Riemannian Flow Matching for InSAR

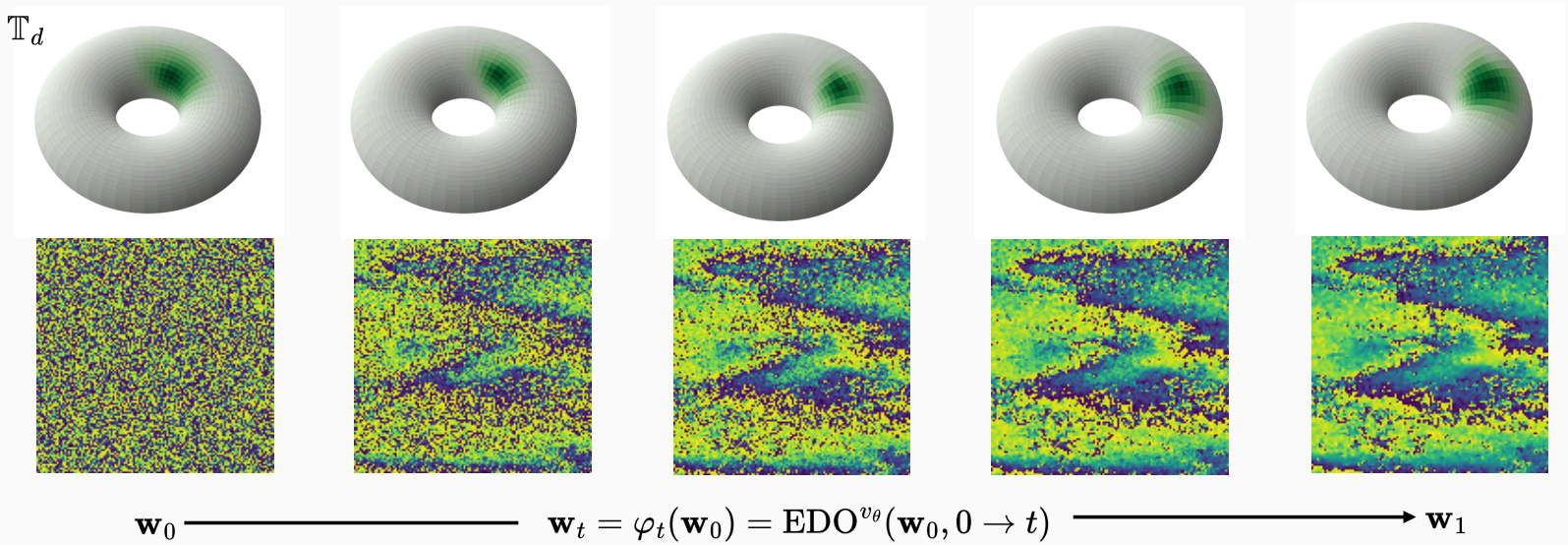
G. Le Bellier, D. El Hajjar, A. Breloy, N. Audebert

Séminaire TéSA, 2025

le cnam

In this talk

- Interferometric SAR (**InSAR**)
- **Diffusion Models** and **flow matching**
- **Riemannian Flow matching** for **InSAR**
  - Interferometric **phase denoising**
  - **Sampling realistic interferograms**



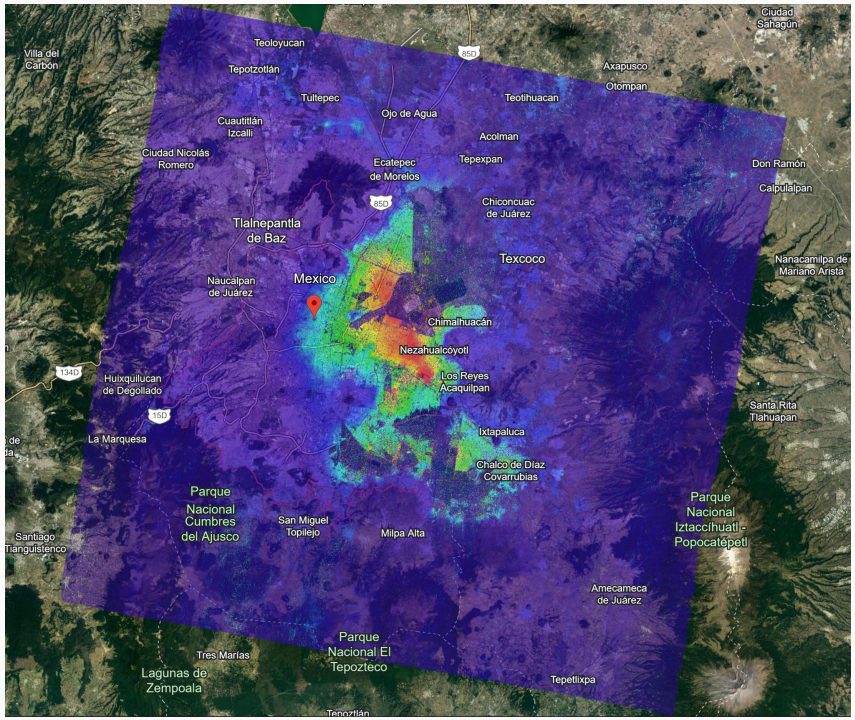
Outline

Interferometric SAR

Diffusion models and flow matching

Riemannian flow matching for InSAR

Motivations



Monitoring ground subsidence

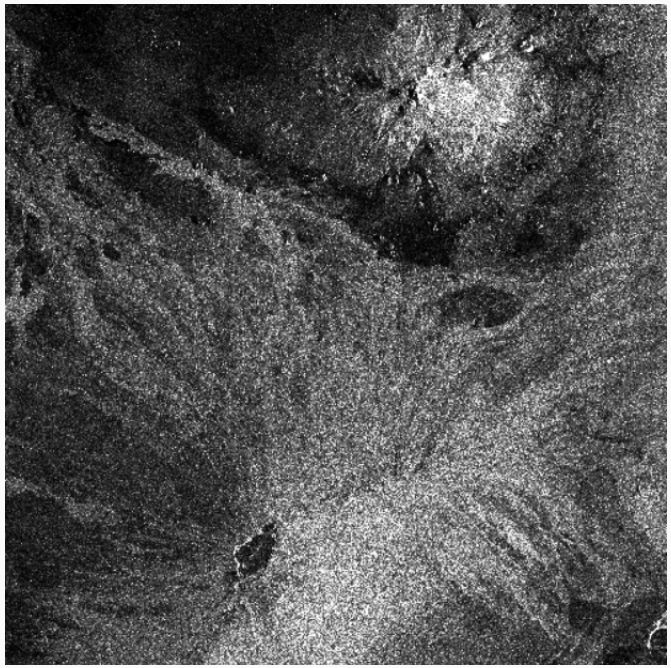
- Large scale and long term
- Ground-based methods (GPS, topography)
  - Costly
  - Limited spatial coverage

Interferometry with SAR (InSAR)

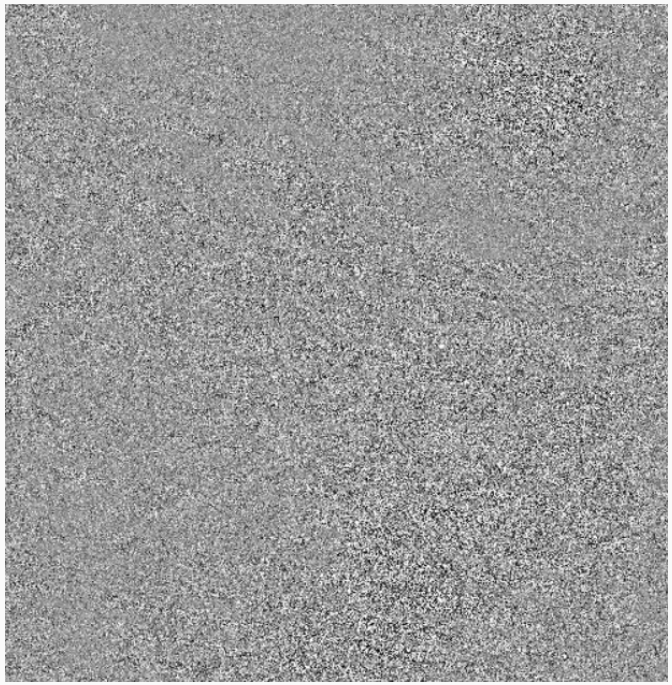
- Covers large areas
- Systematic acquisitions
- Unaffected by weather
- Sub-centimeter accuracy

