A tutorial on score-based generative models with medical imaging applications

Jeffrey A. Fessler

EECS Department, BME Department, Dept. of Radiology
University of Michigan

http://web.eecs.umich.edu/~fessler

MIDAS mini-symposium
Generative AI: Diffusion Models for Scientific Machine Learning
2023-09-15

Acknowledgments:
Jason Hu, Xiaojian Xu, Mike McCann (LANL)

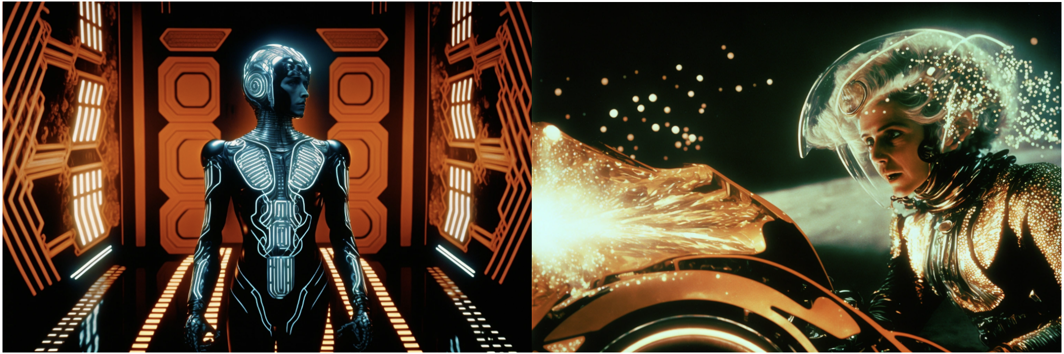
Outline

Introduction / tutorial
- Generative models
- Score matching / diffusion models
- Medical imaging applications

Patch-based score modeling
- Current results
- Summary

Bibliography

Extra: toy exploration



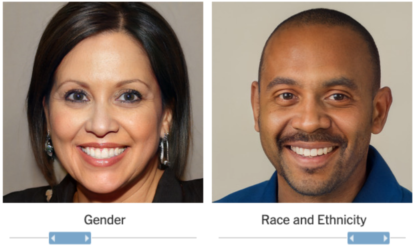
Computer ("AI") generated stills from hypothetical movie: Chilean director Alejandro Jodorowsky's 1976 version of "Tron" using midjourney.com as reported in 2023-01-13 NY Times article "This film does not exist" by director Frank Pavich.

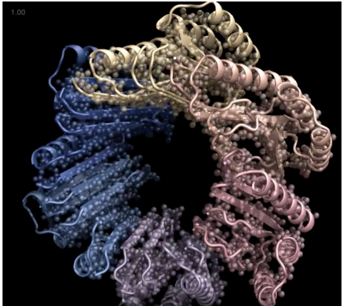
Generative models are hot in the news

- 2020-11-21 NY Times "Designed to Deceive: Do These People Look Real to You?"
  Article about generated (aka fake) faces.

- 2022-10-21 NY Times "A Coming-Out Party for Generative A.I., Silicon Valley's New Craze"
  (about "Stable Diffusion" image generator)
  https://nyti.ms/3SjsN0k

- 2023-01-09 NY Times "A.I. Turns Its Artistry to Creating New Human Proteins"
  https://nyti.ms/3IzY66m





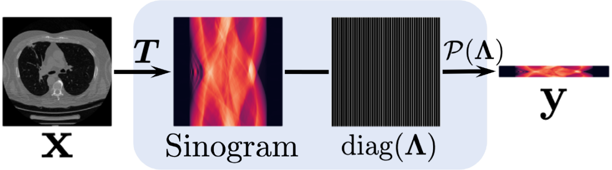
Generative models are hot in imaging / inverse problems

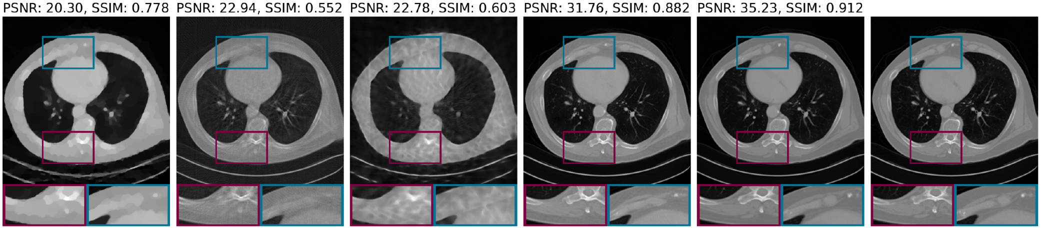
Zhao, Ye, Bresler: Jan. 2023 IEEE SpMag survey paper [1]

- Generative adversarial network (GAN) models
- Variation auto-encoder (VAE) models [2]
- Normalizing flows [3, 4]
- Score-based diffusion models
  - Zaccharie Ramzi et al., NeurIPS Workshop 2020 [5]
  - Yang Song & Liyue Shen et al., NeurIPS Workshop 2021, ICLR 2022 [6, 7]
  - Ajil Jalal et al. . . . Jon Tamir, NeurIPS 2021 [8]
  - Hyungjin Chung & Jong Chul Ye, MIA, Aug. 2022 [9]
  - Luo et al., MRM, 2023 [10]
  - . . .
- Kazerouni et al. [11] have github catalog, including 17 (!) survey papers
- . . . (hopelessly incomplete lists)

From Song & Shen et al., ICLR 2022. Trained with 47K 2D images, 23 projection views [7]

$$\boldsymbol{A} = \mathcal{P}(\boldsymbol{\Lambda})\boldsymbol{T}$$





(a) FISTA-TV (b) cGAN (c) Neumann (d) SIN-4c-PRN (e) Ours (f) Ground truth

Generative models: What

- A *generative model* is:
  -

Generative models: What

- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  -

Generative models: What

- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  -

Generative models: What

- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  - Numbers
  -

Generative models: What

- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  - Numbers
  - Text, images, code, music, video, molecules, materials, robotic plans, . . .
  -

Generative models: What

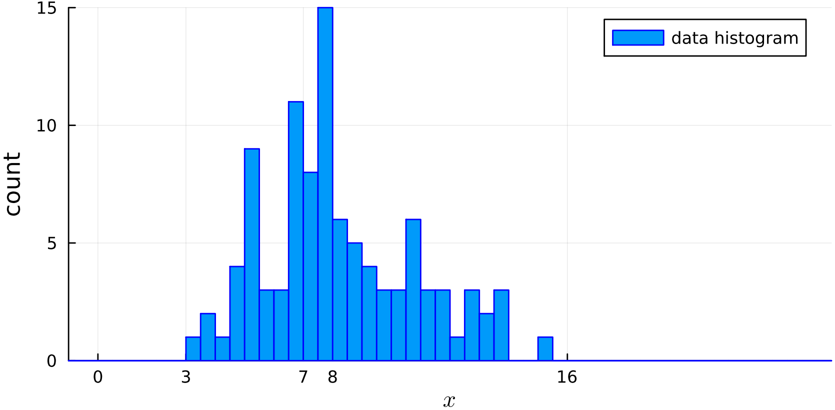
- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  - Numbers
  - Text, images, code, music, video, molecules, materials, robotic plans, ...
  - fake news, ... (*cf.* NeurIPS: Societal Impact and Potential Harmful Consequences)
-

- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  - Numbers
  - Text, images, code, music, video, molecules, materials, robotic plans, ...
  - fake news, ... (*cf.* NeurIPS: Societal Impact and Potential Harmful Consequences)
- Usually the model depends on some (or many) parameters, $\boldsymbol{\theta}$.
  i.e., we work with a parametric model $\mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$.
-

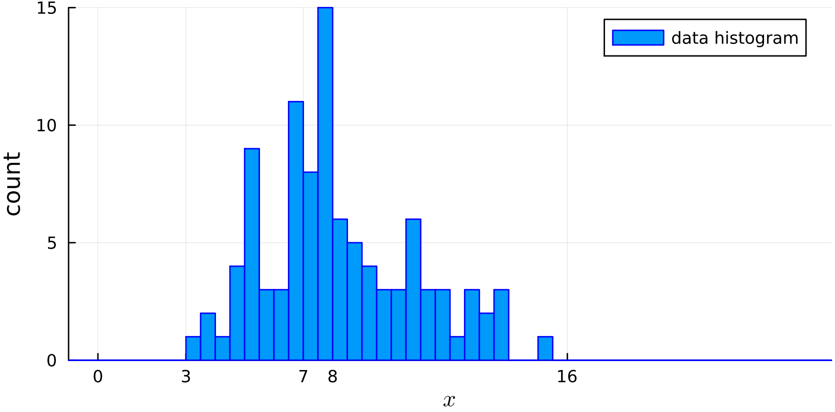
- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  - Numbers
  - Text, images, code, music, video, molecules, materials, robotic plans, ...
  - fake news, ... (*cf.* NeurIPS: Societal Impact and Potential Harmful Consequences)
- Usually the model depends on some (or many) parameters, $\boldsymbol{\theta}$.
  i.e., we work with a parametric model $\mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$.
- Challenge 1: Learning the parameters $\boldsymbol{\theta}$ from some training data.
  Hopefully that data is representative of the population of interest.
  (skin color, brain lesions...)
-

- A *generative model* is:
  - a model for some probability distribution $\mathrm{p}(\boldsymbol{x})$,
  - and usually a method for drawing samples from that distribution.
    "generation" (think: random number generator)
- What will be generated by drawing samples?
  - Numbers
  - Text, images, code, music, video, molecules, materials, robotic plans, ...
  - fake news, ... (*cf.* NeurIPS: Societal Impact and Potential Harmful Consequences)
- Usually the model depends on some (or many) parameters, $\boldsymbol{\theta}$.
  i.e., we work with a parametric model $\mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$.
- Challenge 1: Learning the parameters $\boldsymbol{\theta}$ from some training data.
  Hopefully that data is representative of the population of interest.
  (skin color, brain lesions...)
- Challenge 2: Drawing samples (generating) efficiently from $\mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$.

Toy example



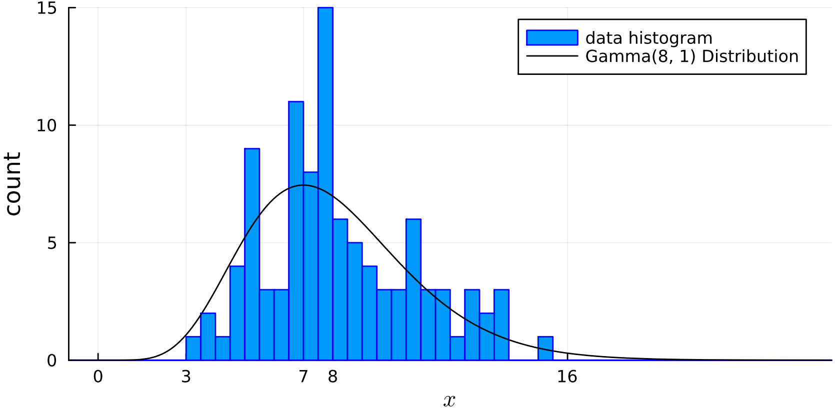
Training data, *e.g.*, upper left pixel value in each of a set of face images
How to generate samples from this distribution?



Trivial option for data generation: draw samples at random from training data.

+ Nonparameteric (no model bias)
– No generalization (nothing new)
+ "Memorization" - no hallucinations!
– Curse of dimensionality

Toy example



- Maximum-likelihood (ML) estimation to fit the two Gamma distribution parameters requires an iterative method.
- After fitting, drawing samples (aka generation) is not trivial, involving an acceptance-rejection method.

