Diffusion Posterior Sampling for General Noisy Inverse Problems

Sep, 2023

Inverse Problem

Recover ground truth signal (image) $\boldsymbol{x} \in R^n$ from noisy measurements $\boldsymbol{y} \in R^m$

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{n} \tag{1}$$

- Forward operator: $\mathcal{A}(\cdot)$
- Noise: $\boldsymbol{n} \in R^m$
- $m < n$

Inverse Problem

Recover ground truth signal (image) $\boldsymbol{x} \in R^n$ from noisy measurements $\boldsymbol{y} \in R^m$

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{n}$$

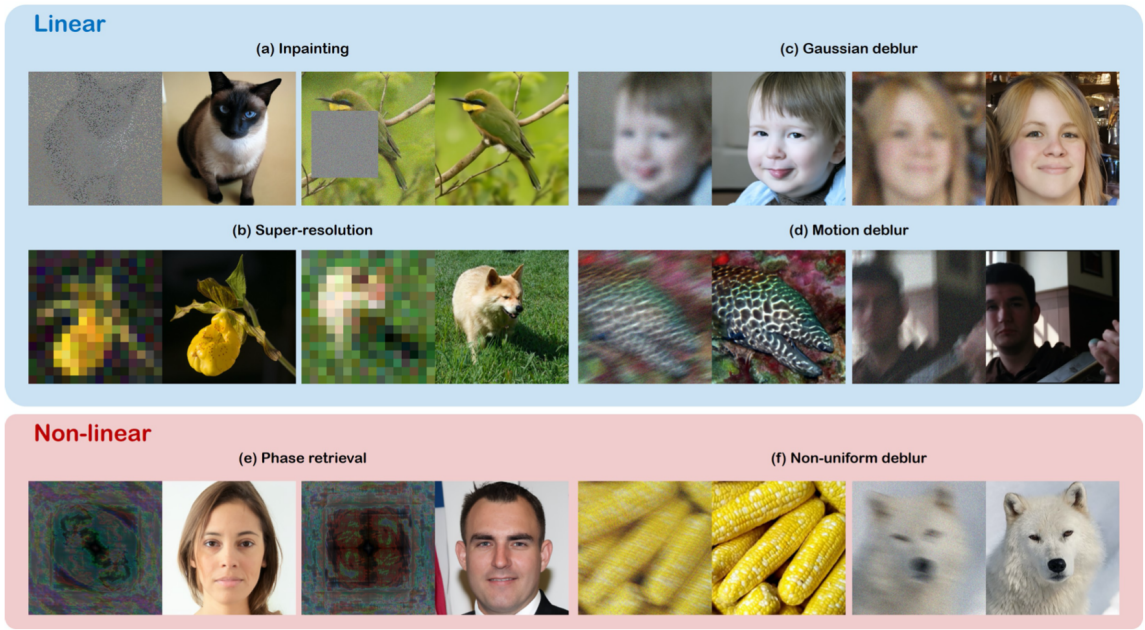

Figure: Illustration of Inverse Problems

Inverse Problem

In the linear case, the forward process can be simplified to matrix multiplication:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n}$$

- For inpainting, matrix $\boldsymbol{A}$ is a diagonal matrix with 1's and 0's
- For gaussian blurring, matrix $\boldsymbol{A}$ is a circulant matrix
- For compressed sensing MRI, matrix $\boldsymbol{A}$ is a Fourier transform matrix

Inverse Problem

Challenges in solving inverse problems:
Recovering ground truth signal (image) $\boldsymbol{x} \in R^n$ from noisy measurements $\boldsymbol{y} \in R^m$, where $\boldsymbol{m} < \boldsymbol{n}$ is an ill-posed problem
Solution: utilizing prior knowledge to regulate the reconstructions:
By Byes rule, we have:

$$p(\boldsymbol{x}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})$$

- Likelihood: $p(\boldsymbol{y}|\boldsymbol{x})$
- Prior: $p(\boldsymbol{x})$

Inverse Problem

Solution: utilizing prior knowledge to regulate the reconstructions:

$$\min_{\boldsymbol{x}} ||\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})||_2^2 + \lambda \mathcal{R}(\boldsymbol{x})$$

1. Example 1:

$$\min_{\boldsymbol{x}} ||\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})||_2^2 + \lambda TV(\boldsymbol{x})$$

, where $TV(\boldsymbol{x}) = \sum_{i=1}^{n-1} |\boldsymbol{x}_i - \boldsymbol{x}_{i+1}|$

2. Example 2:

$$\min_{\boldsymbol{z}} ||\mathcal{A}(\boldsymbol{D}\boldsymbol{z}) - \boldsymbol{y}||_2^2 + \lambda ||\boldsymbol{z}||_1$$

, where $\boldsymbol{D}$ can be DCT basis, Wavelet basis,...