Synthetic Aperture Radar (SAR)

SAR produces **complex valued images**



modulus



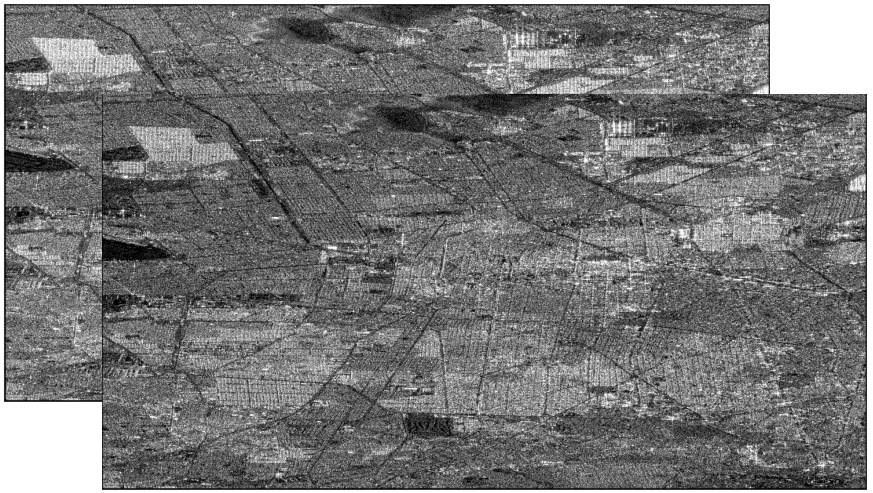
phase

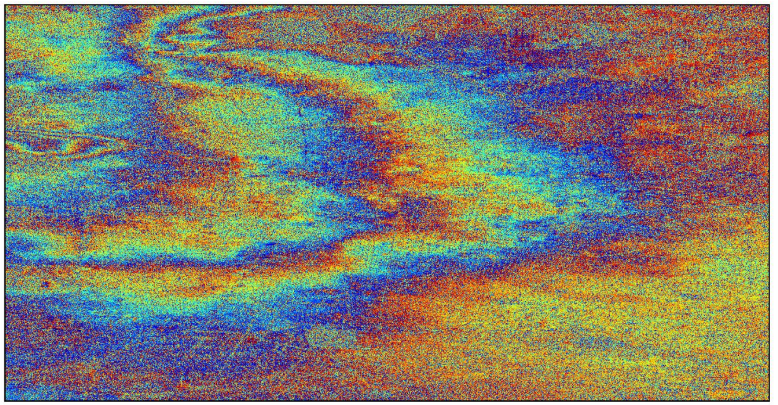
Interferometric SAR

Two images with **same position** at **different time**

Phase difference contains information about **surface displacement**

$$\Delta\Phi = \Delta\Phi_{displ} + \underbrace{\Delta\Phi_{topo} + \Delta\Phi_{orb} + \Delta\Phi_{atm}}_{\text{removed by pre-processing}} + \Delta\Phi_{noise}$$

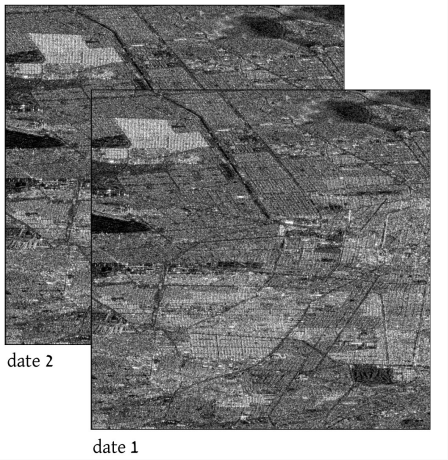


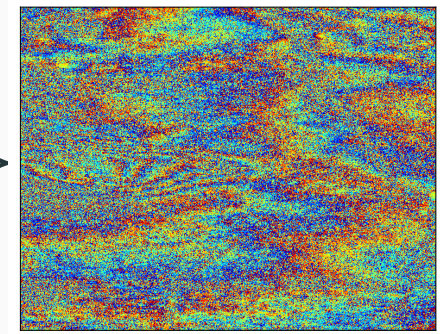


we see fringes because phases are $\mathrm{mod}\ 2\pi$

Multi-temporal InSAR

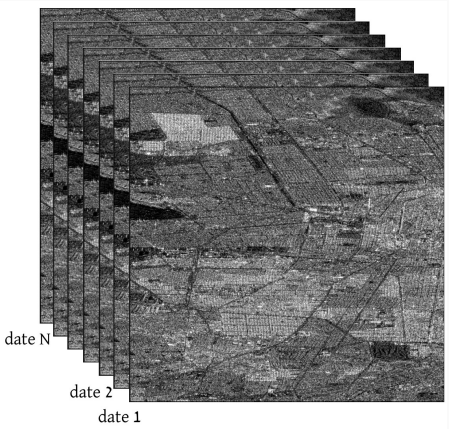
2-Pass InSAR

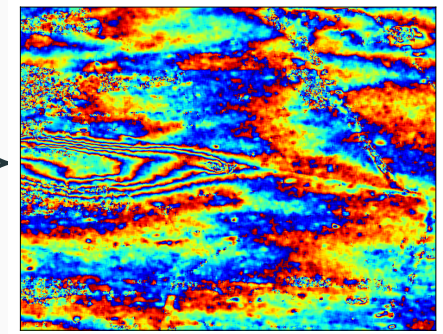




- Only 2 SAR images
- Loss of coherence over time
- Noisy phase difference

Multi-temporal InSAR



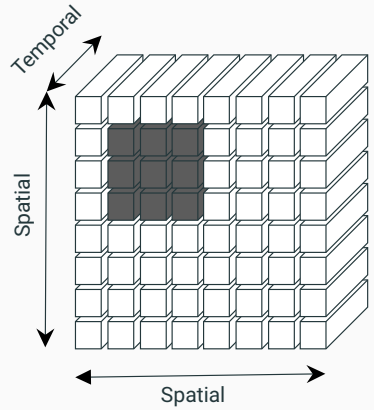


- SAR image time series
- Leverages redundancy
- Requires models & optimization

More on that on Thursday at S5-T2 by Elena Grosso :D

Multi-temporal InSAR covariance modelling

InSAR datacube



Multidimensional pixels (p images)

$$\mathbf{x} = [x^1, \cdots, x^p] \in \mathbb{C}^p$$

Spatial window (n pixels)

$$\{\mathbf{x}_i\}_{i=1}^n \in (\mathbb{C}^p)^n$$

DS-InSAR assumptions

- Zero-mean

$$\mathbb{E}[x^\ell] = 0, \forall \ell \in [\![1, p]\!]$$

- Structured correlations w.r.t. time

$$\mathbb{E}[x^q (x^\ell)^*] = \varphi_{q\ell} \underbrace{e^{j\,(\theta_q - \theta_\ell)}}_{\text{phase diff.}})$$

$\Rightarrow$ not any local covariance matrix!

