Bayesian inference in imaging inverse problems - Part 2

Marcelo Pereyra

Heriot-Watt University & Maxwell Institute for Mathematical Sciences

June 2025, Bologna, Italy.

Outline

Imaging inverse problems

- We are interested in an unknown image $x^\star \in \mathbb{R}^d$.
- We measure $y \in \mathrm{Y}$, related to $x^\star$ by some mathematical model.
- For example, in many imaging problems

$$y = Ax^\star + w,$$

for some operator A that is poorly conditioned or rank deficient, and an unknown perturbation or "noise" w.

- The recovery of $x^\star$ from y is usually not well posed. Additional information is required in order to deliver meaningful solutions.

Mathematical imaging frameworks

- There are three main mathematical and computational frameworks for inference in imaging inverse problems:
  1. Applied analysis
  2. Bayesian statistics.
  3. Machine learning.
- These frameworks have complementary strengths and weaknesses.
- Our aim is a unifying framework of theory, methods, and algorithms that inherits the benefits of each approach.

The Bayesian statistical approach

- Model $x^\star$ as a realisation of a r.v. $\mathbb{x}$ on $\mathbb{R}^d$. Use the distribution of $\mathbb{x}$ to regularise the problem and promote expected properties.

- The observation y is a realisation of a r.v. $(\mathbb{y}|\mathbb{x} = x^\star)$.

- Inferences about $x^\star$ from y are derived from the joint distribution of $(\mathbb{x}, \mathbb{y})$ - specified via the decomposition $p(x, y) = p(y|x)p(x)$.

- This determines the posterior distribution, with density

$$p(x|y) = \frac{p(y|x)p(x)}{\int_{\mathbb{R}^d} p(y|\tilde{x})p(\tilde{x})\mathrm{d}\tilde{x}},$$

which models our beliefs about $\mathbb{x}$ after observing $\mathbb{y} = y$.

Bayesian computation

- A simple algorithm to compute probabilities and expectations w.r.t. $p(x|y)$ is the Unadjusted Langevin Algorithm (ULA), given by

$$X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \delta_k \nabla \log p(X_k) + \sqrt{2\delta_k} Z_{k+1} \,,$$

  where $Z_{k+1} \sim \mathcal{N}(0, \mathrm{I}_d)$ and $(\delta_k)_{k\in\mathbb{N}}$ is a sequence of step-sizes.

- The samples generated by ULA can be used to compute Monte Carlo estimates of $\hat{x}_{MMSE}$ and perform advanced inferences.

- Recall that, given a set of samples $X_1, \ldots, X_M$ distributed according to $p(x|y)$, we approximate posterior expectations and probabilities

$$\bar{h} = \frac{1}{M} \sum_{m=1}^{M} h(X_m) \to \mathrm{E}\{h(x)|y\}, \quad \text{as } M \to \infty$$

Bayesian computation

- A stochastic gradient descent (SGD) to compute $\hat{x}_{MAP}$ is given by

$$X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \delta_k \nabla \log p(X_k) + \delta_k Z_{k+1}\,,$$

again, $Z_{k+1} \sim \mathcal{N}(0, \mathrm{I}_d)$ and $(\delta_k)_{k\in\mathbb{N}}$ is a sequence of step-sizes.

- Given a sequence of non-increasing weights $(\omega_k)_{k\in\mathbb{N}}$, we iteratively approximate $\hat{x}_{MAP}$ by

$$\bar{x}_{MAP} = \frac{\sum_{k=1}^{M} \omega_k X_k}{\sum_{k=1}^{M} \omega_k}\,.$$

- ULA and SGD are remarkably well understood and provably convergent under easily verifiable conditions on $p(x|y)$.

The SDE underpinning ULA and SGD

- Recall that ULA and SGD arise from discrete-time approximations of the Langevin diffusion process

$$\boldsymbol{X} : \quad \mathrm{d}\boldsymbol{X}_t = \frac{1}{2}\nabla \log p\left(\boldsymbol{X}_t|y\right)\mathrm{d}t + \mathrm{d}W_t, \quad 0 \leq t \leq T, \quad \boldsymbol{X}(0) = x_0.$$

  where W is the Brownian motion on $\mathbb{R}^d$.

- When $x \mapsto p(x|y) \in \mathcal{C}^1$ with $x \mapsto \nabla \log p(x|y)$ Lipschitz continuous, X_t converges exponentially fast to $p(x|y)$ as $t \to \infty$.

- ULA and SGD stem from a basic Euler approximations of $\boldsymbol{X}$.
- We recommend to use an accelerated approximation of $\boldsymbol{X}$ for significantly faster convergence (see 10.1137/19M1283719).