- Generating samples, *e.g.*, computer-assisted content creation
  computer-generated graphics, music, poetry, college essays, . . .
  -

- Generating samples, *e.g.*, computer-assisted content creation
  computer-generated graphics, music, poetry, college essays, ...
  - Often those examples involve conditional distributions $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$
  - generate a $\underbrace{\text{jazz melody}}_{\boldsymbol{x}}$ in $\underbrace{\text{the style of Miles Davis}}_{\boldsymbol{z}}$
  - design a $\underbrace{\text{composite material}}_{\boldsymbol{x}}$ with $\underbrace{\text{density } \rho \text{ and elasticity } E \ \ldots}_{\boldsymbol{z}}$
-

- Generating samples, *e.g.*, computer-assisted content creation computer-generated graphics, music, poetry, college essays, . . .
  - Often those examples involve conditional distributions $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{x} | \boldsymbol{z})$
  - generate a $\underbrace{\text{jazz melody}}_{\boldsymbol{x}}$ in $\underbrace{\text{the style of Miles Davis}}_{\boldsymbol{z}}$
  - design a $\underbrace{\text{composite material}}_{\boldsymbol{x}}$ with $\underbrace{\text{density } \rho \text{ and elasticity } E \ \ldots}_{\boldsymbol{z}}$
- LLM / ChatGPT: *Write a short joke about AI*
  -

Generative models: Why

- Generating samples, *e.g.*, computer-assisted content creation computer-generated graphics, music, poetry, college essays, ...
  - Often those examples involve conditional distributions $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$
  - generate a $\underbrace{\text{jazz melody}}_{\boldsymbol{x}}$ in $\underbrace{\text{the style of Miles Davis}}_{\boldsymbol{z}}$
  - design a $\underbrace{\text{composite material}}_{\boldsymbol{x}}$ with $\underbrace{\text{density } \rho \text{ and elasticity } E \dots}_{\boldsymbol{z}}$
- LLM / ChatGPT: *Write a short joke about AI*
  - `Why did the AI cross the road? To get to the other side of the algorithm!`
  - `Why did the AI cross the road? To get to the other cache.`
  - `Why did the AI cross the road? To get to the other dataset!`
-

- Generating samples, *e.g.*, computer-assisted content creation computer-generated graphics, music, poetry, college essays, ...
  - Often those examples involve conditional distributions $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$
  - generate a $\underbrace{\text{jazz melody}}_{\boldsymbol{x}}$ in $\underbrace{\text{the style of Miles Davis}}_{\boldsymbol{z}}$
  - design a $\underbrace{\text{composite material}}_{\boldsymbol{x}}$ with $\underbrace{\text{density } \rho \text{ and elasticity } E \ \ldots}_{\boldsymbol{z}}$
- LLM / ChatGPT: *Write a short joke about AI*
  - `Why did the AI cross the road? To get to the other side of the algorithm!`
  - `Why did the AI cross the road? To get to the other cache.`
  - `Why did the AI cross the road? To get to the other dataset!`
- *Is ChatGPT based on a generative model?*

  `Yes, ChatGPT is based on a generative language model. Specifically, it is built using the GPT (Generative Pre-trained Transformer) architecture, which is a type of deep neural network designed for generating natural language text.`
-

- Generating samples, *e.g.*, computer-assisted content creation computer-generated graphics, music, poetry, college essays, ...
  - Often those examples involve conditional distributions $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{x} | \boldsymbol{z})$
  - generate a $\underbrace{\text{jazz melody}}_{\boldsymbol{x}}$ in $\underbrace{\text{the style of Miles Davis}}_{\boldsymbol{z}}$
  - design a $\underbrace{\text{composite material}}_{\boldsymbol{x}}$ with $\underbrace{\text{density } \rho \text{ and elasticity } E \ \ldots}_{\boldsymbol{z}}$
- LLM / ChatGPT: *Write a short joke about AI*
  - `Why did the AI cross the road? To get to the other side of the algorithm!`
  - `Why did the AI cross the road? To get to the other cache.`
  - `Why did the AI cross the road? To get to the other dataset!`
- *Is ChatGPT based on a generative model?*
  `Yes, ChatGPT is based on a generative language model. Specifically, it is built using the GPT (Generative Pre-trained Transformer) architecture, which is a type of deep neural network designed for generating natural language text.`
- UMGPT: `Why was the computer cold? It left its Windows open! (2016 Dribbble)`

- Bayesian inference (*e.g.*, *science and engineering*)
  -

Generative models: Why

- Bayesian inference (*e.g.*, *science and engineering*)
  - Given (test) data $\boldsymbol{y}$ related to a latent variable $\boldsymbol{x}$
  - Known likelihood function $\mathrm{p}(\boldsymbol{y}|\boldsymbol{x})$, e.g., for human-made sensors
  - Want to estimate $\boldsymbol{x}$ from $\boldsymbol{y}$
  -

- Bayesian inference (*e.g.*, *science and engineering*)
  - Given (test) data $\boldsymbol{y}$ related to a latent variable $\boldsymbol{x}$
  - Known likelihood function $\mathrm{p}(\boldsymbol{y}|\boldsymbol{x})$, e.g., for human-made sensors
  - Want to estimate $\boldsymbol{x}$ from $\boldsymbol{y}$
  - Maximum-likelihood estimation is insufficient for under-determined problems

$$\arg\max_{\boldsymbol{x}} \mathrm{p}(\boldsymbol{y}|\boldsymbol{x})$$

    (sparse-view X-ray CT, accelerated MRI scans, . . .)
  - Bayesian methods use the posterior

$$\mathrm{p}(\boldsymbol{x}|\boldsymbol{y}) = \frac{\mathrm{p}(\boldsymbol{y}|\boldsymbol{x})\,\mathrm{p}(\boldsymbol{x})}{\mathrm{p}(\boldsymbol{y})}$$

  -

Generative models: Why

- Bayesian inference (*e.g., science and engineering*)
  - Given (test) data $\boldsymbol{y}$ related to a latent variable $\boldsymbol{x}$
  - Known likelihood function $\mathrm{p}(\boldsymbol{y}|\boldsymbol{x})$, e.g., for human-made sensors
  - Want to estimate $\boldsymbol{x}$ from $\boldsymbol{y}$
  - Maximum-likelihood estimation is insufficient for under-determined problems

$$\arg\max_{\boldsymbol{x}} \mathrm{p}(\boldsymbol{y}|\boldsymbol{x})$$

    (sparse-view X-ray CT, accelerated MRI scans, ...)
  - Bayesian methods use the posterior

$$\mathrm{p}(\boldsymbol{x}|\boldsymbol{y}) = \frac{\mathrm{p}(\boldsymbol{y}|\boldsymbol{x})\,\mathrm{p}(\boldsymbol{x})}{\mathrm{p}(\boldsymbol{y})}$$

  - Here the prior $\mathrm{p}(\boldsymbol{x})$ is for quantifying (prior) probability, not necessarily for generation.

Benefits of Bayesian methods

A model for the posterior $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$ opens many doors:

- Maximizing $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$ is maximum a posteriori (MAP) estimation
- The conditional mean $\mathrm{E}[\boldsymbol{x}|\boldsymbol{y}] = \int \boldsymbol{x} \, \mathrm{p}(\boldsymbol{x}|\boldsymbol{y}) \, \mathrm{d}\boldsymbol{x}$ is the MMSE estimator
- Sampling from the posterior facilitates uncertainty quantification in inference

All of these require the prior $\mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$.

Benefits of Bayesian methods

A model for the posterior $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$ opens many doors:

- Maximizing $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$ is maximum a posteriori (MAP) estimation
- The conditional mean $\mathrm{E}[\boldsymbol{x}|\boldsymbol{y}] = \int \boldsymbol{x}\,\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})\,\mathrm{d}\boldsymbol{x}$ is the MMSE estimator
- Sampling from the posterior facilitates uncertainty quantification in inference

All of these require the prior $\mathrm{p}(\boldsymbol{x};\boldsymbol{\theta})$. Or do they?

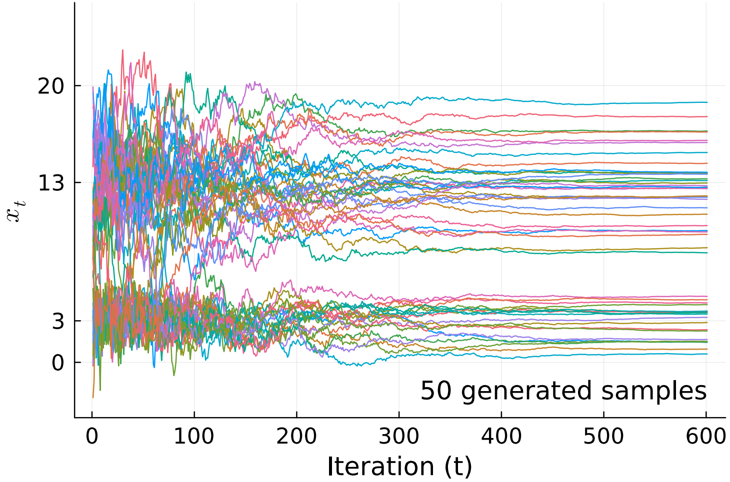
Sampling from the *prior* $\mathrm{p}(\boldsymbol{x};\boldsymbol{\theta})$ just needs its score function $\nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x};\boldsymbol{\theta})$, using Langevin dynamics, aka stochastic gradient ascent of log-prior:

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \alpha_t \nabla \log \mathrm{p}(\boldsymbol{x}_{t-1};\boldsymbol{\theta}) + \beta_t \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \quad t = 1, \ldots, T.$$

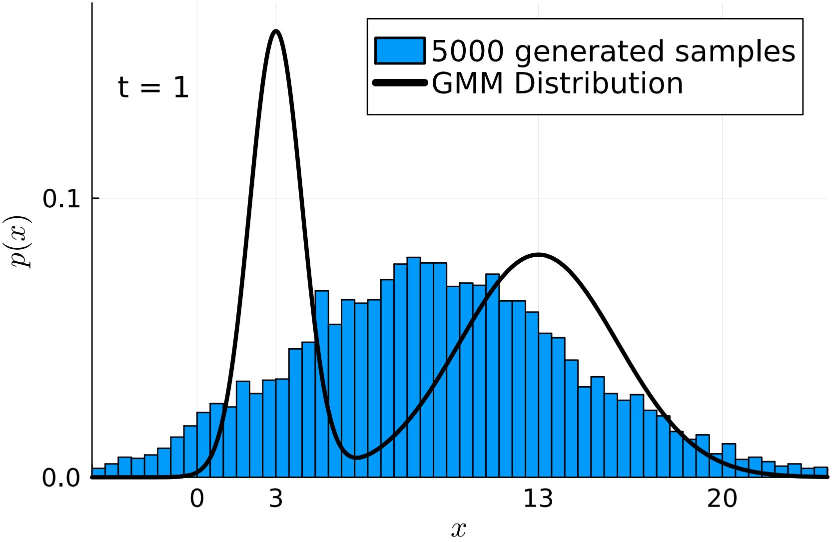
- Draws samples from $\mathrm{p}(\boldsymbol{x};\boldsymbol{\theta})$ for suitable choices of $\{\alpha_t\}$, $\{\beta_t\}$, and (large) T [12].
- If $\alpha_t = 0$ and $\beta_t = \beta$, then akin to (isotropic) diffusion or Brownian motion

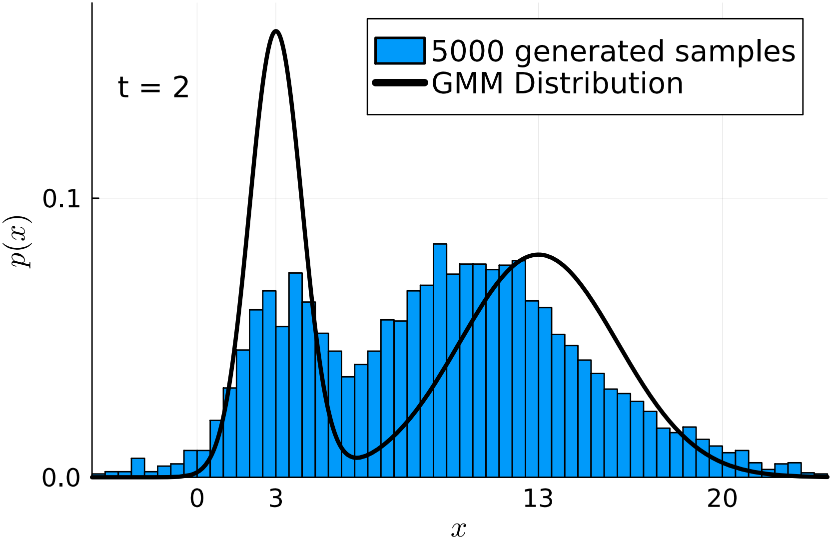
Langevin example: traces

$x_t = x_{t-1} + \alpha_t \nabla \log \mathrm{p}(x_{t-1}; \boldsymbol{\theta}) + \beta_t \mathcal{N}(0,1), \quad t = 1, \ldots, T$, for a GMM $\{\alpha_t\}$ and $\{\beta_t\}$ decaying geometrically, with $x_0 \sim \mathcal{N}(\mu, \sigma)$.