Multi-InSAR covariance matrix model

Structured **modulus-argument decomposition**

$$\mathbb{E}\left[\mathbf{x}\mathbf{x}^H\right] = \mathbf{\Sigma} = \mathrm{mod}(\mathbf{\Sigma}) \odot \arg(\mathbf{\Sigma}) \triangleq \mathbf{\Psi} \odot (\mathbf{w}\mathbf{w}^H)$$

with

- $\mathbf{\Psi}$: coherence and variance
- $\mathbf{w} = [e^{j\theta_1}, ..., e^{j\theta_P}]^\top$: phase at each time

This structure implies the **phase-closure** property :

$$\arg(\mathbf{\Sigma}_{q\ell}) + \arg(\mathbf{\Sigma}_{\ell j}) + \arg(\mathbf{\Sigma}_{jq}) = 0$$

Interferometric phase linking (IPL)

Interferometric phase linking problem

From pixel patch $\{\mathbf{x}_i\}_{i=1}^n$, estimate $\mathbf{w}$

Two main **IPL** approaches

- **Maximum likelihood estimation**

  assume $\mathbf{x} \sim \mathcal{CN}(\mathbf{0}, \mathbf{\Psi} \odot \mathbf{w}\mathbf{w}^H)$ and roll

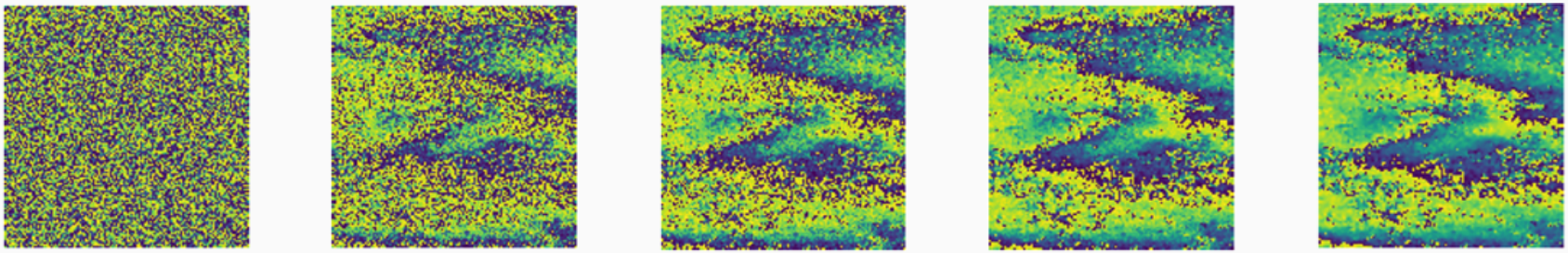
- **Covariance matrix fitting**

  force phase closure on plug-in estimate

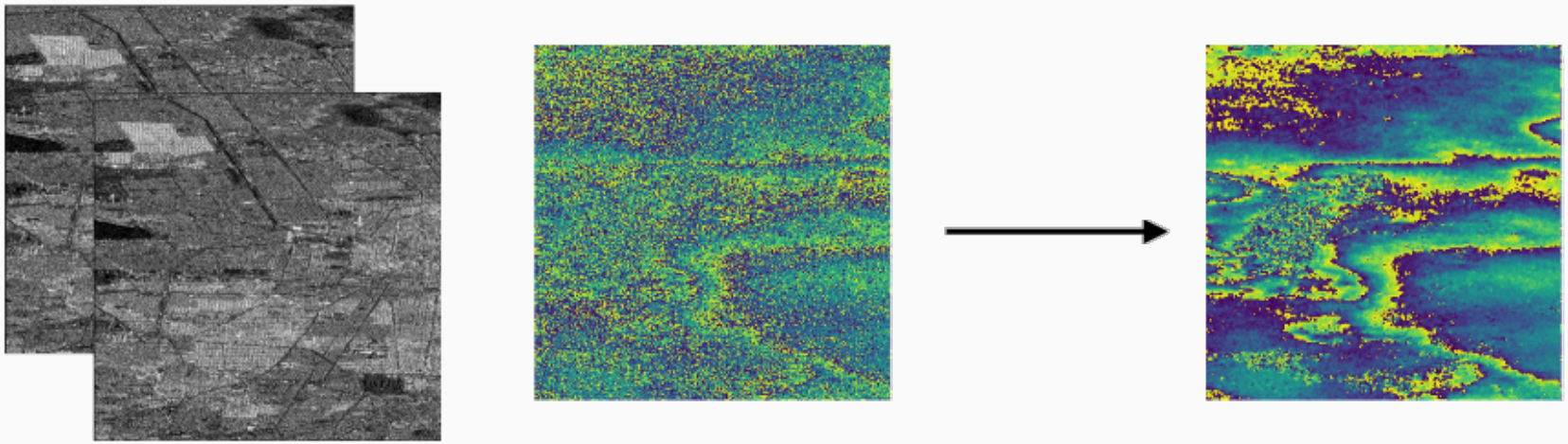
$$\min_{\mathbf{w} \in \mathbb{T}_p} \quad d^2(\hat{\mathbf{\Sigma}},\ \mathrm{mod}(\hat{\mathbf{\Sigma}}) \odot \mathbf{w}\mathbf{w}^H)$$

Generative models for InSAR : two goals

Sampling



Denoising



Outline

Interferometric SAR

Diffusion models and flow matching

Riemannian flow matching for InSAR

Generative modeling

Goal : generate new samples similar to existing ones

Probabilistic view : data was sampled from an underlying density

What we want :

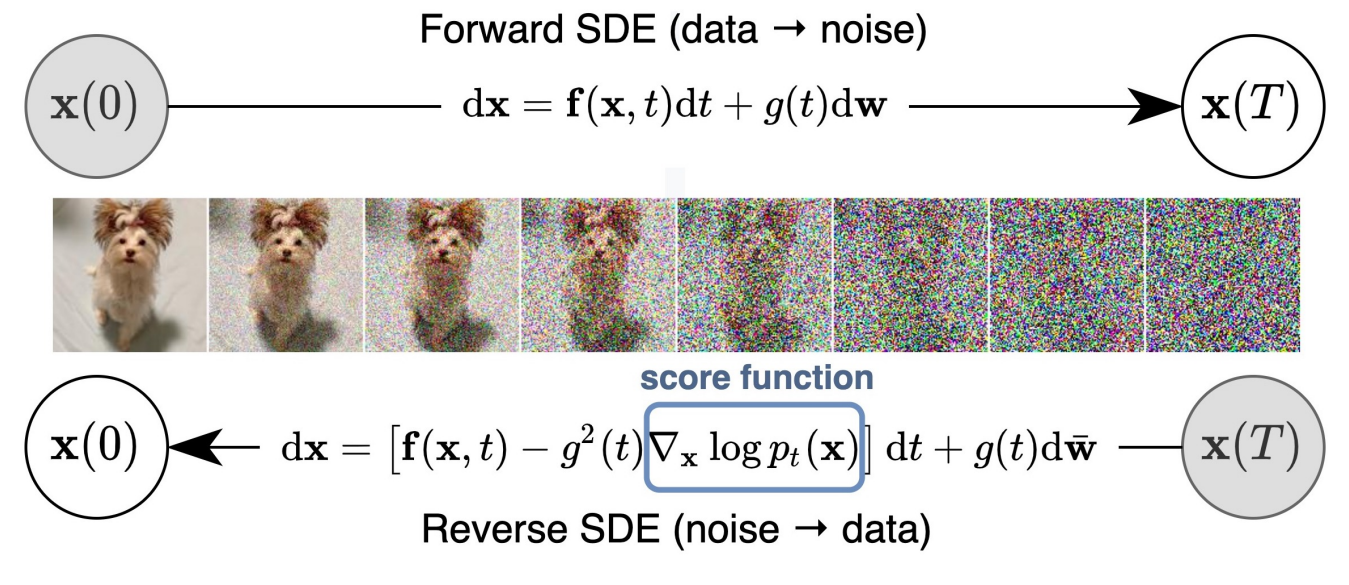
- data probability density function $p_{\text{data}}(\mathbf{x})$
- a way to sample data as $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$

What we have :

- samples $\{\mathbf{x}_i\}_{i=1}^{n}$
- problems

Diffusion models, overview

Diffusion models : “long sequence of small denoisers”