Proximal ULA and SGD computation

- When $U : x \mapsto -\log p(x)$ is convex but not Lipschitz differentiable (or has a poor Lipschitz constant), we use the proximal ULA and SGD:

$$X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \frac{\delta_k}{\lambda}\left(\mathrm{prox}_U^\lambda(X_k) - X_k\right) + \sqrt{2\delta_k} Z_{k+1}\,,$$

$$X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \frac{\delta_k}{\lambda}\left(\mathrm{prox}_U^\lambda(X_k) - X_k\right) + \delta_k Z_{k+1}\,.$$

where instead of $\nabla \log p(x)$ we evaluate the *proximal* operator

$$\mathrm{prox}_U^\lambda : x \mapsto \operatorname*{argmin}_{z \in \mathbb{R}^d} U(z) + \frac{1}{2\lambda}\|z - x\|_2^2\,.$$

- Proximal ULA and SGD target a regularised approximation of $p(x|y)$.

- $\lambda > 0$ controls an (asymptotic) bias vs. convergence speed trade-off.

The Plug & Play (PnP) approach

- PnP methods stem from the observation that

$$\mathrm{prox}_U^{\lambda} : x \mapsto \underset{x \in \mathbb{R}^d}{\operatorname{argmin}}\, U(z) + \frac{1}{2\lambda}\|z - x\|_2^2$$

can be viewed as a MAP *denoiser* to recover z from a noisy observation $x \sim \mathcal{N}(z, \lambda I)$, when z has marginal $p(z) \propto \exp\{-U(z)\}$.

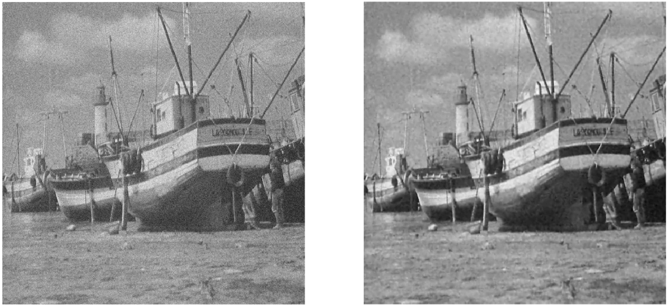


Figure: Image denoising with the proximal operator of the TGV pseudo-norm.

The Plug & Play (PnP) approach

- Instead of specifying U explicitly, PnP strategies "plug" a state-of-the-art denoiser $D_\epsilon : \mathbb{R} \mapsto \mathbb{R}$ in lieu of $\nabla \log p(x)$ or $\mathrm{prox}_U^\lambda(x)$ in an iterative sampling or optimisation.

- For example, in the context of ULA and SGD, one would consider

$$\mathrm{PnP\text{-}ULA}: \; X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \frac{\delta_k}{\epsilon}(D_\epsilon(X_k) - X_k) + \sqrt{2\delta_k} Z_{k+1} \, ,$$

and

$$\mathrm{PnP\text{-}SGD}: \; X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \frac{\delta_k}{\epsilon}(D_\epsilon(X_k) - X_k) + \delta_k Z_{k+1} \, .$$

- Given some training data $\{x_i'\}_{i=1}^M$, training a network to approximate an optimal MSE denoiser can deliver remarkable results. Why?!

Neural networks for denoising

- Consider a neural network $f_w : \mathbb{R} \mapsto \mathbb{R}$, parametrised by its weights and biases gathered in $w \in W$, where W is some measurable space.
- Let $\{x_i'\}_{i=1}^M$ be a training sample from a distribution with density $p(x)$.
- We generate $\{y_i'\}_{i=1}^M$ by contaminating $\{x_i'\}_{i=1}^M$ with Gaussian noise with mean zero and covariance $\epsilon\mathbb{I}$, i.e., $y_i' \sim \mathcal{N}(x_i', \epsilon\mathbb{I})$.
- The optimal MSE denoiser to recover $\{x_i'\}_{i=1}^M$ from $\{y_i'\}_{i=1}^M$ is the Bayesian estimator $\mathrm{E}(x|y)$ associated the prior $p(x)$.
- During training, when w is set such that

$$w^* = \underset{w \in W}{\operatorname{argmin}} \sum_{i=1}^{M} \| f_w(y_i') - x_i' \|_2^2$$

the resulting network f_{w^*} approximates the operator $y \mapsto \mathrm{E}(x|y)$.

This talk

Our aim here is to formalise the Bayesian perspective on inference with PnP priors and explore foundational questions, e.g.:

1. Under what conditions on D_ϵ are Bayesian PnP models well-posed and amenable to efficient computation? When do the key quantities of interest exist and inherit the well-posed nature of the model?
2. Can we guarantee the convergence of PnP-ULA and PnP-SGD under easily verifiable conditions, with non-asymptotic accuracy bounds?
3. Are these Bayesian PnP methods and algorithms delivering solutions that are meaningful from a non-subjective point of view?

We also present an alternative Bayesian strategy for inference with data-driven priors derived from generative models (e.g., VAEs, GANs, etc.).

This talk

For technical details please see:

1. R. Laumont, V. de Bortoli, A. Almansa, J. Delon, A. Durmus, and M. Pereyra, "Bayesian imaging using Plug and Play priors: when Langevin meets Tweedie", SIAM Journal on Imaging Sciences, 15 (2), 2022. `https://doi.org/10.1137/21M1406349`.

