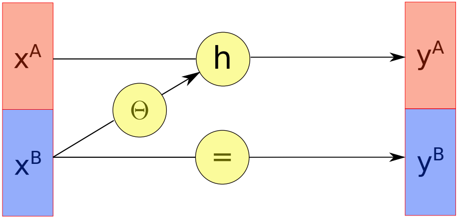
With random permutation Π, we define forward flow $(\alpha_0 \to \alpha_1)$ as:

$$\boldsymbol{x}' := \Pi(\boldsymbol{x}), \qquad \mathbf{w}, \mathbf{b} := h_\Theta(\boldsymbol{x}'_{D/2+1:D}) \tag{16}$$

$$f_k^{(\Theta_k)}(\boldsymbol{x}) := \mathrm{Concat}([\boldsymbol{x}'_{1:D/2} \odot \exp(\mathbf{w}) + \mathbf{b}, \boldsymbol{x}'_{D/2+1:D}]) \tag{17}$$

Coupling Flows

One clever diffeomorphism is a **coupling flow**:



With random permutation Π, we define forward flow $(\alpha_0 \to \alpha_1)$ as:

$$\boldsymbol{x}' := \Pi(\boldsymbol{x}), \qquad \mathbf{w}, \mathbf{b} := h_\Theta(\boldsymbol{x}'_{D/2+1:D}) \tag{16}$$

$$f_k^{(\Theta_k)}(\boldsymbol{x}) := \mathrm{Concat}([\boldsymbol{x}'_{1:D/2} \odot \exp(\mathbf{w}) + \mathbf{b}, \boldsymbol{x}'_{D/2+1:D}]) \tag{17}$$

Subsequently inverse flow $(\alpha_1 \to \alpha_0)$ is defined as follows:

$$\mathbf{w}, \mathbf{b} := h_\Theta(\boldsymbol{y}_{D/2+1:D}) \tag{18}$$

$$f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) := \Pi^{-1}\left(\mathrm{Concat}([(\boldsymbol{y}_{1:D/2} - \mathbf{b}) \oslash \exp(\mathbf{w}), \boldsymbol{y}_{D/2+1:D}])\right) \tag{19}$$

Computing the $\log|\det(\cdot)|$

Best part, the Jacobian of $f_k^{(\Theta_k)^{-1}}$ has the the following block form:

$$\nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \begin{bmatrix} I_{D/2\times D/2} & \mathbf{0}_{D/2\times D/2} \\ \frac{\partial f_{k,D/2+1:D}^{(\Theta_k)^{-1}}}{\partial \boldsymbol{y}_{1:D/2}} & \operatorname{diag}\left(\exp\left(-\mathbf{w}\right)\right) \end{bmatrix}$$

Computing the $\log|\det(\cdot)|$

Best part, the Jacobian of $f_k^{(\Theta_k)^{-1}}$ has the the following block form:

$$\nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \begin{bmatrix} I_{D/2\times D/2} & \mathbf{0}_{D/2\times D/2} \\ \frac{\partial f_{k,D/2+1:D}^{(\Theta_k)^{-1}}}{\partial \boldsymbol{y}_{1:D/2}} & \operatorname{diag}\left(\exp\left(-\mathbf{w}\right)\right) \end{bmatrix} \tag{20}$$

Which enables us to compute its determinant in linear time:

$$\det \nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \exp\left(\sum -\mathbf{w}_k\right) \tag{21}$$

Computing the $\log|\det(\cdot)|$

Best part, the Jacobian of $f_k^{(\Theta_k)^{-1}}$ has the the following block form:

$$\nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \begin{bmatrix} I_{D/2 \times D/2} & \mathbf{0}_{D/2 \times D/2} \\ \frac{\partial f_{k,D/2+1:D}^{(\Theta_k)^{-1}}}{\partial \boldsymbol{y}_{1:D/2}} & \operatorname{diag}\left(\exp\left(-\mathbf{w}\right)\right) \end{bmatrix} \tag{20}$$

Which enables us to compute its determinant in linear time:

$$\det \nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \exp\left(\sum -\mathbf{w}_k\right) \tag{21}$$