The score matching view of diffusion

- **Step 1** : Learning models with **score matching**
- **Step 2** : Sampling with **Langevin dynamics**

next slides mostly inspired (if not taken) from Stanford-CS236

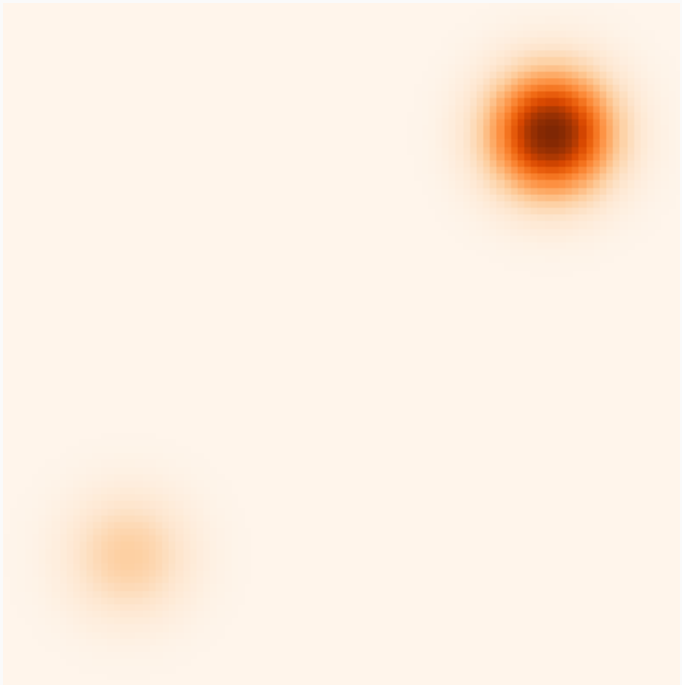
Score function

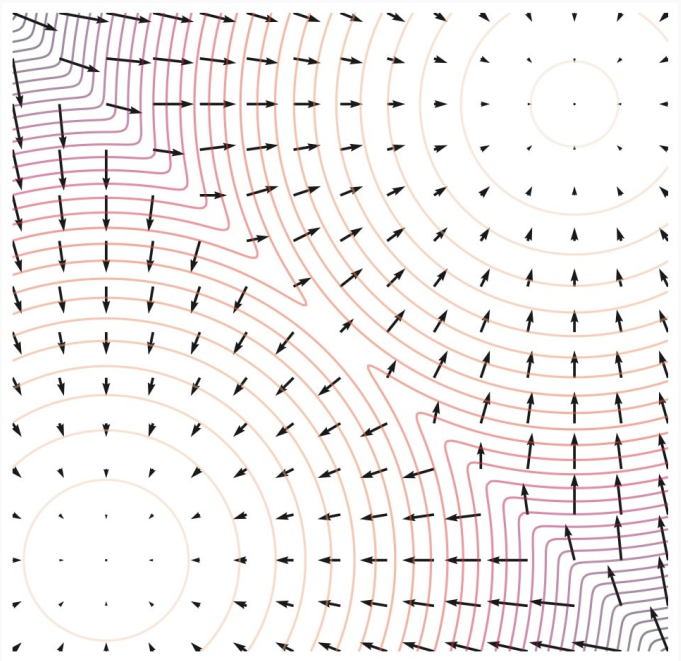
Definition

The score $\boldsymbol{s} : \mathbb{R}^d \to \mathbb{R}^d$ of a distribution p is

$$\boldsymbol{s}(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

"assigns to each x the steepest direction to increase likelihood"



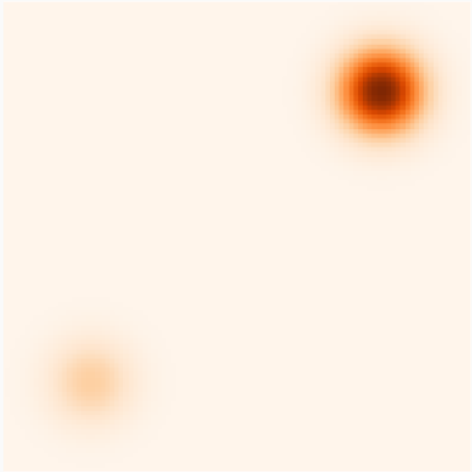


Not the "Fisher-Rao" score (we differentiate w.r.t. x here)

Score estimation

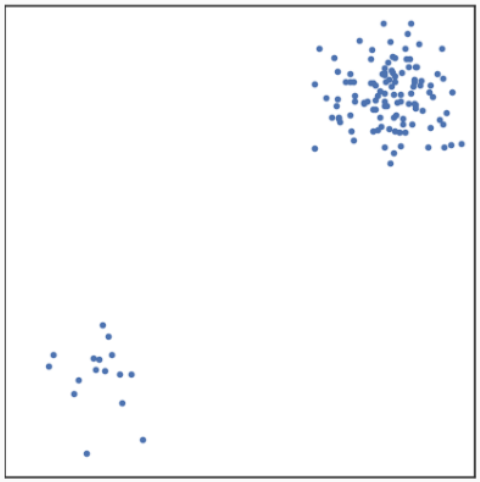
Probability density

$p_{\text{data}}(\mathbf{x})$



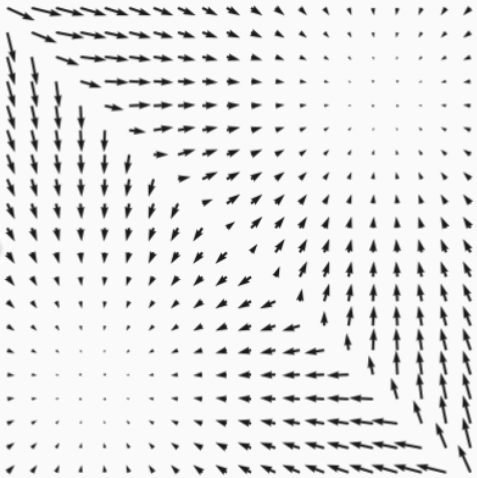
i.i.d. samples

$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$



Score function

$\boldsymbol{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$



Score matching

- Samples $\{\mathbf{x}_i\}_{i=1}^n \sim p_{\text{data}}(\mathbf{x})$
- Parametric function $\boldsymbol{s}_\theta : \mathbb{R}^d \to \mathbb{R}^d$ (neural netork)
- Find θ such that $\boldsymbol{s}_\theta(\mathbf{x}) \simeq \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Score matching with Fisher divergence

$$\begin{aligned} \mathcal{L}(\theta) &= \tfrac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\left[||\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - s_\theta(\mathbf{x})||_2^2\right] \\ &= \tfrac{1}{2}\sum_{i=1}^n ||\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - s_\theta(\mathbf{x})||_2^2 \end{aligned}$$

It depends on p_{data} :(

Score matching

Theorem (Hyvärinen 2005)

Under regularity assumptions

$$\begin{aligned}\mathcal{L}(\theta) &= \tfrac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\left[||\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - s_\theta(\mathbf{x})||_2^2\right] \\ &= \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\left[\tfrac{1}{2}||s_\theta(\mathbf{x})||_2^2 + \operatorname{tr}(\nabla_{\mathbf{x}} s_\theta(\mathbf{x}))\right] + \text{const.}\end{aligned}$$

where $\nabla_{\mathbf{x}} s_\theta(\mathbf{x})$ is the Jacobian of $s_\theta(\mathbf{x})$

- Does not depend on p_{data} :)
- Still, evaluationg $\nabla_{\mathbf{x}} s_\theta(\mathbf{x})$ is costly $\mathcal{O}(d)$ backpropagations
- Can be well approximated by **denoising score matching**

Denoising scroe matching

$\mathbf{x}$

$p_{\text{data}}(\mathbf{x})$

$q_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})$

$\tilde{\mathbf{x}}$

$q_\sigma(\tilde{\mathbf{x}})$

Add gaussian noise to data

$$\tilde{\mathbf{x}} = \mathbf{x} + \sigma\mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$$

$$q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = \mathcal{N}(\mathbf{x}, \sigma^2\mathbf{I})$$

$$q_\sigma(\tilde{\mathbf{x}}) = \int q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) p_{\text{data}}(\mathbf{x}) \mathrm{d}\mathbf{x}$$

Score matching on q_σ (Vincent 2011)

$$\begin{aligned} \tilde{\mathcal{L}}(\theta) &= \tfrac{1}{2}\mathbb{E}_{\tilde{\mathbf{x}}\sim q_\sigma}\left[||\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - \boldsymbol{s}_\theta(\tilde{\mathbf{x}})||_2^2\right] \\ &= \tfrac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}, \tilde{\mathbf{x}}\sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}\left[||\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})||_2^2\right] + \text{const.} \end{aligned}$$

Denoising score matching

If $q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = \mathcal{N}(\mathbf{x}, \sigma^2\mathbf{I})$, everything gets simple

$$\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2}$$

In conclusion

$$\tilde{\mathcal{L}}(\theta) = \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}, \tilde{\mathbf{x}}\sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})}\left[||\boldsymbol{s}_\theta(\tilde{\mathbf{x}}) - \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2}||_2^2\right]$$

The score model simply tries to **predict the added noise**!