2. R. Laumont, V. de Bortoli, A. Almansa, J. Delon, A. Durmus, and M. Pereyra, "On Maximum-a-Posteriori estimation with Plug and Play priors and stochastic gradient descent", 2021. Preprint `https://hal.archives-ouvertes.fr/hal-03348735/`.

3. M. Holden, M. Pereyra, K. Zygalakis, "Bayesian Imaging with Data-Driven Priors Encoded by Neural Networks", SIAM Journal on Imaging Sciences, 15 (2), 2022. `https://doi.org/10.1137/21M1406313`.

Outline

The oracle Bayesian model

We analyse Bayesian models with data-driven priors in an *M-complete* modelling framework:

- There exists a true - albeit unknown - marginal distribution for $\mathbb{x}$ and posterior distribution for $(\mathbb{x}|\mathbb{y} = y)$.

- Basing inferences on these oracle models is theoretically optimal.

- We henceforth denote this optimal prior distribution by μ. When μ admits a density w.r.t. the Leb. measure on $\mathbb{R}^d$, we denote it by $p^\star$.

- In that case, the posterior for $x|y$ has density

$$p^\star(x|y) = \frac{p(y|x)p^\star(x)}{\int_{\mathbb{R}^d} p(y|\tilde{x})p^\star(\tilde{x})\mathrm{d}\tilde{x}} \,.$$

The oracle Bayesian model

We analyse Bayesian models with data-driven priors in an *M-complete* modelling framework:

- In this conceptual construction, μ naturally depends on the application.
- In problems where there is training data $\{x_i'\}_{i=1}^M$ available, we regard $\{x_i'\}_{i=1}^M$ as a sample from μ.
- For presentation simplicity, we henceforth assume that $p^\star$ exists. However, our results hold even this is not the case.
- This is important to provide robustness to situations where $p^\star$ is nearly degenerate or improper (e.g., manifold hypothesis).

The oracle Bayesian model

- We cannot verify that $p^\star$ is proper and differentiable, with $x \mapsto \nabla \log p^\star(x|y)$ Lipschitz. Problematic for ULA and SGD.

- Let $\epsilon > 0$. To guarantee that gradient algorithms are sensible, we introduce the regularised approximation μ_ϵ of μ with density

$$p_\epsilon^\star(x) = (2\pi\epsilon)^{-d/2} \int_{\mathbb{R}^d} \exp\left[-\|x - \tilde{x}\|_2^2/(2\epsilon)\right] p^\star(\tilde{x}) \mathrm{d}\tilde{x}\,.$$

- By involving the likelihood $p(y|x)$, we derive the regularised posterior

$$p_\epsilon^\star(x|y) = \frac{p(y|x) p_\epsilon^\star(x)}{\int_{\mathbb{R}^d} p(y|\tilde{x}) p_\epsilon^\star(\tilde{x}) \mathrm{d}\tilde{x}}\,.$$

The oracle Bayesian model

Laumont et al. (2021a) establishes that, under mild and easily verifiable conditions on $p(y|x)$, the following holds:

1. The regularised densities $p^\star_\epsilon(x)$ and $p^\star_\epsilon(x|y)$ are proper and smooth, but not necessarily Lipschitz differentiable.
2. The approximation error w.r.t. the oracle models $p^\star(x)$ and posterior $p^\star(x|y)$ is controlled by ϵ and vanishes as $\epsilon \to 0$.

Guaranteeing that $x \mapsto \nabla \log p^\star_\epsilon(x|y)$ is Lipschitz continuous requires an additional assumption.

The oracle MMSE denoiser

- To study $\nabla \log p_\epsilon^\star(x|y)$ and rigorously derive PnP Bayesian methods, we introduce the oracle MMSE denoiser:

$$D_\epsilon^\star(x) = (2\pi\epsilon)^{-d/2} \int_{\mathbb{R}^d} \tilde{x} \exp\left[-\|x-\tilde{x}\|^2/(2\epsilon)\right] p^\star(\tilde{x}) \mathrm{d}\tilde{x}\,,$$

to recover an image $x \sim \mu$ from a noisy observation $x_\epsilon \sim \mathcal{N}(x, \epsilon \mathbb{I}_d)$.

- From Tweedie's identity, the gradient

$$\epsilon \nabla \log p_\epsilon^\star(x) = D_\epsilon^\star(x) - x\,.$$

- Laumont et al. (2021a) establishes that $\nabla \log p_\epsilon^\star(x)$ is Lipschitz when the denoiser $D_\epsilon^\star$ is guaranteed to achieve a finite MSE. This is a natural assumption given that $D_\epsilon^\star$ is the MMSE Bayesian estimator.

The oracle Bayesian model

- Moreover, under mild assumptions on the likelihood $p(y|x)$, the posterior $p^\star_\epsilon(x|y)$ is well posed in the sense of Hadamard.
- This implies that key quantities computed from $p^\star_\epsilon(x|y)$ are stable w.r.t perturbations of y.
- We can then easily establish the convergence of gradient-based computation algorithms for $p^\star_\epsilon(x|y)$ based on oracle denoiser $D^\star_\epsilon$.