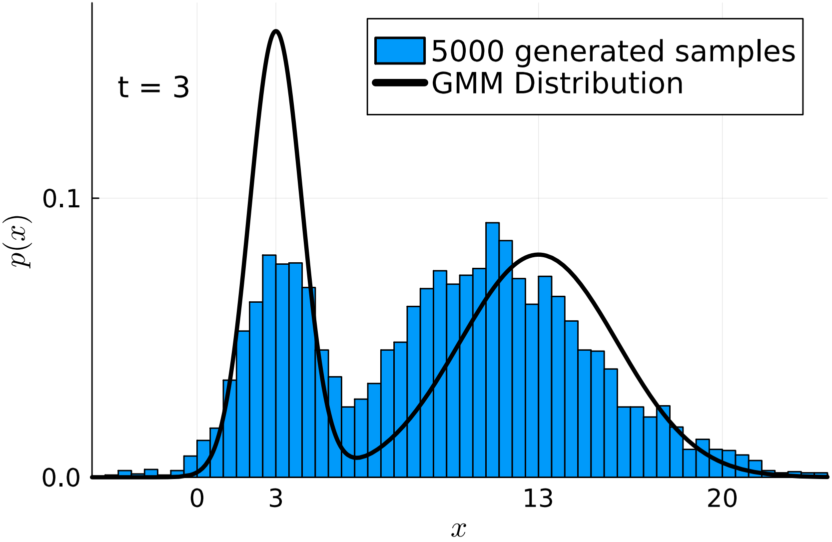
Langevin example: histogram



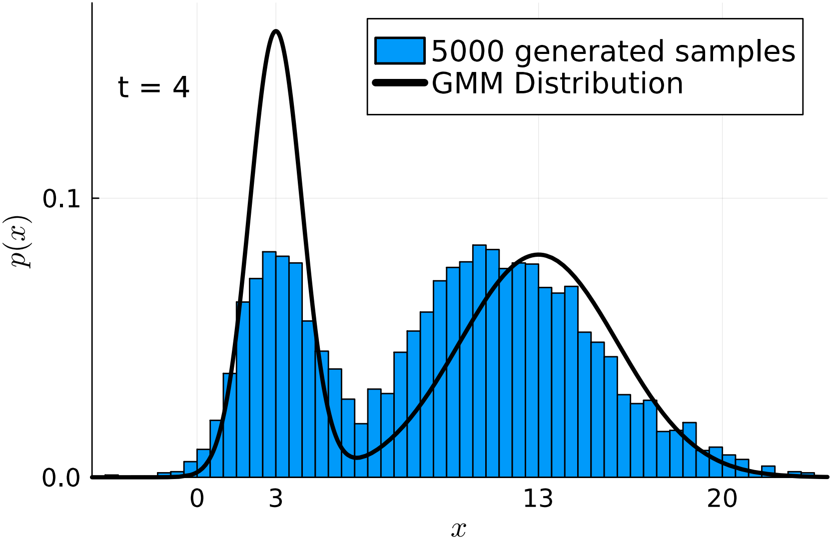


5000 generated samples
GMM Distribution
t = 2
0.1
0.0
$p(x)$
0
3
13
20
x

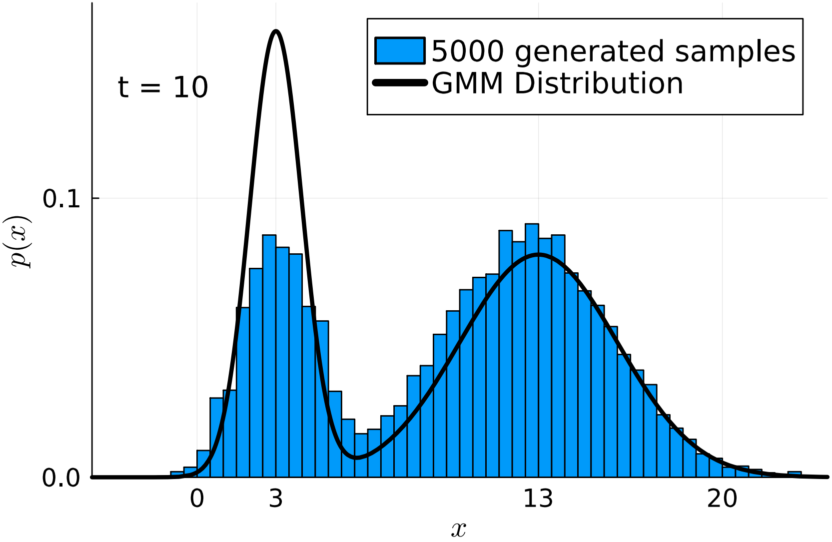
Langevin example: histogram



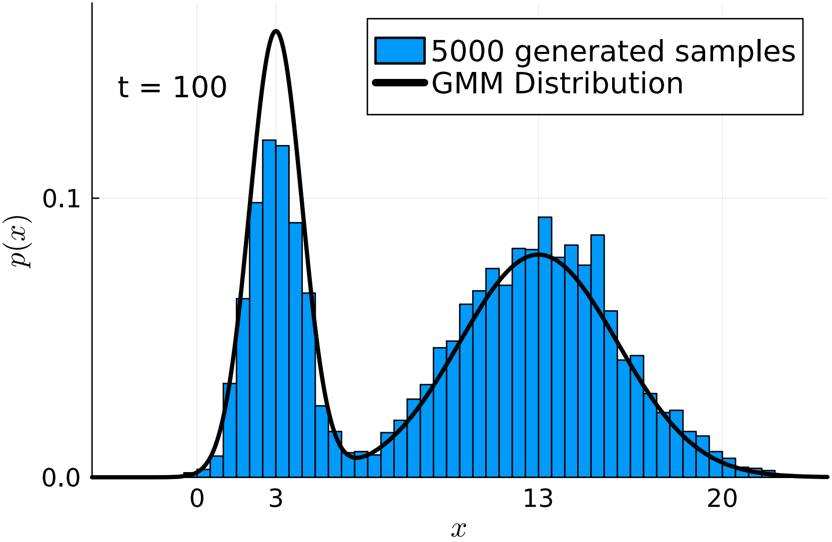
Langevin example: histogram



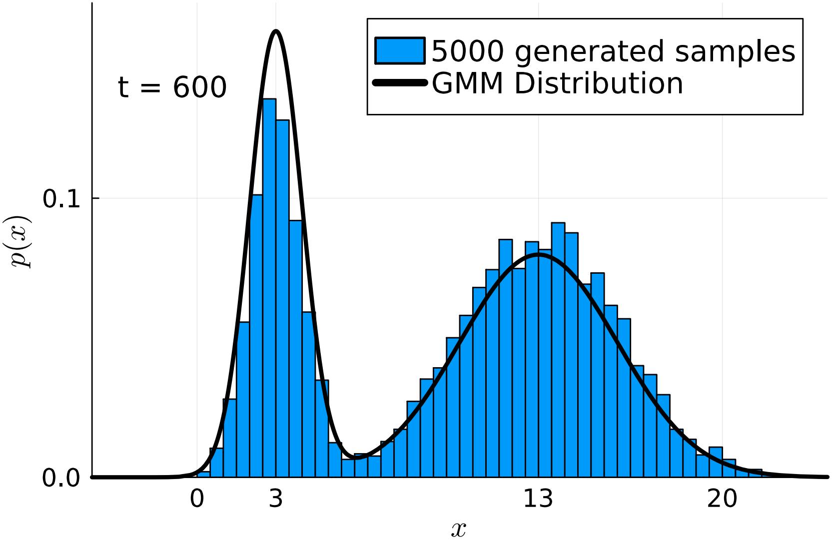
Langevin example: histogram



Langevin example: histogram



Langevin example: histogram



Sampling from the *posterior* $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$ and MAP estimation is similar, using Langevin dynamics, aka stochastic gradient ascent of log-posterior:

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \alpha_t \left(\nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}_{t-1}; \boldsymbol{\theta}) + \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}_{t-1}; \boldsymbol{\theta}) \right) + \beta_t \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \quad t = 1, \ldots, T.$$

- Draws samples from $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$ for suitable choices of $\{\alpha_t\}$, $\{\beta_t\}$, and (large) T.
-

Sampling from the *posterior* $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$ and MAP estimation is similar,
using Langevin dynamics, aka stochastic gradient ascent of log-posterior:

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \alpha_t \left(\nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}_{t-1}; \boldsymbol{\theta}) + \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}_{t-1}; \boldsymbol{\theta}) \right) + \beta_t \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \quad t = 1, \ldots, T.$$

- Draws samples from $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta})$ for suitable choices of $\{\alpha_t\}$, $\{\beta_t\}$, and (large) T.
- So how do we learn a score function?

- Typical distribution models: $\mathrm{p}(\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})}\, \mathrm{e}^{-U(\boldsymbol{x};\boldsymbol{\theta})}$.
  Goal: learn $\boldsymbol{\theta}$ from training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$
- For IID samples $\{\boldsymbol{x}_t\}$, one could try to learn $\boldsymbol{\theta}$ by ML estimation:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \mathrm{p}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T; \boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{t=1}^{T} \log(\mathrm{p}(\boldsymbol{x}_t; \boldsymbol{\theta}))$$
$$= \arg\max_{\boldsymbol{\theta}} \left(-T\, Z(\boldsymbol{\theta}) + \sum_{t=1}^{T} -U(\boldsymbol{x}_t; \boldsymbol{\theta}) \right).$$

  Typically intractable due to the partition function $Z(\boldsymbol{\theta})$.
-

Distribution learning vs score learning

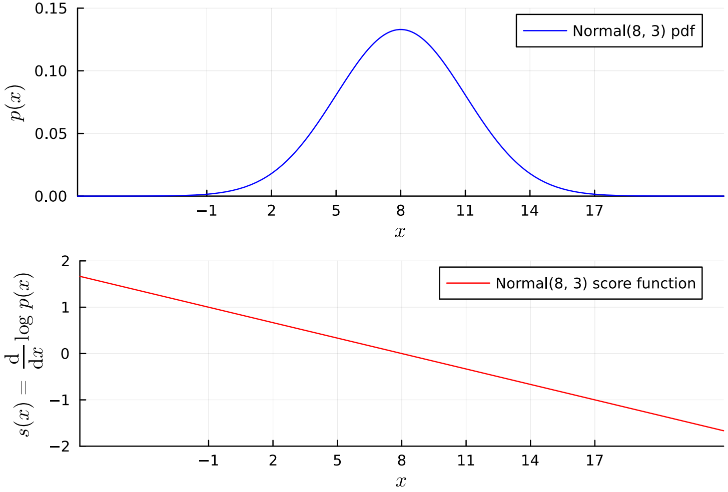
- Typical distribution models: $\mathrm{p}(\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})}\,\mathrm{e}^{-U(\boldsymbol{x};\boldsymbol{\theta})}$.
  Goal: learn $\boldsymbol{\theta}$ from training data $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T$
- For IID samples $\{\boldsymbol{x}_t\}$, one could try to learn $\boldsymbol{\theta}$ by ML estimation:

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \arg\max_{\boldsymbol{\theta}} \mathrm{p}(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T;\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{t=1}^{T} \log(\mathrm{p}(\boldsymbol{x}_t;\boldsymbol{\theta})) \\ &= \arg\max_{\boldsymbol{\theta}} \left(-T\,{\color{red}Z(\boldsymbol{\theta})} + \sum_{t=1}^{T} -U(\boldsymbol{x}_t;\boldsymbol{\theta})\right).\end{aligned}$$

  Typically intractable due to the partition function $Z(\boldsymbol{\theta})$.
- In contrast, the score function is easier to handle:

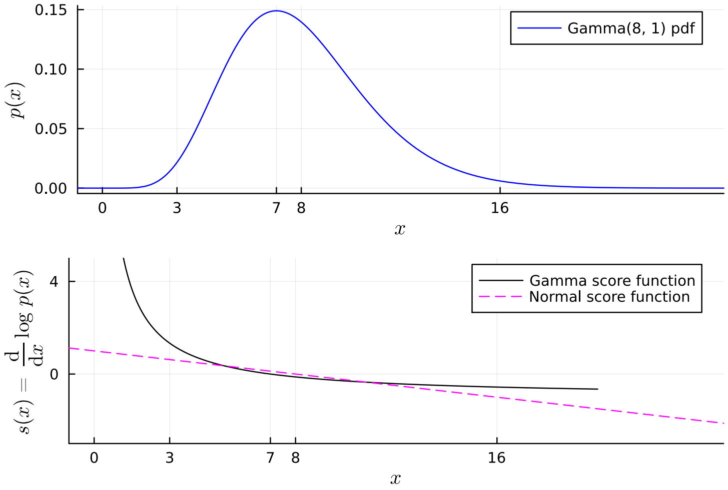
$$\boldsymbol{s}(\boldsymbol{x};\boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x};\boldsymbol{\theta}) = \nabla_{\boldsymbol{x}}\left(-\log Z(\boldsymbol{\theta}) - U(\boldsymbol{x};\theta)\right) = -\nabla_{\boldsymbol{x}} U(\boldsymbol{x};\theta).$$