Multi-scale denoising score matching

Score matching w.r.t. noise level

Overload model : noise level σ is an input

$$\boldsymbol{s}_\theta(\mathbf{x}, \sigma) : \mathbb{R}^d \times \mathbb{R}^+ \longrightarrow \mathbb{R}^d$$

Multi-scale loss

$$\begin{aligned}
\mathcal{L}(\theta) &= \frac{1}{L}\sum_{\ell=1}^{L} \lambda(\sigma_\ell)\mathbb{E}_{q_{\sigma_\ell}(\tilde{\mathbf{x}})}\left[||\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma_\ell}(\tilde{\mathbf{x}}) - \boldsymbol{s}_\theta(\tilde{\mathbf{x}}, \sigma_\ell)||_2^2\right] \\
&= \frac{1}{L}\sum_{\ell=1}^{L} \lambda(\sigma_\ell)\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}, \mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[||\boldsymbol{s}_\theta(\mathbf{x} + \sigma_\ell\mathbf{z}, \sigma_\ell) - \mathbf{z}/\sigma_\ell||_2^2\right]
\end{aligned}$$

At this point, **step 1** is done!

Step 2 : how to sample data from learned $\boldsymbol{s}_\theta(\mathbf{x}, \sigma)$?

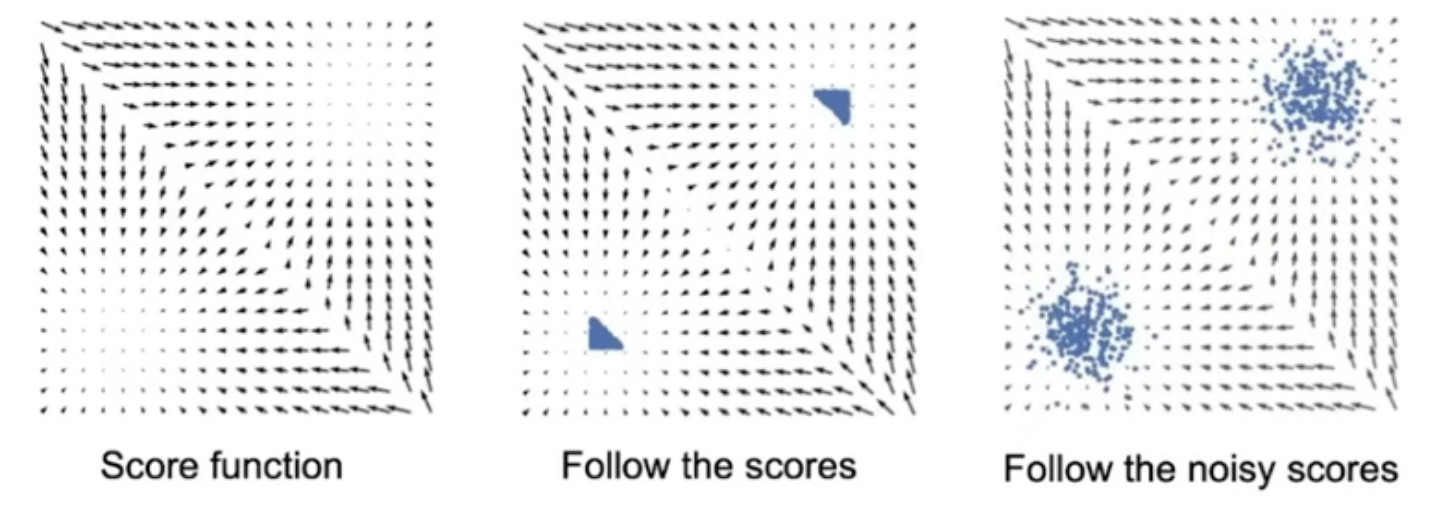
Sampling with Langevin dynamics

Langevin dynamics

- $\mathbf{x}_0 \sim \pi(\mathbf{x}) \leftarrow$ random sampling
- Follow T iterations of

$$\mathbf{x}_t \leftarrow \mathbf{x}_t + \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log p(\mathbf{x}_{t-1}) + \epsilon \boldsymbol{z}_t$$

if $\epsilon << 0$ and $T \to \infty$, then $\mathbf{x}_T \sim p$



Sampling with a multi-scale score model

Two steps :

- Learn **score** s_θ on $\{\mathbf{x}_i\}_{i=1}^n$ with **denoising score matching**
- Uste the **learned score** within **Langevin dynamics**

$$\mathbf{x}_t \leftarrow \mathbf{x}_t + \frac{\epsilon}{2} s_\theta(\mathbf{x}) + \epsilon \boldsymbol{z}_t, \quad t \in [\![1, T]\!]$$

- If $s_\theta(\mathbf{x}) \simeq \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$, then we approximate $\mathbf{x}_T \sim p_{\text{data}}$!

In practice

- We learn a score **conditioned to the scale** $s_\theta(\mathbf{x}, \sigma)$
- Dynamics is **annealed** within a loop over $\sigma \downarrow$

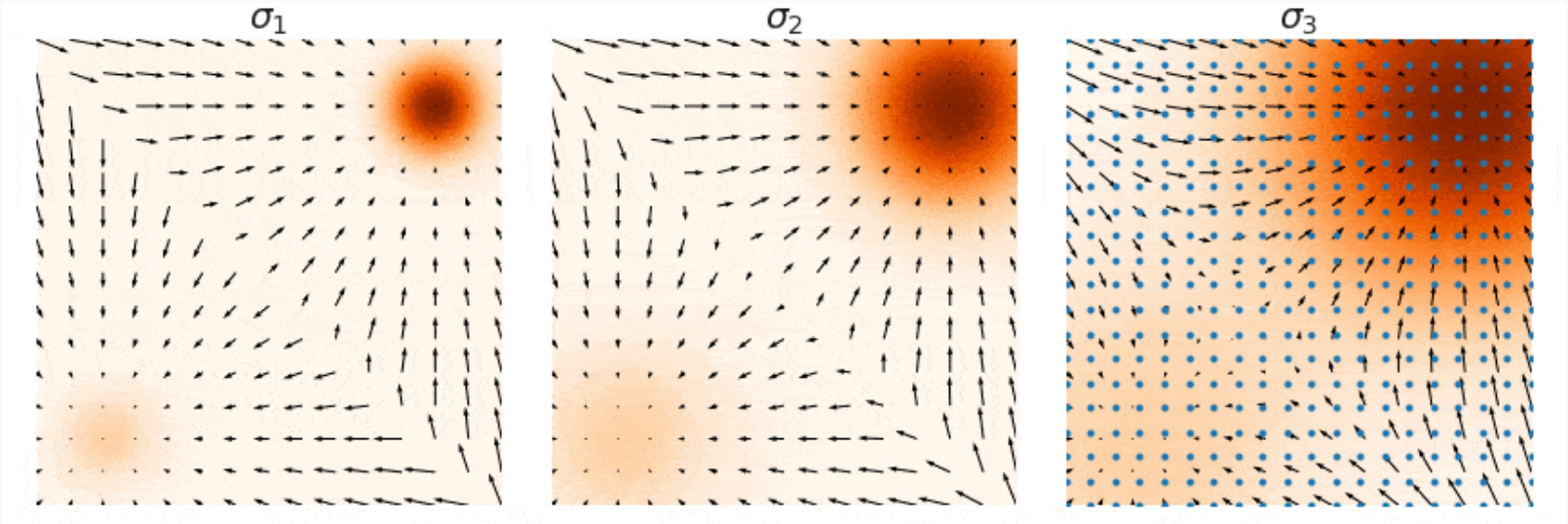
Annealed Langevin dynamics

Algorithm 1 Annealed Langevin dynamics.