Outline

PnP Bayesian computation algorithms

- Generic denoisers D_ϵ, such as neural networks, are not usually gradient mappings.
- As a result, PnP-ULA and PnP-SGD algorithms implemented with D_ϵ are not related to gradient flows. This makes their analysis difficult.
- We focus on deep neural network denoisers that are, by construction, Lipschitz continuous, and which seek to approximate $D^\star_\epsilon$.
- D_ϵ's Lipschitz constant is controlled during training by spectral normalisation.
- We establish convergence results and characterise accuracy w.r.t. the oracle models - the key factor is how well D_ϵ approximates $D^\star_\epsilon$.

PnP Unadjusted Langevin Algorithm

- First, Laumont et al. (2021a) establishes that PnP-ULA below convergences geometrically fast to a neighbourhood of $p^\star_\epsilon(x|y)$:

$$\begin{aligned}\mathrm{PnP\text{-}ULA}: \ X_{k+1} = X_k &+ \delta_k \nabla \log p(y|X_k) + \frac{\delta_k}{\epsilon}(D_\epsilon(X_k) - X_k) \\ &+ \frac{\delta_k}{\lambda}(\Pi_{\mathrm{C}}(X_k) - X_k) + \sqrt{2\delta_k} Z_{k+1}\,.\end{aligned}$$

  $\Pi(\cdot)$ is the projection operator, $\mathrm{C} \subset \mathbb{R}^d$ is any large compact set, and λ controls the target's far-tail behaviour.

- The accuracy depends on the magnitude of the error between $D^\star_\epsilon$ and D_ϵ within an $0, R$-ℓ_2 ball, as well as on the algorithm parameters.
- Under additional assumptions, we can establish convergence to a neighbourhood of $p^\star(x|y)$.

PnP Stochastic Gradient Descent

- Laumont et al. (2021b) establishes that any stable sequence

$$\mathrm{PnP\text{-}SGD}: \; X_{k+1} = X_k + \delta_k \nabla \log p(y|X_k) + \frac{\delta_k}{\epsilon}(D_\epsilon(X_k) - X_k) + \delta_k Z_{k+1}\,,$$

  converges to a neighbourhood of the set of critical points

$$S = \{x : \nabla \log p_\epsilon^\star(x|y) = 0)\}\,.$$

- Again, the accuracy depends on the magnitude of the error between $D_\epsilon^\star$ and D_ϵ, as well as on the algorithm parameters.

Outline

Problem setup

- Forward model: $y = Ax + w$ where $w \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$ with $\sigma = 1/255$.
- Deblurring: A encodes a box filter of size 9×9 pixels.
- Inpainting: A is a mask operator hidding 80% of the image pixels.
- Clean images:

Simpson.

Cameraman.

Traffic.

Algorithm parameters

- Comparison with the provably convergent PnP-ADMM algorithm of

  E. K. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, *Plug-and-play methods provably converge with properly trained denoisers*, in Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, 2019, pp. 5546–5557, http://proceedings.mlr.press/v97/ryu19a.html, https://arxiv.org/abs/1905.05406.

  where D_ϵ is a spectrally normalised DnCNN (neural network) denoiser trained such that $(D_\epsilon - Id)$ is L-Lipschitz with $\mathrm{L} < 1$, with $\epsilon_{deb.} = (5/255)^2$ and $\epsilon_{inp.} = (40/255)^2$.

- For meaningful comparison, we use the same denoiser and set $\epsilon = (5/255)^2$, $C = [-1, 2]^d$, and λ for geometric convergence.

- MC estimates calculated from a 1-in-2500 thinned ULA Markov chain.

	n	$n_{burn-in}$	δ	Initialization
PnP-ULA	2.5e7	2.5e6	$\delta_{max}/3$	y

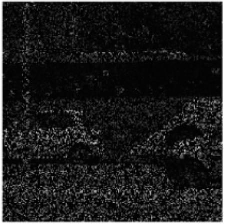
(a) Blurry Simpson, (PSNR=22.44).

(b) Blurry Cameraman, (PSNR=20.30).

(c) Blurry Traffic, (PSNR=20.34).

(a) Simpson image with 80% missing pixels (PSNR=7.45).

(b) Cameraman with 80% missing pixels (PSNR=6.67).



(c) Traffic with 80% missing pixels (PSNR=8.35).