Score function example: 1D Gaussian



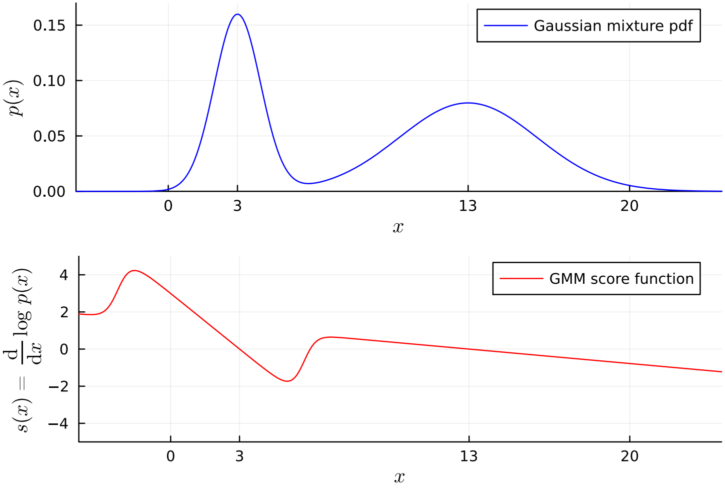
$$\mathsf{p}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \mathrm{e}^{-(x-\mu)^2/2\sigma^2} \Longrightarrow s(x) = \frac{\mathrm{d}}{\mathrm{d}x} \log \mathsf{p}(x) = \frac{1}{\sigma^2}(x - \mu)$$

Score function example: 1D Gamma

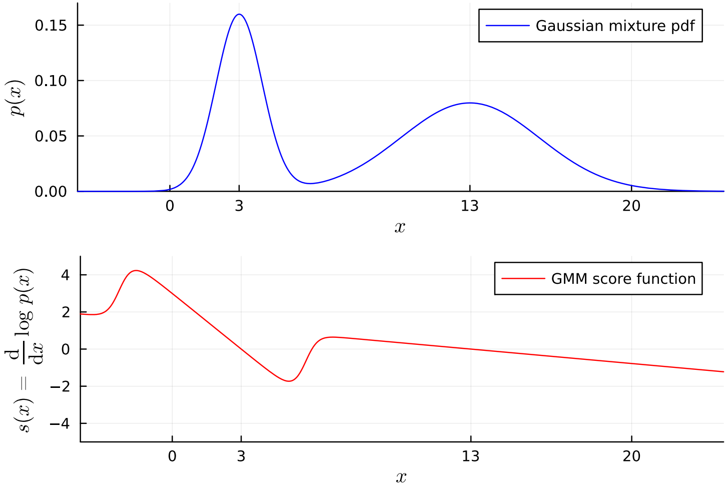


Note sign of score function to left and right of mode.

Score function example: 1D GMM



Score function example: 1D GMM



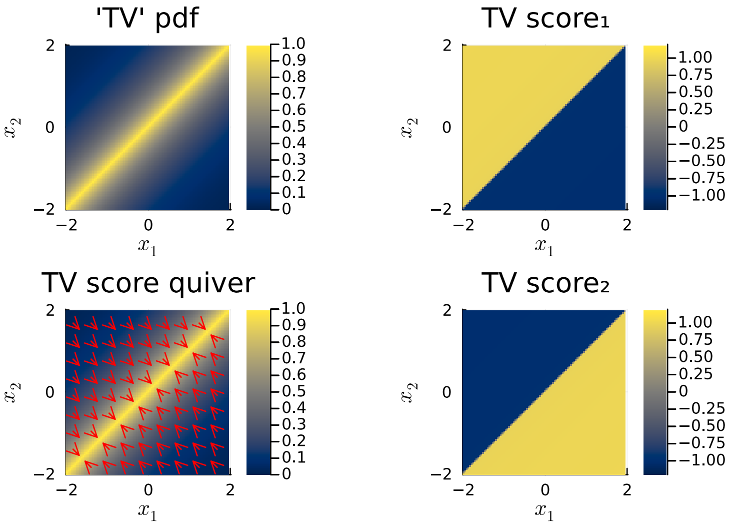
- Could you recover the pdf $\mathrm{p}(x)$ from its score function $s(x)$ in 1D?

Score function example: 2D

Total variation (TV) prior for 2×1 patch:

$$\mathrm{p}(\boldsymbol{x}) \propto \mathrm{e}^{-\beta |x_2 - x_1|}$$

$$s(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x})$$
$$\propto \operatorname{sign}(x_1 - x_2) \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$



Learning score functions

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function

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

  but $\mathrm{p}(\boldsymbol{x})$ is unknown

-

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function

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

  but $\mathrm{p}(\boldsymbol{x})$ is unknown

- Nonparametric density estimation approach?
  - Kernel density estimation: $\mathrm{q}_\sigma(\boldsymbol{x}) \triangleq \frac{1}{T} \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i) \stackrel{?}{\approx} \mathrm{p}(\boldsymbol{x})$
  - Apply score definition to q_σ:

$$\boldsymbol{s}_\sigma(\boldsymbol{x}) \triangleq \nabla_{\boldsymbol{x}} \log \mathrm{q}_\sigma(\boldsymbol{x}) = \frac{\frac{1}{T} \sum_{t=1}^{T} \nabla\, g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)}{\frac{1}{T} \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)}$$

  -

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function

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

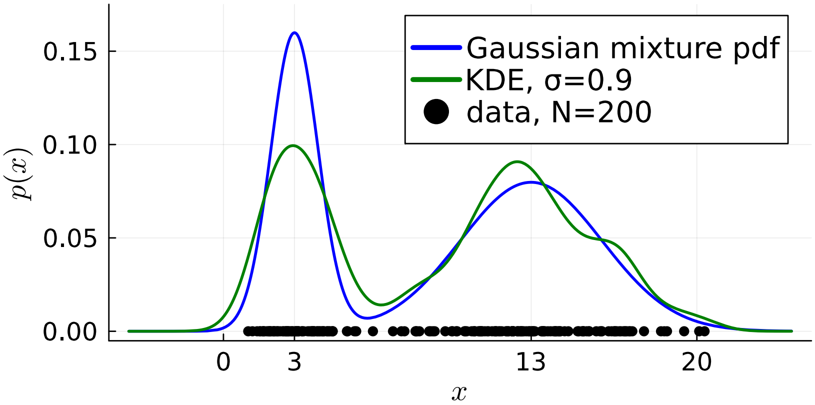
but $\mathrm{p}(\boldsymbol{x})$ is unknown

- Nonparametric density estimation approach?
  - Kernel density estimation: $\mathrm{q}_\sigma(\boldsymbol{x}) \triangleq \frac{1}{T} \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i) \overset{?}{\approx} \mathrm{p}(\boldsymbol{x})$
  - Apply score definition to q_σ:

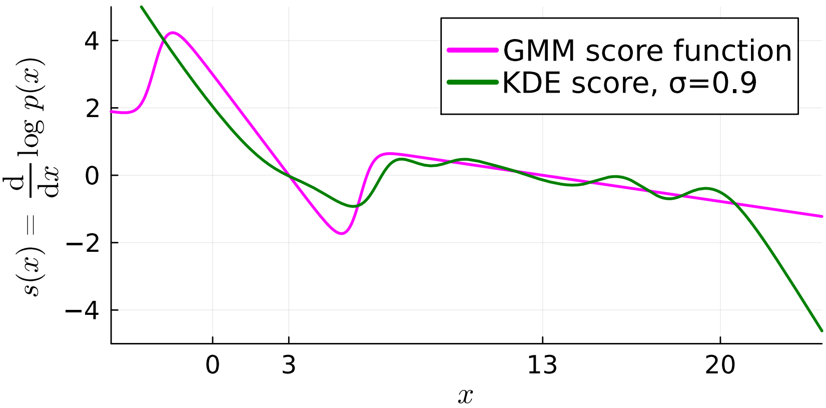
$$\boldsymbol{s}_\sigma(\boldsymbol{x}) \triangleq \nabla_{\boldsymbol{x}} \log \mathrm{q}_\sigma(\boldsymbol{x}) = \frac{\frac{1}{T} \sum_{t=1}^{T} \nabla g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)}{\frac{1}{T} \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)}$$

  - Seems impractical:
    $O(1)$ training time, but $O(T)$ work at test time; curse of dimensionality

Score from kernel density estimate



Score from kernel density estimate



Score matching (parametric)

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) \stackrel{?}{=} \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$
-

Score matching (parametric)

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) \stackrel{?}{=} \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$
- Explicit score matching (ESM)
  - Estimate data distribution (kernel density estimation): $\mathrm{q}_\sigma(\boldsymbol{x}) = \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)$
  - Match model score to data score (Hyvärinen, 2005 [13]):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ESM},\mathrm{q}}(\boldsymbol{\theta}), \quad J_{\mathrm{ESM},\mathrm{q}}(\boldsymbol{\theta}) \triangleq \frac{1}{2} \mathrm{E}_{\mathrm{q}(\boldsymbol{x})}\left[\left\|\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) - \nabla \log \mathrm{q}(\boldsymbol{x})\right\|_2^2\right]. \quad (1)$$

-

Score matching (parametric)

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) \stackrel{?}{=} \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$
- Explicit score matching (ESM)
  - Estimate data distribution (kernel density estimation): $\mathrm{q}_\sigma(\boldsymbol{x}) = \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)$
  - Match model score to data score (Hyvärinen, 2005 [13]):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ESM},\mathsf{q}}(\boldsymbol{\theta}), \quad J_{\mathrm{ESM},\mathsf{q}}(\boldsymbol{\theta}) \triangleq \frac{1}{2} \mathrm{E}_{\mathsf{q}(\boldsymbol{x})}\left[\|\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) - \nabla \log \mathrm{q}(\boldsymbol{x})\|_2^2 \right]. \quad (1)$$

- Implicit score matching (ISM):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ISM},\mathsf{q}_0}(\boldsymbol{\theta}), \quad J_{\mathrm{ISM},\mathsf{q}_0}(\boldsymbol{\theta}) = \frac{1}{T} \sum_{t=1}^{T} \sum_{i} \left(\partial_i s_i(\boldsymbol{x}_t; \boldsymbol{\theta}) + \frac{1}{2} \left| s_i(\boldsymbol{x}_t; \boldsymbol{\theta}) \right|^2 \right) \quad (2)$$

$$\partial_i s_i(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\partial}{\partial x_i} s_i(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\partial^2}{\partial x_i^2} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta}). \quad (3)$$

-

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) \overset{?}{=} \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$
- Explicit score matching (ESM)
  - Estimate data distribution (kernel density estimation): $\mathrm{q}_\sigma(\boldsymbol{x}) = \sum_{t=1}^T g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)$
  - Match model score to data score (Hyvärinen, 2005 [13]):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ESM,q}}(\boldsymbol{\theta}), \quad J_{\mathrm{ESM,q}}(\boldsymbol{\theta}) \triangleq \frac{1}{2} \mathrm{E}_{\mathrm{q}(\boldsymbol{x})} \left[\| \boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) - \nabla \log \mathrm{q}(\boldsymbol{x}) \|_2^2 \right]. \tag{1}$$

- Implicit score matching (ISM):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ISM,q_0}}(\boldsymbol{\theta}), \quad J_{\mathrm{ISM,q_0}}(\boldsymbol{\theta}) = \frac{1}{T} \sum_{t=1}^T \sum_i \left(\partial_i s_i(\boldsymbol{x}_t; \boldsymbol{\theta}) + \frac{1}{2} \left| s_i(\boldsymbol{x}_t; \boldsymbol{\theta}) \right|^2 \right) \tag{2}$$

$$\partial_i s_i(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\partial}{\partial x_i} s_i(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\partial^2}{\partial x_i^2} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta}). \tag{3}$$

- $O(T)$ work at training time; work at test time depends on parametric model $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta})$
-

Score matching (parametric)

- Given training data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$, learn score function $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) \stackrel{?}{=} \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta})$
- Explicit score matching (ESM)
  - Estimate data distribution (kernel density estimation): $\mathrm{q}_\sigma(\boldsymbol{x}) = \sum_{t=1}^{T} g_\sigma(\boldsymbol{x} - \boldsymbol{x}_i)$
  - Match model score to data score (Hyvärinen, 2005 [13]):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ESM,q}}(\boldsymbol{\theta}), \quad J_{\mathrm{ESM,q}}(\boldsymbol{\theta}) \triangleq \frac{1}{2} \mathrm{E}_{\mathrm{q}(\boldsymbol{x})}\left[\left\| \boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) - \nabla \log \mathrm{q}(\boldsymbol{x}) \right\|_2^2 \right]. \qquad (1)$$