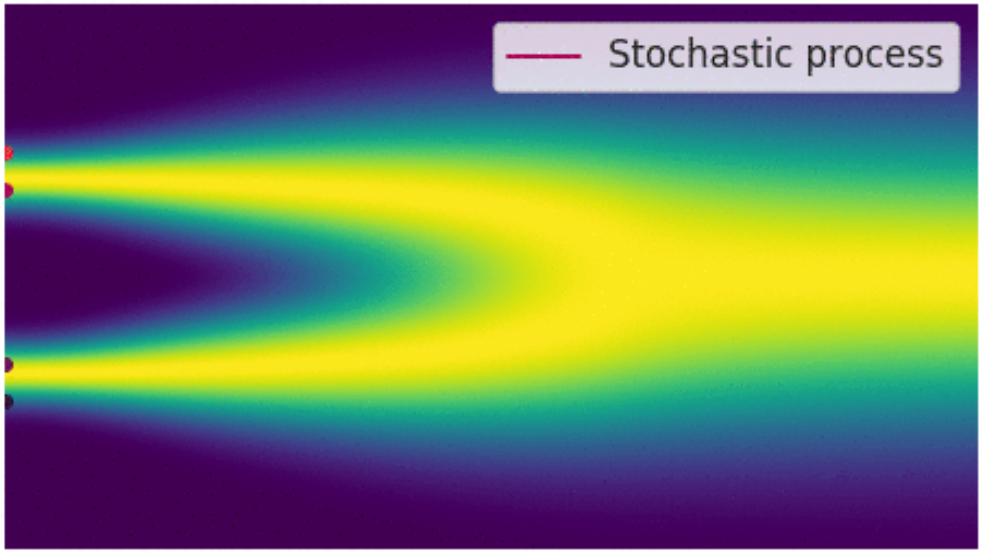
Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$.

Initialize $\tilde{\mathbf{x}}_0$
for $i \leftarrow 1$ to L **do**
  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$ ▷ α_i is the step size.
  for $t \leftarrow 1$ to T **do**
    Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$
  end for
  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
end for
return $\tilde{\mathbf{x}}_T$

Dynamique de Langevin avec recuit simulé



The “continuous” case $\mathrm{card}(\{\sigma_i\}_{i=1}^{L}) = L \to \infty$



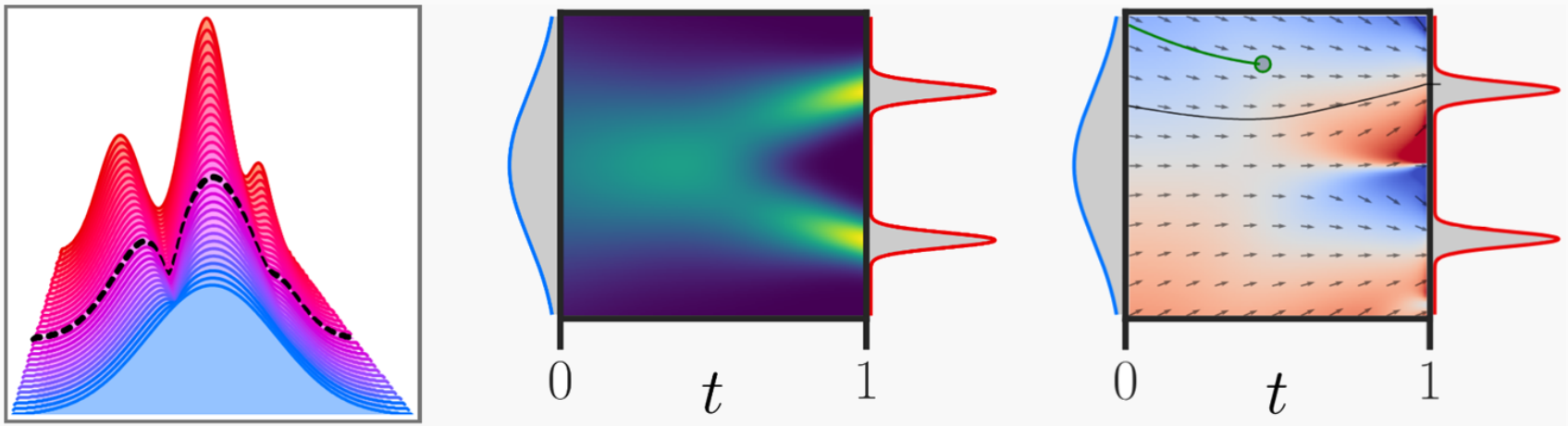
Sampling $\simeq$ inverting a **stochastic differential equation** (SDE)

$$\mathrm{d}\mathbf{x}_t = f(\mathbf{x}_t, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t$$

From there, what is **flow matching**?

- Learn **another transport** from $\pi()$ to p_{data}
- Find an **ordinary differential equation** (ODE) deterministic transformation!

Flow matching - goal



We seek a velocity field u_t such that

$$\frac{d}{dt}\varphi_t(x) = u_t(\varphi_t(x)), \quad \forall t \in [0,1] \tag{1}$$

for which $\varphi_0 = \pi$ and $\varphi_0 = p_{data}$

Flow matching - algorithm

Learn a function $v_\theta(t, x)$ with the algorithm

- Sample $z_0 \sim \mathcal{N}(0, \mathbf{I})$, $z_1 \sim p_{data}$, and $t \sim \mathcal{U}(0, 1)$
- Stochastic gradient descent on θ with loss

$$\left\| v_\theta(t, x_t) - \frac{z_1 - z_0}{1 - t} \right\|^2$$

Looks like score matching, but we predict a direction rather than noise

What does it do on average?

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{\substack{x_0, x_1 \sim p(x_0, x_1) \\ t \sim U([0,1])}} \left\| v_\theta(t, x_t) - u_t(x_t \mid x_0, x_1) \right\|^2$$

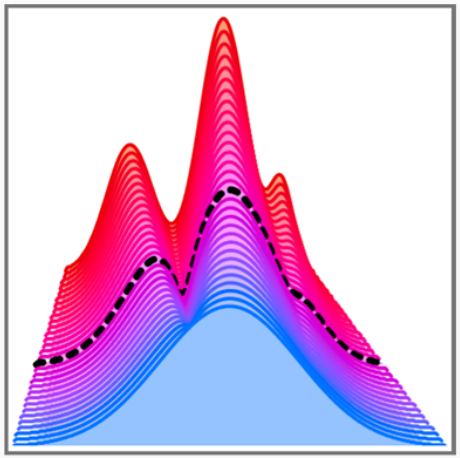
Visualize with the following [blog-post] or [playground]