Deblurring estimation results

PSNR=34.24, SSIM= 0.94

PSNR=32.56, SSIM=0.92

PSNR=32.48, SSIM=0.93

PSNR=30.37, **SSIM=0.93**

PSNR=29.47, SSIM=0.91

PSNR=30.81, SSIM=0.89

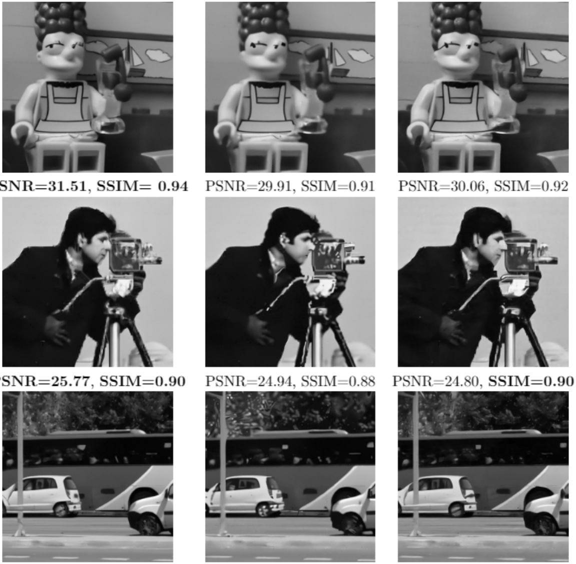
PSNR=29.86, SSIM=0.89

PSNR=28.13, SSIM=0.84

PSNR=29.44, SSIM=0.87

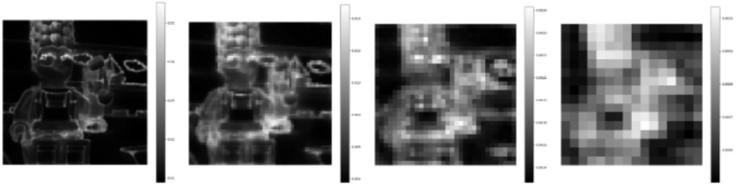
PnP-ULA | PnP-SGD | PnP-ADMM

Inpainting estimation results

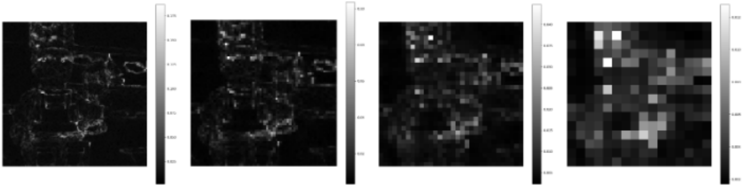


PnP-ULA	PnP-SGD	PnP-ADMM
PSNR=31.51, SSIM= 0.94	PSNR=29.91, SSIM=0.91	PSNR=30.06, SSIM=0.92
PSNR=25.77, SSIM=0.90	PSNR=24.94, SSIM=0.88	PSNR=24.80, **SSIM=0.90**
PSNR=27.02, SSIM=0.85	PSNR=25.63, SSIM=0.81	PSNR=26.46, SSIM=0.84

Uncertainty visualisation (deblurring & Inpainting)



(a) Scale 1. (b) Scale 2. (c) Scale 3. (d) Scale 4.



(a) Scale 1. (b) Scale 2. (c) Scale 3. (d) Scale 4.

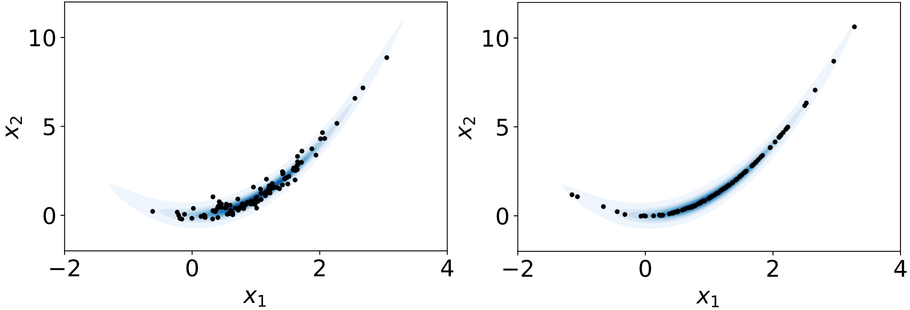
Outline

Generative modelling

We now focus on an alternative to PnP Bayesian inference based on deep generative models (e.g., VAEs, GANs). Again, our aim is to explore foundational questions and demonstrate the approach:

1. Again, let $\{x_i'\}_{i=1}^M$ be a training sample from the true prior μ.
2. We adopt a manifold hypothesis and suppose that $\mathbb{x}$ takes values close to an unknown p-dimensional submanifold of $\mathbb{R}^d$.
3. To estimate the manifold, we introduce a latent r.v. $\mathbb{z}$ on $\mathbb{R}^p$, with $p \ll d$, and a mapping $\nu_\theta : \mathbb{R}^p \mapsto \mathbb{R}^d$, such that the push-forward measure of $\mathbb{z} \sim \mathcal{N}(0, \mathrm{I}_p)$ under ν_θ is close to $\{x_i'\}_{i=1}^M$ (in dist.).
4. We implement ν_θ as a neural network. We can learn ν_θ from $\{x_i'\}_{i=1}^M$ by using, e.g., a VAE, a GAN, or a flow approach. We use VAEs.

Toy example - Rosenbrock distribution



Left: training data from the two-dimensional Rosenbrock distribution. Right: push-forward of $\mathbb{z} \sim \mathcal{N}(0, \mathrm{I}_p)$ under ν_θ as implemented by a VAE, with $p = 1$.