- Implicit score matching (ISM):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} J_{\mathrm{ISM,q_0}}(\boldsymbol{\theta}), \quad J_{\mathrm{ISM,q_0}}(\boldsymbol{\theta}) = \frac{1}{T} \sum_{t=1}^{T} \sum_{i} \left(\partial_i s_i(\boldsymbol{x}_t; \boldsymbol{\theta}) + \frac{1}{2} \left| s_i(\boldsymbol{x}_t; \boldsymbol{\theta}) \right|^2 \right) \qquad (2)$$

$$\partial_i s_i(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\partial}{\partial x_i} s_i(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\partial^2}{\partial x_i^2} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta}). \qquad (3)$$

- $O(T)$ work at training time; work at test time depends on parametric model $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta})$
- Denoising score matching (DSM)

- Vincent, 2011 [14] showed this remarkable equivalence between ESM and DSM:

$$
\begin{aligned}
J_{\mathrm{ESM},\mathsf{q}_\sigma}(\boldsymbol{\theta}) &= \mathsf{E}_{\mathsf{q}_\sigma(\boldsymbol{x})}\left[\frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x};\boldsymbol{\theta}) - \nabla \log \mathsf{q}_\sigma(\boldsymbol{x})\right\|_2^2\right] \\
&= \frac{1}{T}\sum_{t=1}^{T}\int \frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x};\boldsymbol{\theta}) - \nabla \log \mathsf{q}_\sigma(\boldsymbol{x})\right\|_2^2 g_\sigma(\boldsymbol{x} - \boldsymbol{x}_t)\,\mathrm{d}\boldsymbol{x} \\
\stackrel{!}{=} J_{\mathrm{DSM},\mathsf{q}_\sigma}(\boldsymbol{\theta}) &\triangleq \frac{1}{T}\sum_{t=1}^{T}\mathsf{E}_{g_\sigma(\boldsymbol{z})}\left[\frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x}_t + \boldsymbol{z};\boldsymbol{\theta}) + \frac{\boldsymbol{z}}{\sigma^2}\right\|_2^2\right] \\
&\approx \frac{1}{T}\sum_{t=1}^{T}\frac{1}{M}\sum_{m=1}^{M}\frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x}_t + \boldsymbol{z}_{t,m};\boldsymbol{\theta}) + \frac{\boldsymbol{z}_{t,m}}{\sigma^2}\right\|_2^2
\end{aligned}
$$

-

- Vincent, 2011 [14] showed this remarkable equivalence between ESM and DSM:

$$
\begin{aligned}
J_{\mathrm{ESM},\mathsf{q}_\sigma}(\boldsymbol{\theta}) &= \mathsf{E}_{\mathsf{q}_\sigma(\boldsymbol{x})}\left[\frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x};\boldsymbol{\theta}) - \nabla \log \mathsf{q}_\sigma(\boldsymbol{x})\right\|_2^2\right] \\
&= \frac{1}{T}\sum_{t=1}^{T}\int \frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x};\boldsymbol{\theta}) - \nabla \log \mathsf{q}_\sigma(\boldsymbol{x})\right\|_2^2 \, g_\sigma(\boldsymbol{x}-\boldsymbol{x}_t)\,\mathrm{d}\boldsymbol{x} \\
\overset{!}{=} J_{\mathrm{DSM},\mathsf{q}_\sigma}(\boldsymbol{\theta}) &\triangleq \frac{1}{T}\sum_{t=1}^{T}\mathsf{E}_{g_\sigma(\boldsymbol{z})}\left[\frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x}_t+\boldsymbol{z};\boldsymbol{\theta}) + \frac{\boldsymbol{z}}{\sigma^2}\right\|_2^2\right] \\
&\approx \frac{1}{T}\sum_{t=1}^{T}\frac{1}{M}\sum_{m=1}^{M}\frac{1}{2}\left\|\boldsymbol{s}(\boldsymbol{x}_t+\boldsymbol{z}_{t,m};\boldsymbol{\theta}) + \frac{\boldsymbol{z}_{t,m}}{\sigma^2}\right\|_2^2
\end{aligned}
$$

- The last term is a kind of denoising operation.

- Q: How much noise (what σ) to use in DSM?
-

Noise-conditional score matching

- Q: How much noise (what σ) to use in DSM?
- A: many!
-

Noise-conditional score matching

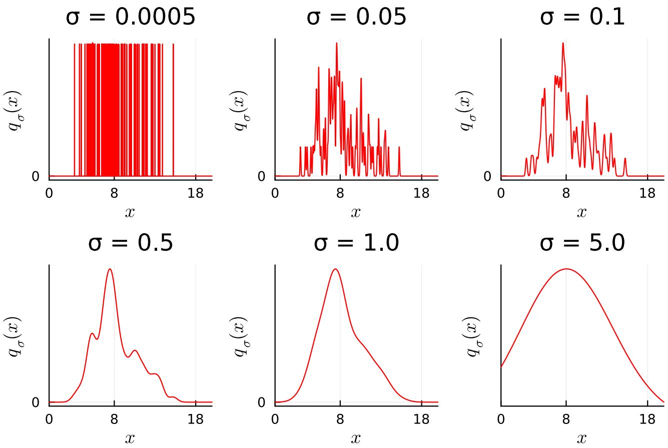
- Q: How much noise (what σ) to use in DSM?
- A: many!
- Noise-conditional score matching (NCSM) [15, eqn. (5)]:

$$\ell(\boldsymbol{\theta}; \sigma) \triangleq \frac{1}{2} \mathrm{E}_{\mathrm{q}_0(\boldsymbol{x})} \left[\mathrm{E}_{g_\sigma(\boldsymbol{z})} \left[\left\| \boldsymbol{s}(\boldsymbol{x} + \boldsymbol{z}; \boldsymbol{\theta}, \sigma) + \frac{\boldsymbol{z}}{\sigma^2} \right\|_2^2 \right] \right], \quad \mathcal{L}(\boldsymbol{\theta}; \{\sigma_l\}) = \frac{1}{L} \sum_{l=1}^{L} \sigma_l^2 \, \ell(\boldsymbol{\theta}; \sigma_l),$$

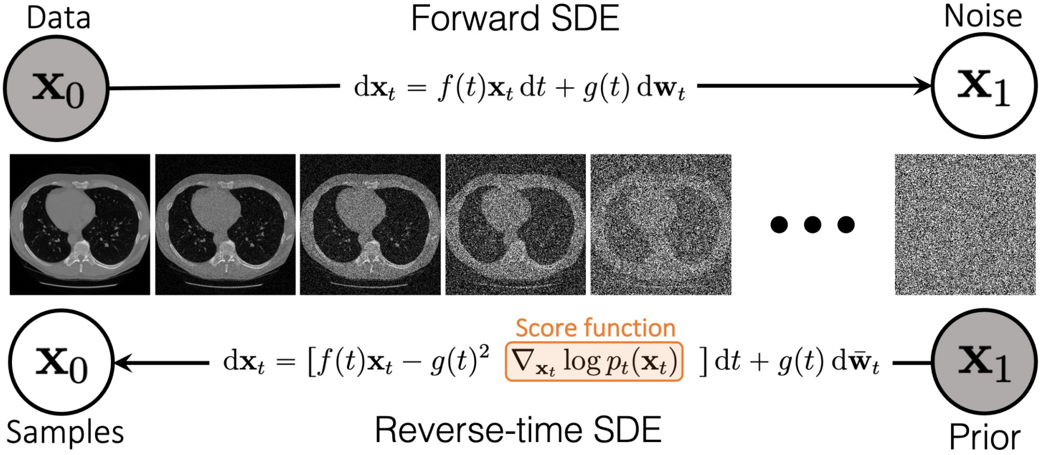
  where $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}, \sigma)$ denotes a *noise-conditional score network* (NCSN).
- Recommended choice [16]: $\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}, \sigma) \triangleq \tilde{\boldsymbol{s}}(\boldsymbol{x}; \boldsymbol{\theta}) / \sigma$, where $\tilde{\boldsymbol{s}}$ is unitless

DSM 1D illustration

$T = 100$
training samples



Shen & Song et al., NeurIPS 2021 [6]



Score-based diffusion models: trade-offs

- No adversarial training needed
- High quality sample generation (if enough training data)
-

Score-based diffusion models: trade-offs

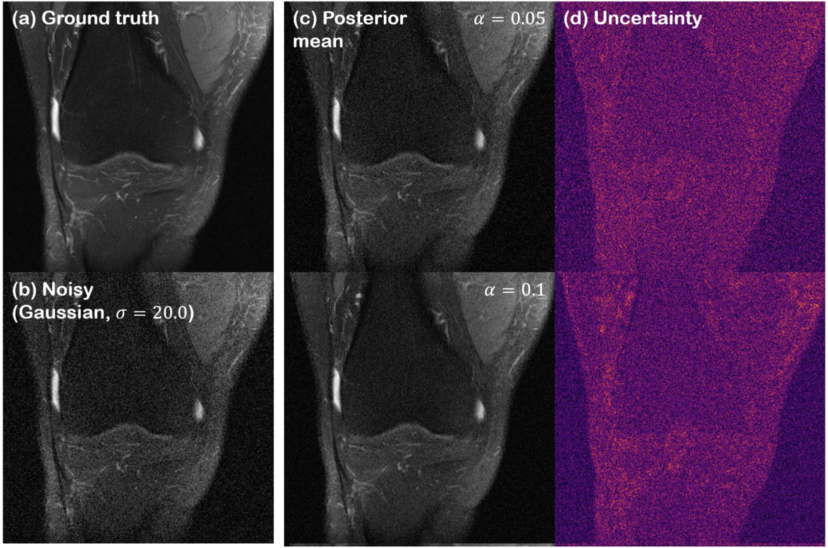
- No adversarial training needed
- High quality sample generation (if enough training data)
- Expensive sample generation (vs GAN models)
  - Distillation methods [17]
  - Consistency models [18]
  - Geometric decomposition [19]
  - Multi-scale [20, 21] and pyramidal [22] and coarse-to-fine [23] models
  - Faster ODE solvers [24]
  - Warm starts [25]
  - Latent diffusion models: use VAE and diffuse in latent space [26–28]. Used in Stable Diffusion by start-up Stability AI
  - 3D image reconstruction using 2D models [29, 30]

Medical imaging applications

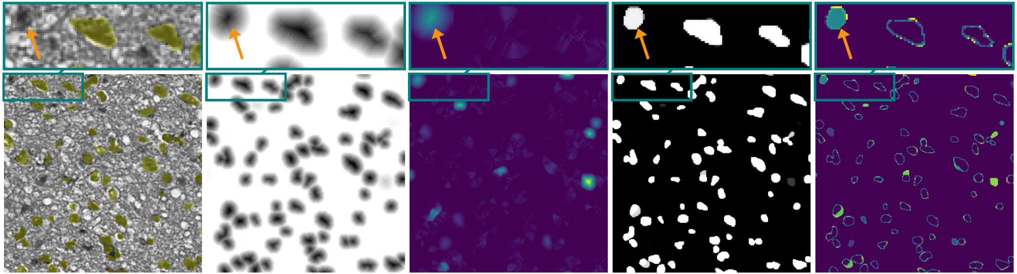
- Segmentation [31]
- Sparse-view CT reconstruction [32]
- Motion correction in MRI [33]
- Image analysis [11]
- Denoising / super-resolution [34]
- ...