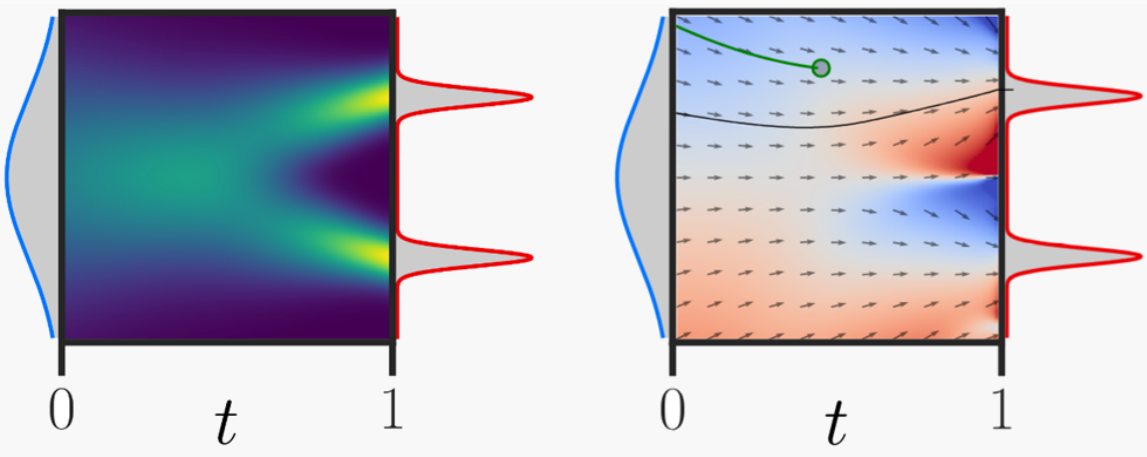
Flow matching - sampling

Sampling from the model

$$\hat{x}_1 = \varphi_{t=1}(x_0) = \text{EDO}^{v_\theta}(x_0, 0 \to 1)$$

where $\text{EDO}^{v_\theta}(\cdot, t_0 \to t_1)$ solves (1) from t_0 to t_1 with $v_\theta(t, \cdot)$ instead of u_t





Outline

Interferometric SAR

Diffusion models and flow matching

Riemannian flow matching for InSAR

Back to InSAR

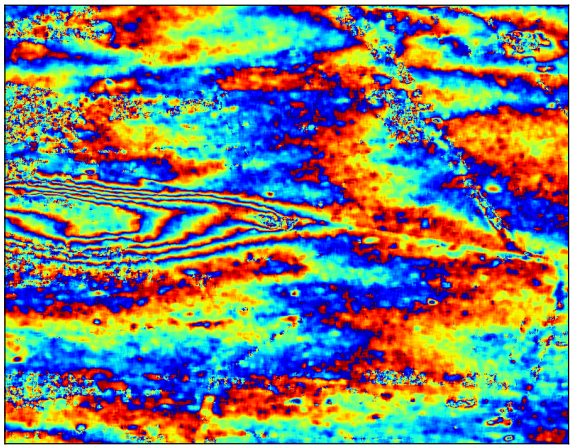
An image of **InSAR phases**

$\Leftrightarrow$ d pixels that are $\mathrm{mod} 2\pi$

$\Leftrightarrow$ d complex phases

$\Leftrightarrow$ d circles

$\Leftrightarrow$ a **d-torus**



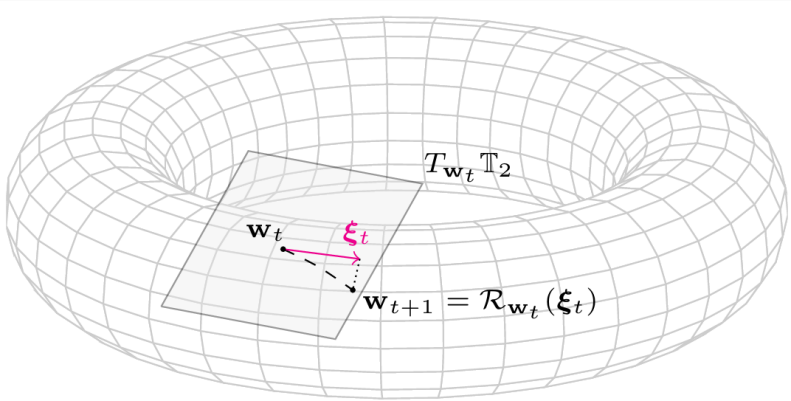
A **d-torus** with the **Euclidean metric** is a **Riemannian Manifold**

Flow matching for **distributions on Riemannian manifolds** exists!

Flow matching on Riemannian manifolds (Chen 2024)

RFM : equations are more intricate but **conceptually the same**!

- **Velocities** lie on the **tangent space**



- **Straight lines** between two points are **geodesics**

$$\mathbf{w}_t = \exp_{\mathbf{w}_1}\left(\kappa(t)\log_{\mathbf{w}_1}(\mathbf{w}_0)\right)$$

With simple operators for the torus $\begin{cases} \exp_w(u) = (w+u) \pmod{2\pi} \\ \log_{w_a}(w_b) = \text{arctan2}(\sin(w_b - w_a), \cos(w_b - w_a)) \end{cases}$

Flow matching on the torus

Learning

$$\mathcal{L}_{\text{RFM}}(\theta) = \mathbb{E}_{\substack{\mathbf{w}_0,\mathbf{w}_1 \sim p(\mathbf{w}_0,\mathbf{w}_1) \\ t \sim U([0,1])}} \left\| \mathbf{v}_\theta(t, \mathbf{w}_t) - \mathbf{u}_t(\mathbf{w}_t \mid \mathbf{w}_0, \mathbf{w}_1) \right\|_g^2$$

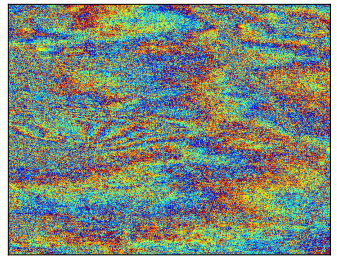
Sampling (inference)

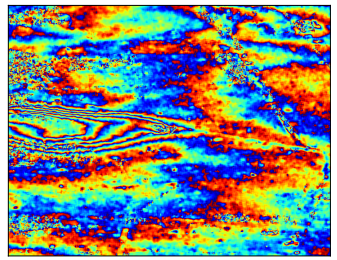
$$\hat{\mathbf{w}}_1 = \varphi_{t=1}(\mathbf{w}_0) = \text{EDO}_{\mathbb{T}_d}^{v_\theta}(\mathbf{w}_0, 0 \to 1) \tag{2}$$

Application to InSAR : some results

Dataset : Mexico city seen from Sentinel-1

- 2 cubes of 40 SAR image phases
  - Raw (2p-InSAR) $\sim p_{data}$
  - Clean (MT-InSAR) $\sim p_{data}^{clean}$
- Every 12 day between 14/08/19 and 6/12/20
- $\simeq$ 3.6k × 16k pixels $\longrightarrow$ many patches
- 40(39)/2 = 780 possible interferograms