Posterior distributions for generative priors

- With $\mathbb{z}$ and ν_θ, we have the likelihood $p(y|z) = p(y|x = \nu_\theta(z))$.
- We use Bayes' theorem to derive the posterior for $\mathbb{z}|\mathbb{y} = y$

$$p(z|y) = \frac{p(y|x = \nu_\theta(z))p(z)}{\int_{\mathbb{R}^p} p(y|\tilde{z})p(\tilde{z})\mathrm{d}\tilde{z}},$$

- Pushing $(\mathbb{z}|\mathbb{y} = y)$ under $\nu_\theta(z)$ leads to the posterior for $(\mathbb{x}|\mathbb{y} = y)$, which supported on a manifold and does not have a density.
- This provides a different approximation to the oracle $p^\star(x|y)$.
- Holden et al. (2022) establishes that $(\mathbb{z}|\mathbb{y} = y)$ and $(\mathbb{x}|\mathbb{y} = y)$ are well-posed in the sense of Hadamard and have finite moments.

Illustrative experiments

- We illustrate the proposed approach with the MNIST dataset.
- We perform the following advanced inferences:
  1. Identify the latent dimension p.
  2. Perform MMSE inference in challenging image denoising, inpainting, and deblurring experiments.
  3. Adopt a likelihood-ratio test based on the statistic $\log p(y)$ to detect out-of-sample observations that should not be analysed with the Bayesian model.
  4. Assess the frequentist accuracy of the Bayesian probabilities reported by the model.
- We report comparisons with MAP estimation under the same model, and with PnP-ADMM by using a DnCNN denoiser.

Identification of manifold dimension p

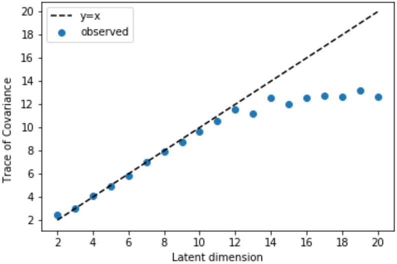
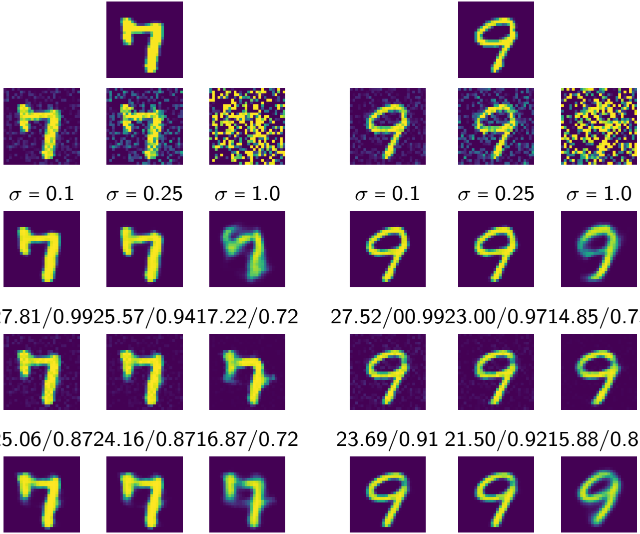
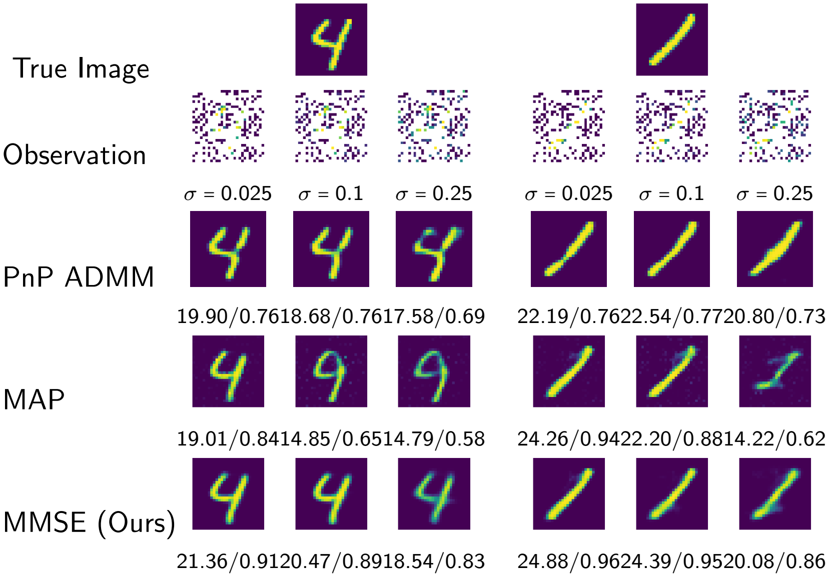


Figure: Trace of sample covariance of $\nu_\theta(x_i)$ across all test images. The amount of information encoded by the prior stabilises for $p \approx 12$, additional dimensions do not significantly increase the amount of prior information encoded .