Example: MR image denoising

Chung et al.,
IEEE T-MI 2023
[34]



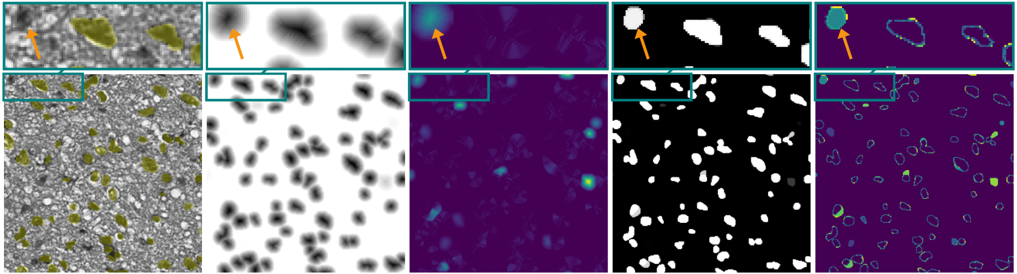
- Bogensperger et al., 2023 [31]; signed distance function instead of binary mask
  MoNuSeg microscopy images, Haematoxylin and Eosin (H&E) stained
  30 train, 14 test, size 1000×1000, >21,000 annotated nuclei



$x \otimes m_{gt}$ mean $\widetilde{m}(0)$ std $\widetilde{m}(0)$ mean m std m

-

- Bogensperger et al., 2023 [31]; signed distance function instead of binary mask
  MoNuSeg microscopy images, Haematoxylin and Eosin (H&E) stained
  30 train, 14 test, size 1000×1000, >21,000 annotated nuclei



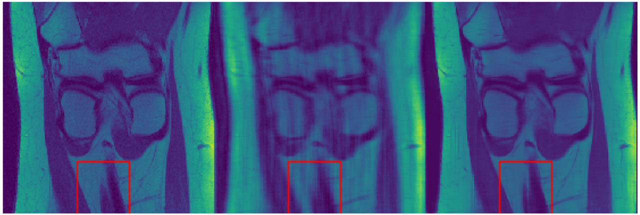
- Uncertainty map indicates possible segmentation errors

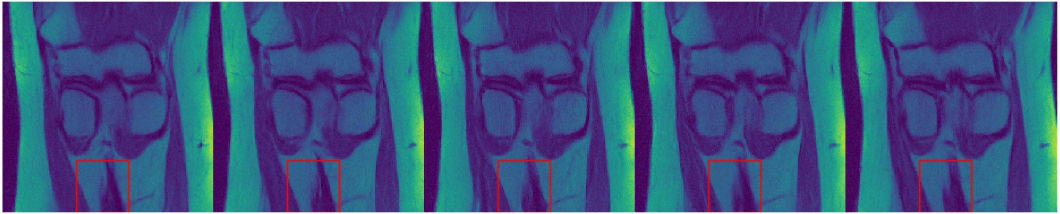
Example: MR image reconstruction & uncertainty

Ramzi et al.,
NeurIPS Workshop 2020 [5]

Fully sampled $\boldsymbol{x}$, zero-filled $\boldsymbol{A}'\boldsymbol{y}$,
Primal-dual enhanced U-Net:

$4\times$ acceleration





Posterior samples via neural score matching and annealed Hamiltonian Monte-Carlo

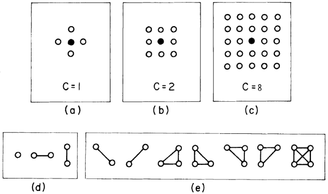
See video of posterior samples
Ramzi et al., 2020 [5]

NY Times article about fake faces

See it?

Markov random field models

(*e.g.*) Geman & Geman 1984 [35]



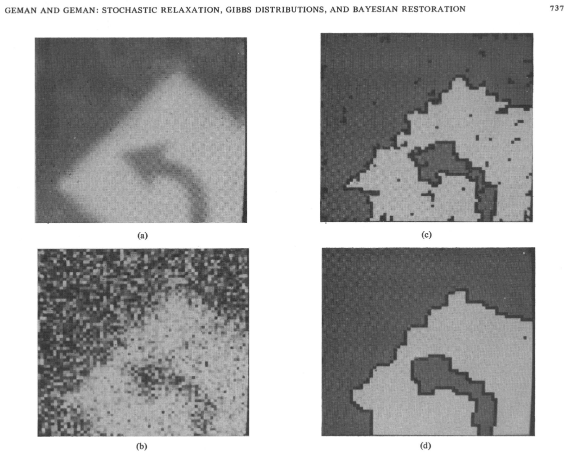


Fig. 7. (a) Blurred image (roadside scene). (b) Degraded image: Additive noise. (c) Restoration including line process; 100 iterations. (d) Restoration including line process; 1000 iterations.

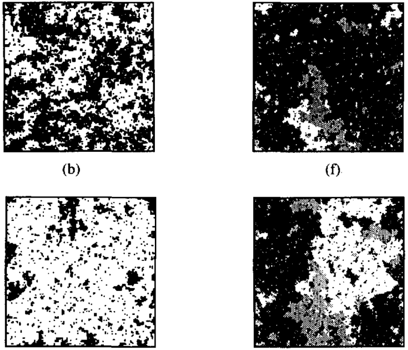
Mostly for inference?

MRF as generators? [36] T-PAMI 1994

An Empirical Study of the Simulation of Various Models Used for Images

A. J. Gray, J. W. Kay, and D. M. Titterington

***Abstract*— Markov random fields are typically used as priors in Bayesian image restoration methods to represent spatial information in the image. Commonly used Markov random fields are not in fact capable of representing the moderate-to-large scale clustering present in naturally occurring images and can also be time consuming to simulate,**



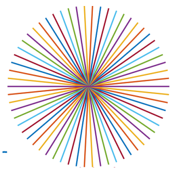
(b) (f)

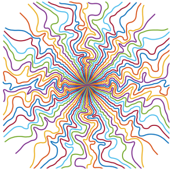
(c) (g)

Whole images vs patches?

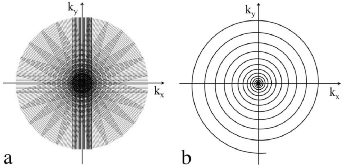
Jan. 2023 survey paper on generative models [1] does not mention “patch” once!?

MRI k-space sampling:

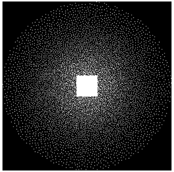




[37]



[38]



[39]

Patch-based models have long history in inverse problems, *e.g.*,

- patch GAN [40–42]
- patch dictionary models [43, 44]
- non-local means, BM3D
- Wasserstein patch prior [45, 46] . . .

Motivating questions

- Could patch-based generative models provide better robustness to distribution shifts, perhaps at the cost of reduced in-distribution performance?
- Especially in applications with very limited training data?
  e.g., dynamic MRI
- Can we use the “latest” generative models, namely score-based models, for patches?

Patch-based score modeling

- Start with MRF formulation, aka *fields of experts* model [47]:

$$\mathrm{p}(\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \mathrm{e}^{-\sum_c V_c(\boldsymbol{x};\boldsymbol{\theta})} = \frac{1}{Z(\boldsymbol{\theta})} \prod_c \mathrm{e}^{-V_c(\boldsymbol{x};\boldsymbol{\theta})}.$$

  - $\boldsymbol{\theta}$: parameter vector that describes the prior
  - V_c : *clique potential* for the cth image *patch*
  - $Z(\boldsymbol{\theta})$: intractable partition function
- Assume statistical spatial stationarity (image shift invariance):

$$V_c(\boldsymbol{x};\boldsymbol{\theta}) = V(\boldsymbol{G}_c\boldsymbol{x};\boldsymbol{\theta}),$$

  - $\boldsymbol{G}_c$: wide binary matrix that grabs pixels of the cth patch from image $\boldsymbol{x}$
  - $V(\boldsymbol{z};\boldsymbol{\theta})$: common parent clique function

- Resulting log-prior:

$$\log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta}) = -\log Z(\boldsymbol{\theta}) - \sum_c V(\boldsymbol{G}_c \boldsymbol{x}; \boldsymbol{\theta})$$

- Corresponding overall *image score function* arises from *patch score function*:

$$\boldsymbol{s}(\boldsymbol{x}; \boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x}; \boldsymbol{\theta}) = -\sum_c \boldsymbol{G}_c' \boldsymbol{s}_V(\boldsymbol{G}_c \boldsymbol{x}; \boldsymbol{\theta}), \qquad \boldsymbol{s}_V(\boldsymbol{v}; \boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{v}} V(\boldsymbol{v}; \boldsymbol{\theta}).$$

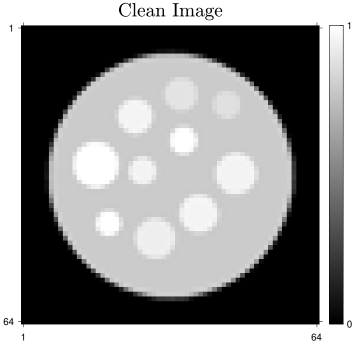
- All we must learn is the patch score function $\boldsymbol{s}_V(\boldsymbol{v}; \boldsymbol{\theta}) : \mathbb{R}^n \mapsto \mathbb{R}^n$, *e.g.*, a MLP.
- For training image patches $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_T\}$, apply *denoising score matching* (DSM) of Vincent, 2011 [14], typically for a range of noise variances σ^2 [12]:

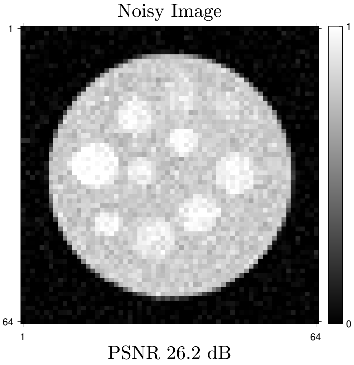
$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{T} \sum_{t=1}^{T} \mathrm{E}_{\sigma \sim p(\sigma)} \left[\sigma^2 \, \mathrm{E}_{\boldsymbol{z} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_n)} \left[\frac{1}{2} \left\| \boldsymbol{s}_V(\boldsymbol{v}_t + \boldsymbol{z}; \boldsymbol{\theta}, \sigma) + \frac{\boldsymbol{z}}{\sigma^2} \right\|_2^2 \right] \right].$$

- Final patch score model is $\boldsymbol{s}_V(\boldsymbol{v}; \hat{\boldsymbol{\theta}}, \sigma_{\min})$.

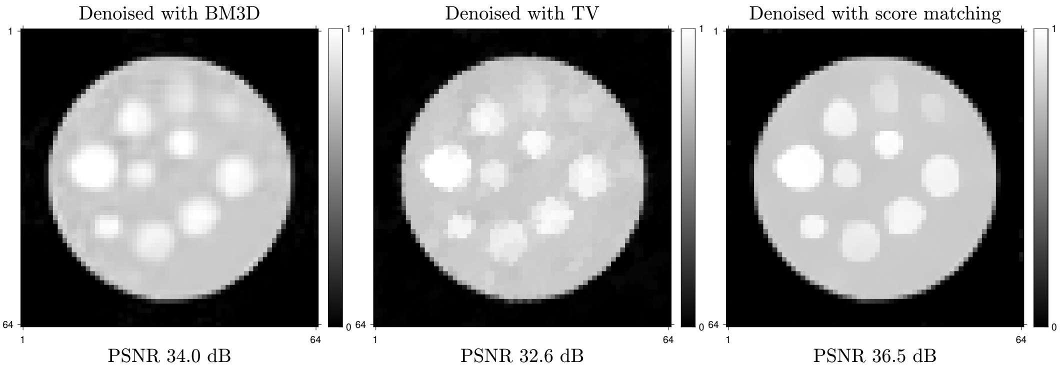
Simple exploration with *anecdotal* results

- 3×3 patches
- MLP patch score model (9, 40, 80, 160, 320, 320, 160, 80, 40, 9)
  first 5 with leaky ReLU, last 3 with tanh
- 1000 similar training examples





Denoising results

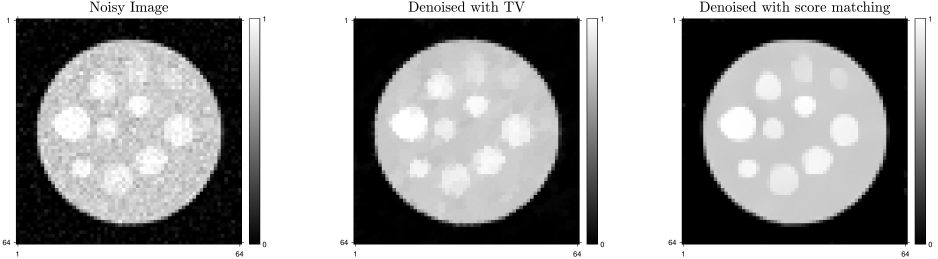


- BM3D from `https://webpages.tuni.fi/foi/GCF-BM3D`
- TV regularization parameter optimized by oracle for best PSNR.
- MAP estimate by greedy gradient ascent of log posterior: (no β!)

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \nabla_{\boldsymbol{x}} \log \mathrm{p}\left(\boldsymbol{x}_k | \boldsymbol{y}; \hat{\boldsymbol{\theta}}\right) = \boldsymbol{x}_k + \alpha_k \left(\nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}_k) - \sum_c \boldsymbol{G}_c' \boldsymbol{s}_V(\boldsymbol{G}_c \boldsymbol{x}_k; \hat{\boldsymbol{\theta}})\right).$$

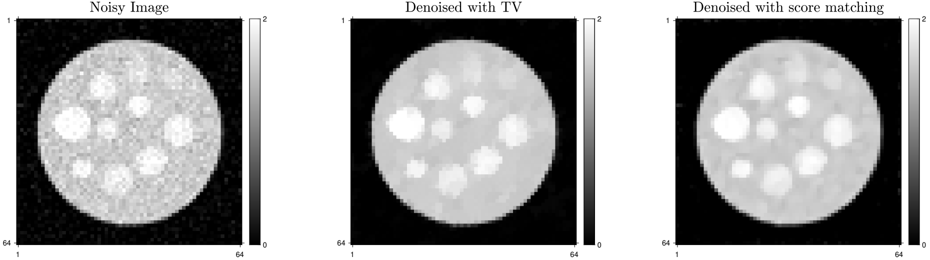
Generalizability to distribution shift? (pitfalls...)