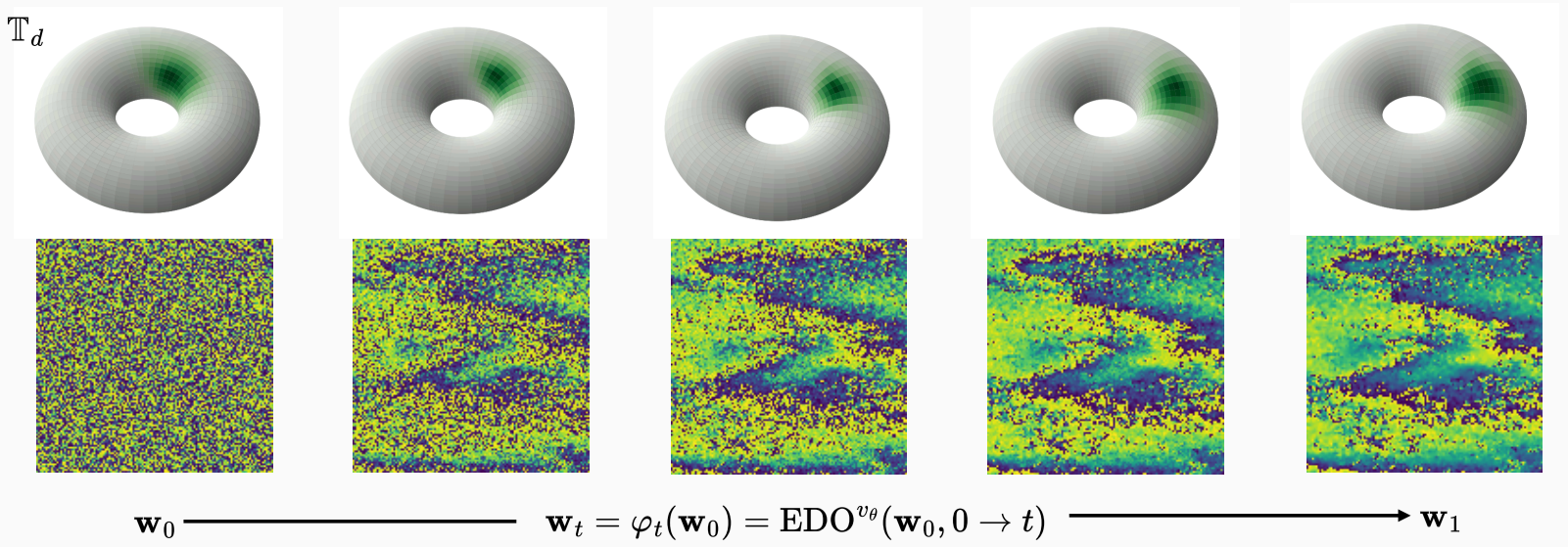




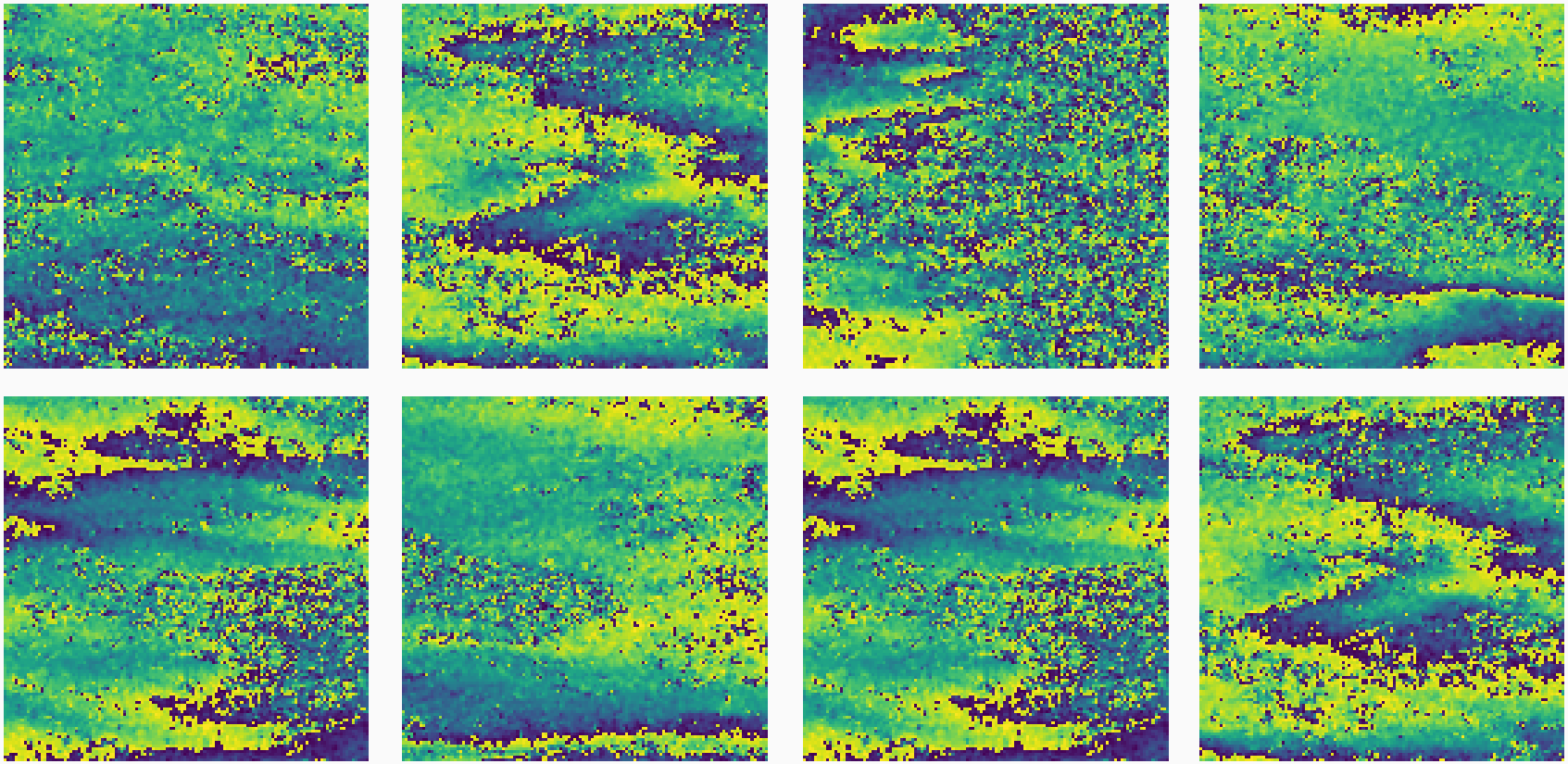
Flow matching tasks

- **Generation** : map $\mathcal{U}(\mathbb{T}_d) \longrightarrow p_{data}$
- **Denoising** : map $p_{data} \longrightarrow p_{data}^{clean}$

Preliminary results - generation (1/2)

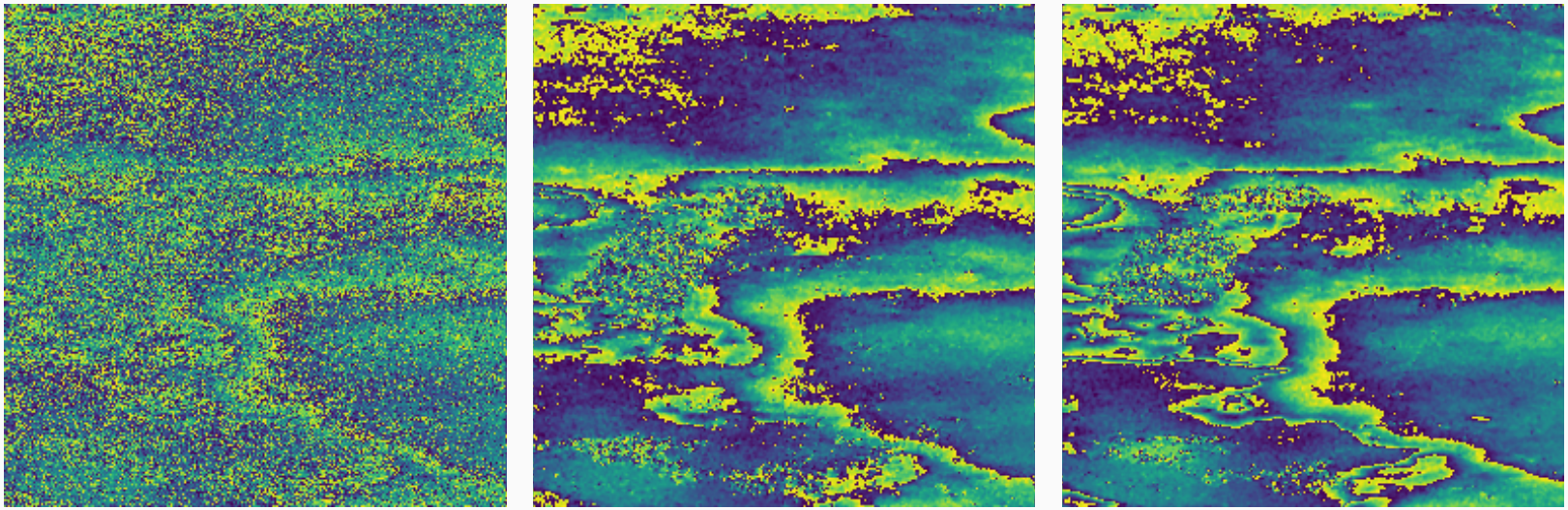


Preliminary results - generation (2/2)



Preliminary results - denoising (1/2)

Raw $\mathbf{w}_0$ | Denoised $\varphi_1(\mathbf{w}_0)$ | Clean $\mathbf{w}_1$



Preliminary results - denoising (2/2)

Raw $\mathbf{w}_0$ | Denoised $\varphi_1(\mathbf{w}_0)$ | Clean $\mathbf{w}_1$