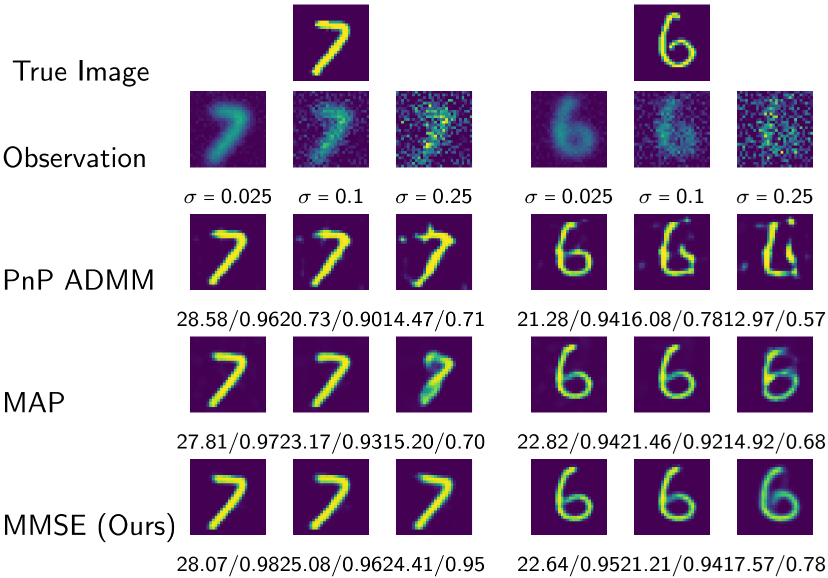
Denoising


True Image
Observation
$\sigma = 0.1$ $\sigma = 0.25$ $\sigma = 1.0$ $\sigma = 0.1$ $\sigma = 0.25$ $\sigma = 1.0$
NNET
27.81/0.99 25.57/0.94 17.22/0.72 27.52/00.99 23.00/0.97 14.85/0.75
MAP
25.06/0.87 24.16/0.87 16.87/0.72 23.69/0.91 21.50/0.92 15.88/0.81
MMSE (Ours)
24.99/0.92 23.99/0.91 17.58/0.82 21.94/0.96 21.85/0.96 18.74/0.81

Inpainting


True Image
Observation
σ = 0.025
σ = 0.1
σ = 0.25
σ = 0.025
σ = 0.1
σ = 0.25
PnP ADMM
19.90/0.7618.68/0.7617.58/0.69
22.19/0.7622.54/0.7720.80/0.73
MAP
19.01/0.8414.85/0.6514.79/0.58
24.26/0.9422.20/0.8814.22/0.62
MMSE (Ours)
21.36/0.9120.47/0.8918.54/0.83
24.88/0.9624.39/0.9520.08/0.86

Deconvolution



Likelihood ratio test for out-of-distribution detection

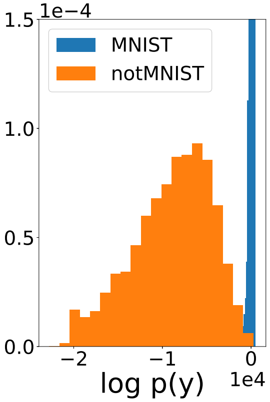


Figure: Denoising

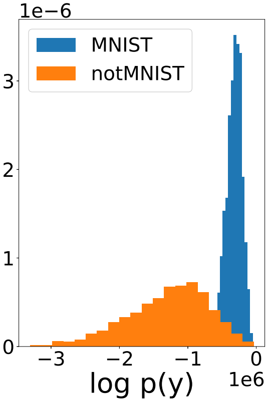


Figure: Inpainting

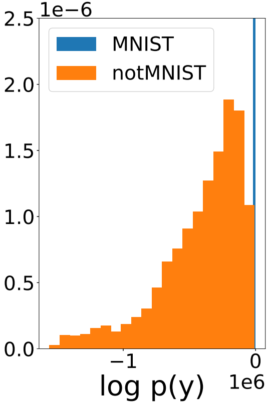


Figure: Deblurring

Figure: Histograms of marginal likelihoods for image denoising, inpainting and deblurring experiments. Out-of-sample detection powers for notMNIST of 99.6%, 88.5% and 99.8% respectively.

Coverage test: frequentist accur. of Bayesian probabilities

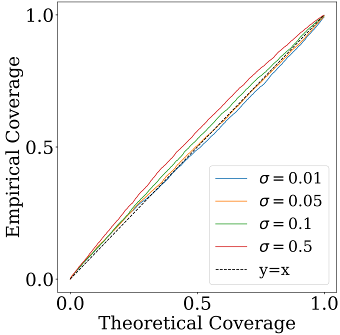


Figure: Denoising

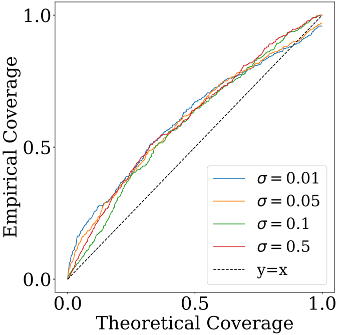


Figure: Inpainting

Outline

Conditional generative priors

- Despite their success in computer vision, scaling generative models to large inference problems reliably is difficult because of mode collapse, spurious modes, or other sources of bias.