Which we can further use to compute composed $\log\det \nabla F^{-1}$:

$$\log\det \nabla F^{-1}(\boldsymbol{y}) = \sum_{\ell=0}^{L-1} \sum_{i=1}^{D/2} -\mathbf{w}_i^{(\ell)} \tag{22}$$

Computing the $\log|\det(\cdot)|$

Best part, the Jacobian of $f_k^{(\Theta_k)^{-1}}$ has the the following block form:

$$\nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \begin{bmatrix} I_{D/2\times D/2} & \mathbf{0}_{D/2\times D/2} \\ \frac{\partial f_{k,D/2+1:D}^{(\Theta_k)^{-1}}}{\partial \boldsymbol{y}_{1:D/2}} & \operatorname{diag}\left(\exp\left(-\mathbf{w}\right)\right) \end{bmatrix} \tag{20}$$

Which enables us to compute its determinant in linear time:

$$\det \nabla f_k^{(\Theta_k)^{-1}}(\boldsymbol{y}) = \exp\left(\sum -\mathbf{w}_k\right) \tag{21}$$

Which we can further use to compute composed $\log\det \nabla F^{-1}$:

$$\log\det \nabla F^{-1}(\boldsymbol{y}) = \sum_{\ell=0}^{L-1} \sum_{i=1}^{D/2} -\mathbf{w}_i^{(\ell)} \tag{22}$$

Implication: h_Θ can be arbitrarily complex!

Outline

1. Dynamic Optimal Transport
2. Normalizing Flows
3. Transport-Regularized Normalizing Flows

Setting the Terminal Condition

The final thing left to discuss is the terminal condition. We need F s.t:

$$F(\boldsymbol{x}, 1) = \boldsymbol{y} \sim \alpha_1 \quad \forall x \in \mathbb{R}^d \tag{23}$$

Setting the Terminal Condition

The final thing left to discuss is the terminal condition. We need F s.t:

$$F(\boldsymbol{x}, 1) = \boldsymbol{y} \sim \alpha_1 \quad \forall x \in \mathbb{R}^d \tag{23}$$

Instead of solving a constrained problem, we soft-constraint using D_{KL}:

$$\min_F \int_0^1 \int_{\mathbb{R}^d} \|\partial_t F(\boldsymbol{x}, t)\|^2 \alpha_0(\boldsymbol{x}) \; dx \; dt + \lambda D_{\mathrm{KL}}(\alpha_1 || F_\sharp \alpha_0) \tag{24}$$

D_{KL} is conventional cost; we just add transport cost, hence new formulation is called **transport-regularized normalizing flows**.

Setting the Terminal Condition

The final thing left to discuss is the terminal condition. We need F s.t:

$$F(\boldsymbol{x}, 1) = \boldsymbol{y} \sim \alpha_1 \quad \forall x \in \mathbb{R}^d \tag{23}$$

Instead of solving a constrained problem, we soft-constraint using D_{KL}:

$$\min_F \int_0^1 \int_{\mathbb{R}^d} \|\partial_t F(\boldsymbol{x}, t)\|^2 \alpha_0(\boldsymbol{x}) \; dx \; dt + \lambda D_{\mathrm{KL}}(\alpha_1 || F_\sharp \alpha_0) \tag{24}$$

D_{KL} is conventional cost; we just add transport cost, hence new formulation is called **transport-regularized normalizing flows**.

Discretizing across a composition of L flows, we have:

$$\min_F L \cdot \mathbb{E}_{\boldsymbol{x} \sim \alpha_0} \left[\sum_{\ell=0}^{L-1} \|F_{\ell+1}(\boldsymbol{x}) - F_\ell(\boldsymbol{x})\|_2^2 \right] + \lambda D_{\mathrm{KL}}(\alpha_1 || F_\sharp \alpha_0) \tag{25}$$

Using this, we can train transport-regularized normalizing flows.

Implementation Details

If you can view this screen, I am making a mistake.

Thank you!

Have an awesome rest of your day!

Slides: `https://jinen.setpal.net/slides/nf.pdf`