Old:



PSNR 26.2 dB | PSNR 32.6 dB | PSNR 36.5 dB

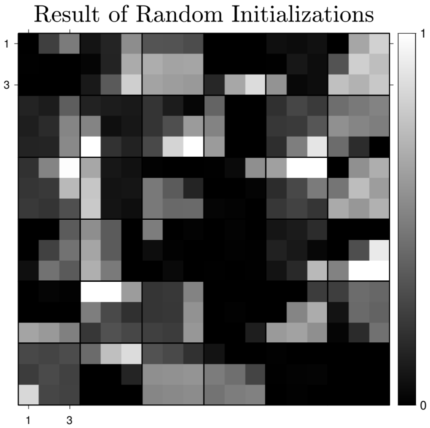
New:



PSNR 26.2 dB | PSNR 32.6 dB | PSNR 33.7 dB

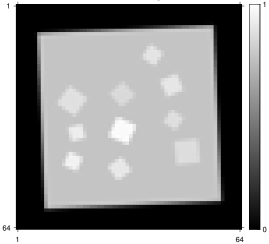
What changed?

MAP from random noise

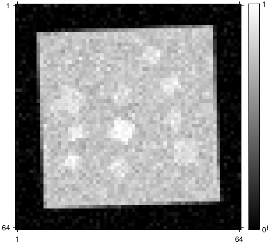


Distribution shift: rectangle test image

Clean Image

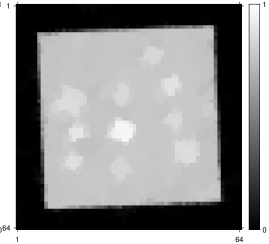


Noisy Image



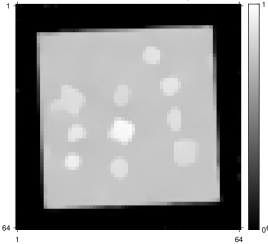
PSNR 26.0 dB

Denoised with TV



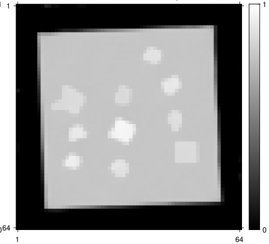
PSNR 33.2 dB

Score denoised, trained w/circles



PSNR 35.6 dB

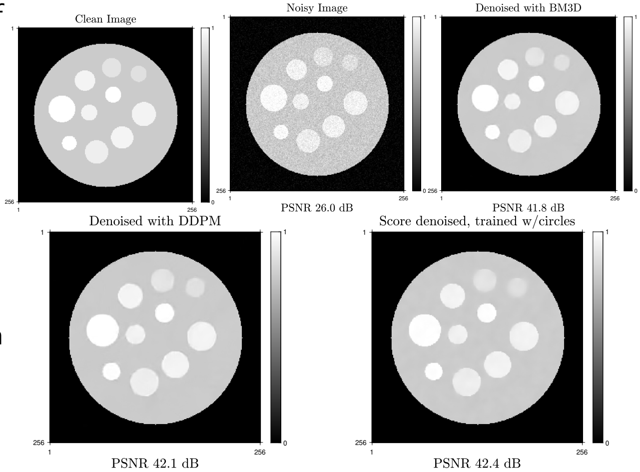
Score Denoised, Trained w/Rectangles



PSNR 37.7 dB

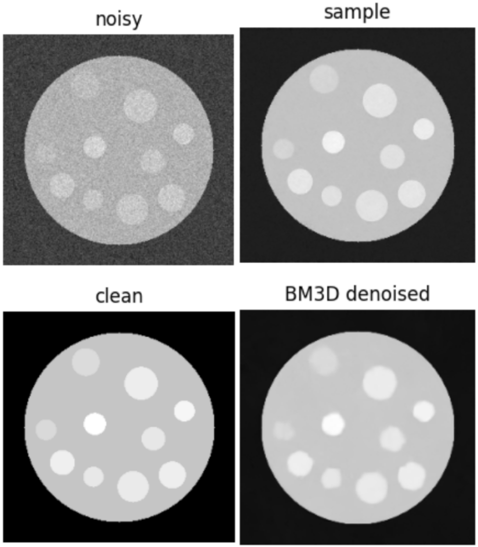
Whole-image vs patch models

- Whole-image diffusion model of Hu et al. (SPIE, 2022) [48]
- `https://github.com/DeweiHu/OCT_DDPM`
- Based on Ho et al. (NeurIPS, 2020) [49] denoising diffusion prob. model (DDPM)
- Trained with 1000 disk images.
- Tested with noisy disk phantom
- One sample from posterior



- Diffusion model of Hu et al. (SPIE, 2022) [48] trained with 3600 flower images.

- Tested with noisy disk phantom (PSNR 20.3 dB)
- One sample from posterior `https://github.com/DeweiHu/OCT_DDPM`



37.0 dB

35.3 dB

- Learning patch score models is feasible with denoising score matching
- Amplitude scale invariance is not inherent to score-based models
  Easily (?) fixed by patch normalization, but what other more subtle pitfalls exist?
- Integrate invariances: amplitude scale / rotation / flip / DC offset
- Compare with whole-image models:
  - "pure" CNN score models with small receptive fields
  - multi-scale score models [20, 21]
  - ...
- Explore trade-offs between generalizability and in-distribution performance
- Is the "optimal" patch size the whole image? (Even for 3D+T?)

Tutorial Julia code: `https://github.com/JeffFessler/ScoreMatching.jl`

Resources

Talk and code available online at
`http://web.eecs.umich.edu/~fessler`

[1] Z. Zhao, J. C. Ye, and Y. Bresler. "Generative models for inverse imaging problems: from mathematical foundations to physics-driven applications." In: *IEEE Sig. Proc. Mag.* 40.1 (2023), 148–63.

[2] E. D. Zhong, T. Bepler, B. Berger, and J. H. Davis. "CryoDRGN: reconstruction of heterogeneous cryo-EM structures using neural networks." In: *Nature Meth.* 18.2 (2021), 176–85.

[3] D. Rezende and S. Mohamed. "Variational inference with normalizing flows." In: *Proc. Intl. Conf. Mach. Learn.* 2015, 1530–8.

[4] F. Altekruger, A. Denker, P. Hagemann, J. Hertrich, P. Maass, and G. Steidl. "PatchNR: learning from very few images by patch normalizing flow regularization." In: *Inverse Prob.* 39.6 (May 2023), p. 064006.

[5] Z. Ramzi, B. Remy, F. Lanusse, J-L. Starck, and P. Ciuciu. "Denoising score-matching for uncertainty quantification in inverse problems." In: *NeurIPS 2020 Workshop on Deep Learning and Inverse Problems*. 2020.

[6] Y. Song, L. Shen, L. Xing, and S. Ermon. "Solving inverse problems in medical imaging with score-based generative models." In: *NeurIPS Deep Inv. Work.* 2021.

[7] Y. Song, L. Shen, L. Xing, and S. Ermon. "Solving inverse problems in medical imaging with score-based generative models." In: *Proc. Intl. Conf. on Learning Representations*. 2022.

[8] A. Jalal, M. Arvinte, G. Daras, E. Price, A. Dimakis, and J. Tamir. "Robust compressed sensing MR imaging with deep generative priors." In: *NeurIPS Workshop Deep Inverse*. 2021.

[9] H. Chung and J. C. Ye. "Score-based diffusion models for accelerated MRI." In: *Med. Im. Anal.* 80 (Aug. 2022), p. 102479.

[10] G. Luo, M. Blumenthal, M. Heide, and M. Uecker. "Bayesian MRI reconstruction with joint uncertainty estimation using diffusion models." In: *Mag. Res. Med.* (2023).

[11] A. Kazerouni, E. K. Aghdam, M. Heidari, R. Azad, M. Fayyaz, I. Hacihaliloglu, and D. Merhof. "Diffusion models in medical imaging: A comprehensive survey." In: *Med. Im. Anal.* 88 (Aug. 2023), p. 102846.

[12] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. "Score-based generative modeling through stochastic differential equations." In: *Proc. Intl. Conf. on Learning Representations*. 2021.

[13] A. Hyvärinen. "Estimation of non-normalized statistical models by score matching." In: *J. Mach. Learning Res.* 6.24 (2005), 695–709.

[14] P. Vincent. "A connection between score matching and denoising autoencoders." In: *Neural Comput.* 23.7 (July 2011), 1661–74.

[15] Y. Song and S. Ermon. "Generative modeling by estimating gradients of the data distribution." In: *NeurIPS*. 2019.

[16] Y. Song and S. Ermon. "Improved techniques for training score-based generative models." In: *NeurIPS*. Vol. 33. 2020, 12438–48.

[17] T. Salimans and J. Ho. "Progressive distillation for fast sampling of diffusion models." In: *Proc. Intl. Conf. on Learning Representations*. 2022.

[18] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever. *Consistency models*. 2023.

[19] H. Chung, S. Lee, and J. C. Ye. *Fast diffusion sampler for inverse problems by geometric decomposition*. 2023.

[20] F. Guth, S. Coste, V. D. Bortoli, and Stéphane Mallat. "Wavelet score-based generative modeling." In: *NeurIPS*. 2022.

[21] Z. Kadkhodaie, F. Guth, Stephane Mallat, and E. P. Simoncelli. "Learning multi-scale local conditional probability models of images." In: *Proc. Intl. Conf. on Learning Representations*. 2023.

[22] D. Ryu and J. C. Ye. *Pyramidal denoising diffusion probabilistic models*. 2022.

[23] S. Lee, H. Chung, J. Kim, and J. C. Ye. *Progressive deblurring of diffusion models for coarse-to-fine image synthesis*. 2022.

[24] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu. "DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps." In: *NeurIPS*. 2022.

[25] H. Chung, B. Sim, and J. C. Ye. "Come-closer-diffuse-faster: accelerating conditional diffusion models for inverse problems through stochastic contraction." In: *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition*. 2022, 12403–12.

[26] A. Vahdat, K. Kreis, and J. Kautz. "Score-based generative modeling in latent space." In: *NeurIPS*. 2021.

[27] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and Bjorn Ommer. "High-resolution image synthesis with latent diffusion models." In: *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition*. 2022, 10674–85.

[28] K. C. Tezcan, N. Karani, C. F. Baumgartner, and E. Konukoglu. "Sampling possible reconstructions of undersampled acquisitions in MR imaging with a deep learned prior." In: *IEEE Trans. Med. Imag.* 41.7 (July 2022), 1885–96.

[29] H. Chung, D. Ryu, M. T. McCann, M. L. Klasky, and J. C. Ye. *Solving 3D inverse problems using pre-trained 2D diffusion models*. 2022.

[30] S. Lee, H. Chung, M. Park, J. Park, W-S. Ryu, and J. C. Ye. *Improving 3D imaging with pre-trained perpendicular 2D diffusion models*. 2023.

[31] L. Bogensperger, D. Narnhofer, F. Ilic, and T. Pock. *Score-based generative models for medical image segmentation using signed distance functions*. 2023.

[32] W. Xia, W. Cong, and G. Wang. *Patch-based denoising diffusion probabilistic model for sparse-view CT reconstruction*. 2022.

[33] B. Levac, A. Jalal, and J. I. Tamir. *Accelerated motion correction for MRI using score-based generative models*. 2022.

[34] H. Chung, E. S. Lee, and J. C. Ye. "MR image denoising and super-resolution using regularized reverse diffusion." In: *IEEE Trans. Med. Imag.* (2023).

[35] S. Geman and D. Geman. "Stochastic relaxation, Gibbs distributions, and Bayesian restoration of images." In: *IEEE Trans. Patt. Anal. Mach. Int.* 6.6 (Nov. 1984), 721–41.

[36] A. J. Gray, J. W. Kay, and D. M. Titterington. "An empirical study of the simulation of various models used for images." In: *IEEE Trans. Patt. Anal. Mach. Int.* 16.5 (May 1994), 507–12.

[37] G. Wang, T. Luo, J-F. Nielsen, D. C. Noll, and J. A. Fessler. "B-spline parameterized joint optimization of reconstruction and k-space trajectories (BJORK) for accelerated 2D MRI." In: *IEEE Trans. Med. Imag.* 41.9 (Sept. 2022), 2318–30.