- To reduce the difficulty of the machine learning problem, we consider a conditional generative model $\mathbb{x} = \nu_\theta^u(\mathbb{z})$, $\mathbb{z} \sim \mathcal{N}(0, \mathrm{I}_p)$, that models the distribution of $\mathbb{x}$ given some additional r.v. $\mathbb{u}$.

- For this construction to be useful, $\mathbb{u}$ should have low uncertainty given y.

Conditional generative priors

- For example, we let $\mathbb{u}$ denote a low resolution version of $\mathbb{x}$, and implement ν_θ^u by using a normalising flow for image super-resolution.

- This leads to the model

$$p(z|y,u) = \frac{p(y|z,u)p(z)}{p(y|u)}\,,$$

with $p(y|z,u) = p(y|x = \nu_\theta^u(z))$ and $p(y|u) = \int_{\mathbb{R}^p} p(y|\tilde{z},u)p(\tilde{z})\mathrm{d}\tilde{z}$.

Empirical Bayesian imaging with conditional generative priors

- We accurately estimate $u^\star$ from y by maximum marginal likelihood estimation:

$$\hat{u} = \underset{\mu}{\operatorname{argmax}}\, p_\theta(y|u)\,.$$

- Adopting an empirical Bayesian strategy, we perform inference on $(\mathbb{x}|\mathbb{y} = y, \mathbb{u} = \hat{u})$ by using

$$p(z|y,\hat{u}) = \frac{p(y|z,\hat{u})p(z)}{p(y|\hat{u})}\,,$$

and pushing $(\mathbb{z}|\mathbb{y} = y, \mathbb{u} = \hat{u})$ to $(\mathbb{x}|\mathbb{y} = y, \mathbb{u} = \hat{u})$ by using ν_θ^u.

Bayesian computation

- A simple algorithm to compute $\hat{u}$ probabilities and expectations w.r.t. $p(z|y,\hat{u})$ is the Stochastic Approximation Proximal Gradient scheme

$$Z_{k+1} = Z_k + \delta_k \nabla_z \log p(y|Z_k, u_k) + \delta_k \nabla_z \log p(Z_k) + \sqrt{2\delta_k} Z_{k+1}\,,$$

  and

$$u_{k+1} = \Pi_U\left[u_k + \gamma_k \nabla_u \log p(Z_{k+1}|y, u_k)\right],$$

  where $Z_{k+1} \sim \mathcal{N}(0, \mathrm{I}_d)$, $(\delta_k)_{k\in\mathbb{N}}$ and $(\gamma_k)_{k\in\mathbb{N}}$ are sequences of step-sizes, and Π_U denotes the Euclidean projection onto the set of admissible values for u.

- This SAPG is reasonably well understood and provably convergent under easily verifiable conditions on $p(z|y)$. See, e.g., `https://doi.org/10.1007/s11222-020-09986-y` and `https://doi.org/10.1137/20M1339829` for details.

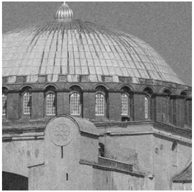
We compare the MMSE estimator obtained with the proposed method and the MAP-style estimator obtained from PnP-ADMM, by using a DnCNN end-to-end denoising prior, on an image deblurring problem.

$x^\star$ | y | PnP-ADMM | Proposed

Figure: Image deblurring experiment: 9×9 uniform blur, $\sigma = 10$.

Illustrative example - Image pan-sharpening

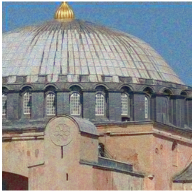
We seek to recover $x^\star$ from two noisy linear observations y_1 and y_2, one with spectral fine details and the other with spatial fine detail.



y_1

y_2

Proposed (31.5dB)



PnP ADMM (28.5dB)

$x^\star$

Outline

Conclusion

- We have studied theory, methods, and algorithms for performing Bayesian inference with data-driven priors encoded by neural networks.
- We rooted our discussion in the Bayesian *M-complete* paradigm that views PnP models as approximations of a regularised oracle model.
- We have considered PnP and generative strategies and established that the Bayesian models involved are well-posed under mild assumptions on the likelihood.
- These conditions are satisfied by Gaussian linear observation models, for example.

Conclusion

- We studied the PnP ULA and SGD algorithms and provided detailed convergence guarantees under easily verifiable and realistic conditions.
- The provided theory does not require the denoiser to be a maximally monotone operator, e.g., to be a gradient or proximal operator.
- We also studied the estimation error involved in using implementable PnP algorithms instead of the oracle model.
- Bayesian model with generative priors rely on MCMC computation on the latent space, and on automatic differentiation to compute gradients. Standard ULA results apply.

Thank you!