[38] W. Wu and K. L. Miller. "Image formation in diffusion MRI: A review of recent technical developments." In: *J. Mag. Res. Im.* 46.3 (Sept. 2017), 646–62.

[39] S. Bhadra, W. Zhou, and M. A. Anastasio. "Medical image reconstruction with image-adaptive priors learned by use of generative adversarial networks." In: *Proc. SPIE 11312 Medical Imaging: Phys. Med. Im.* 2020, p. 113120V.

[40] C. Li and M. Wand. "Precomputed real-time texture synthesis with Markovian generative adversarial networks." In: *Proc. European Comp. Vision Conf.* 2016, 702–16.

[41] P. Isola, J-Y. Zhu, T. Zhou, and A. A. Efros. "Image-to-image translation with conditional adversarial networks." In: *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition*. 2017, 5967–76.

[42] A. Elnekave and Y. Weiss. "Generating natural images with direct patch distributions matching." In: *Proc. European Comp. Vision Conf.* Vol. 13677. 2022.

[43] M. Aharon, M. Elad, and A. Bruckstein. "K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation." In: *IEEE Trans. Sig. Proc.* 54.11 (Nov. 2006), 4311–22.

[44] S. Ravishankar and Y. Bresler. "MR image reconstruction from highly undersampled k-space data by dictionary learning." In: *IEEE Trans. Med. Imag.* 30.5 (May 2011), 1028–41.

[45] J. Hertrich, A. Houdard, and C. Redenbach. "Wasserstein patch prior for image superresolution." In: *IEEE Trans. Computational Imaging* 8 (2022), 693–704.

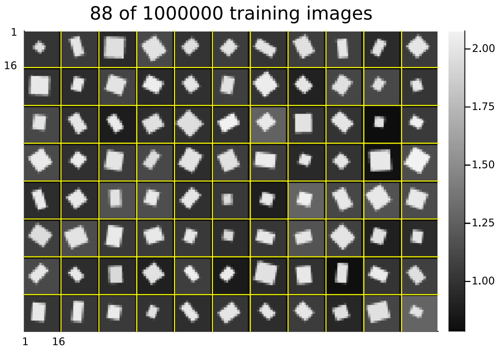
[46] F. Altekruger and J. Hertrich. "WPPNets and WPPFlows: The power of Wasserstein patch priors for superresolution." In: *SIAM J. Imaging Sci.* 16.3 (2023), 1033–67.

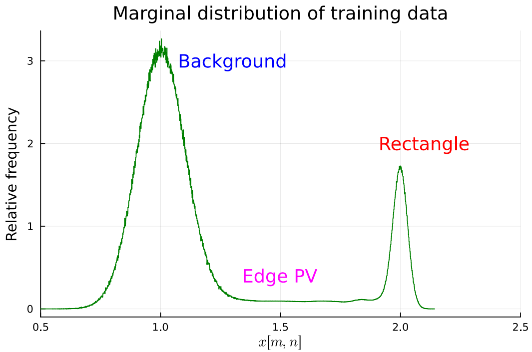
[47] G. E. Hinton. "Training products of experts by minimizing contrastive divergence." In: *Neural Computation* 14.8 (Aug. 2002), 1771–800.

[48] D. Hu, Y. K. Tao, and I. Oguz. "Unsupervised denoising of retinal OCT with diffusion probabilistic model." In: *Proc. SPIE 12032 Medical Imaging: Im. Proc.* 2022, p. 1203206.

[49] J. Ho, A. Jain, and P. Abbeel. "Denoising diffusion probabilistic models." In: *NeurIPS*. Vol. 33. 2020, 6840–51.

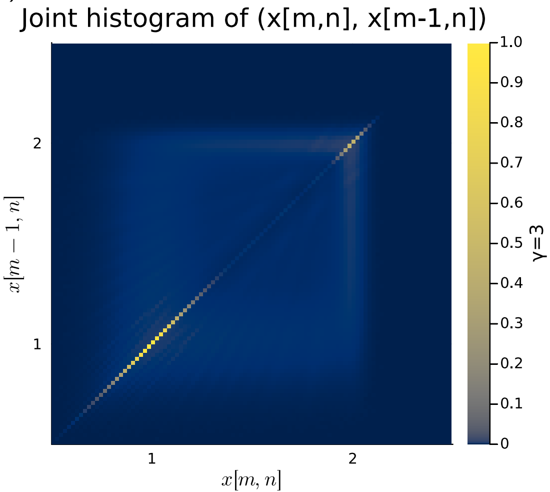
A simple exploration





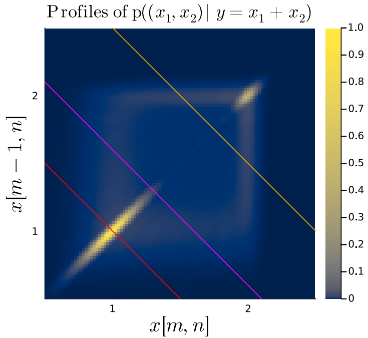
- Stochastic image model with random: center, width, orientation, background $\mathcal{N}(1, 0.1^2)$, rectangle foreground $\mathcal{N}(1, 0.03^2)$
- 10^6 training images of size 16×16 with partial volume effects.
- Data lies on 7-dimensional manifold.

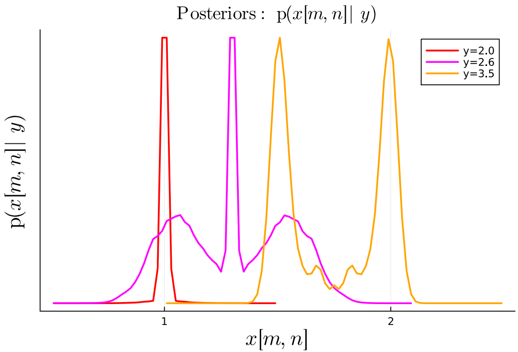
2×1 patches (cf TV)



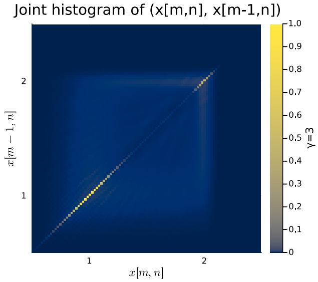
$$\mathrm{p}((x[m,n], x[m,n-1]) \mid y = x[m,n] + x[m,n-1])$$

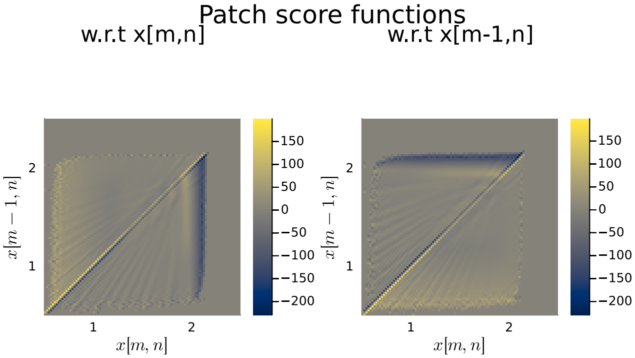
- MRI "center of k-space"
- MRI "2× acceleration





Patch statistics: score functions



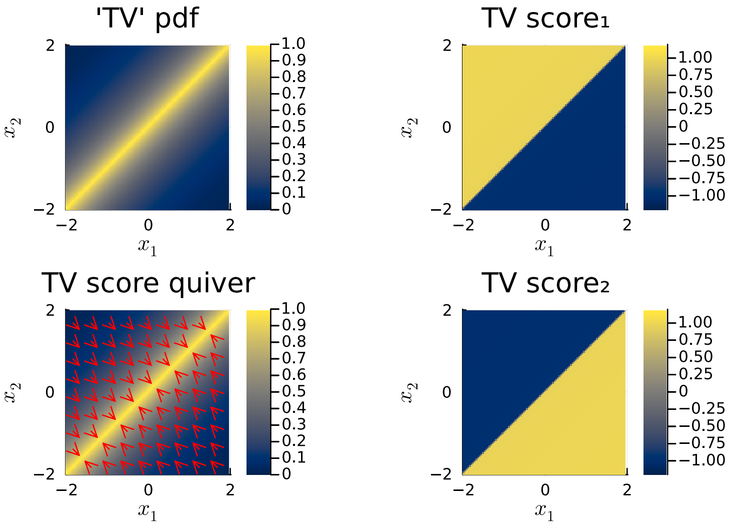


(Manifold data $\Longrightarrow$ score function $\boldsymbol{s}(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} \log \mathrm{p}(\boldsymbol{x})$ is not well-defined.)

“TV” score function

Total variation (TV) prior for 2×1 patch:

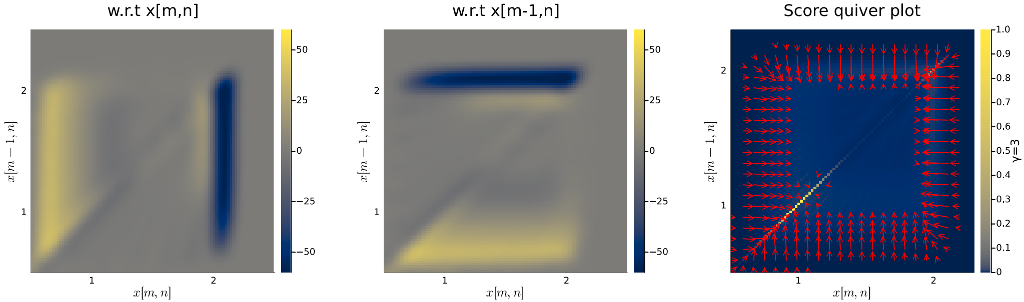
$$\mathrm{p}(\boldsymbol{x}) \propto \mathrm{e}^{-\beta |x_2 - x_1|}$$



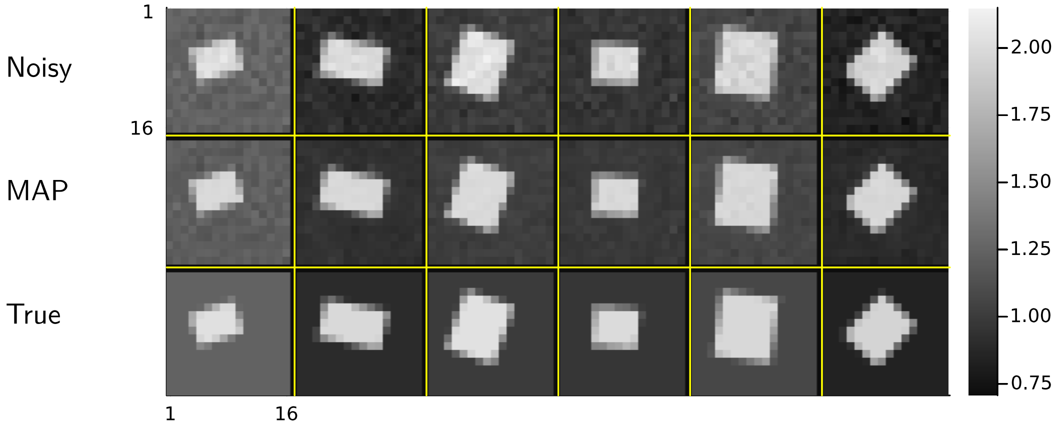
Smoothed score function

Following trends in score matching [12, 14]

Adding gaussian noise to training data $\equiv$ smoothing score function

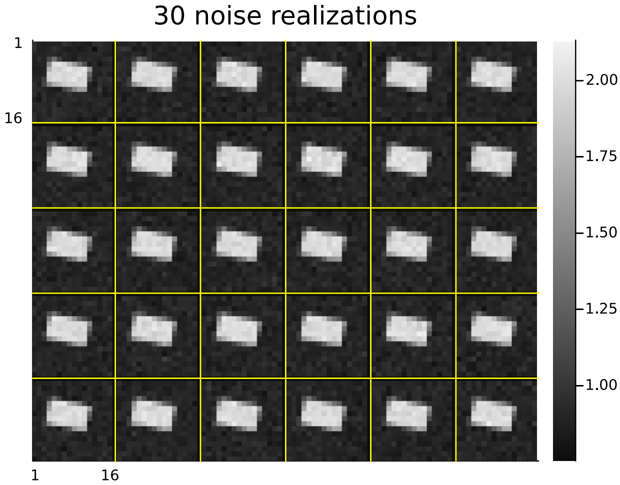


Noisy 29.5dB, MAP 29.9dB, True



Uncertainty?

- Sample from $\mathrm{p}(\boldsymbol{x}|\boldsymbol{y})$
- Perform multiple realizations



Multiple realizations