Inverse problem

How can we use a pre-trained generative model as prior?

- CSGM: Compressed Sensing with Generative Model

$$\min_{\boldsymbol{z}} ||\mathcal{A}(G(\boldsymbol{z})) - \boldsymbol{y}||_2^2 + ||\boldsymbol{z}||_2^2$$

  , where G is is a pre-trained generative model such as GAN or VAE

- Question: How to utilize diffusion model as prior?

Score-Based Diffusion Models

Diffusion models define the generative process as the *reverse* of the noising process $\boldsymbol{x}(t), t \in [0, T]$.

1. $\boldsymbol{x}(0) \sim p_{data}, \boldsymbol{x}(T) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2. Forward process:

$$d\boldsymbol{x} = -\frac{\beta(t)}{2}\boldsymbol{x}dt + \sqrt{\beta(t)}d\boldsymbol{w} \tag{2}$$

3. Reverse process:

$$d\boldsymbol{x} = [-\frac{\beta(t)}{2}\boldsymbol{x} - \beta(t)\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)]dt + \sqrt{\beta(t)}d\boldsymbol{w} \tag{3}$$

4. Approximate $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$ with $\boldsymbol{s}_{\theta^*}(\boldsymbol{x}_t, t)$,

$$\theta^* = argmin_\theta E[||\boldsymbol{s}_\theta(\boldsymbol{x}(t), t) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}(t)|x(0))||_2^2] \tag{4}$$

Score-Based Diffusion Models

In the Discrete Setting:

- Forward Process:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \tag{5}$$

- Reverse process:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) := \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \Sigma_\theta(\boldsymbol{x}_t, t)) \tag{6}$$

- Approximate $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ with $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}; \tilde{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I}) \tag{7}$$

- Training objective:

$$\min_\theta E_{t,\boldsymbol{x}_0} ||\tilde{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)||_2 \tag{8}$$

Frame Title

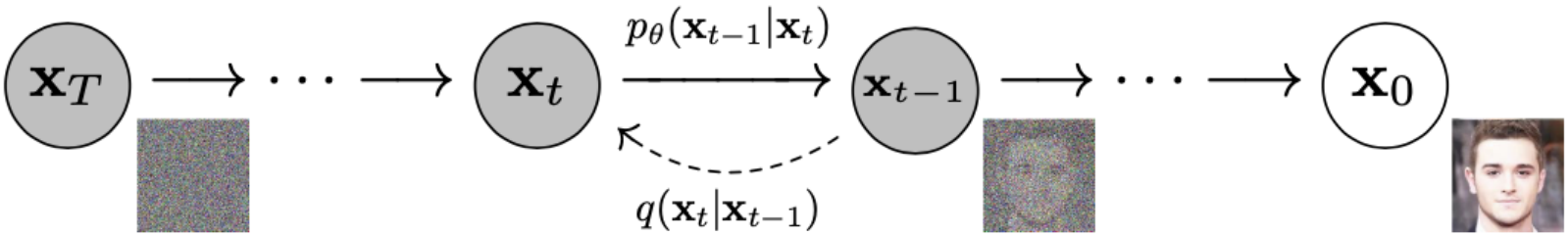

Figure: Directed Graphical Model for Diffusion Model

Solving Inverse Problem with Diffusion Model

In the Discrete Setting:

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Figure: DDPM Training and Sampling

Solving Inverse Problem with Diffusion Model

Question: how to utilize diffusion models for inverse problem solving?
SR3, an End-to-End approach:

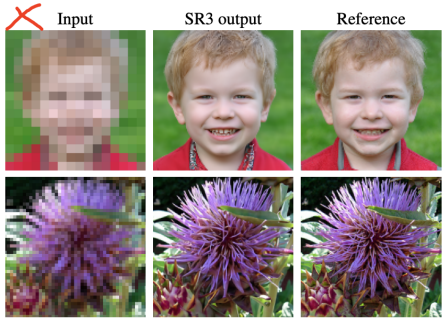

Figure: SR3

Algorithm 1 Training a denoising model f_θ

1: **repeat**
2: $(\boldsymbol{x}, \boldsymbol{y}_0) \sim p(\boldsymbol{x}, \boldsymbol{y})$
3: $\gamma \sim p(\gamma)$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take a gradient descent step on
$\nabla_\theta \left\| f_\theta(\boldsymbol{x}, \sqrt{\gamma}\boldsymbol{y}_0 + \sqrt{1-\gamma}\boldsymbol{\epsilon}, \gamma) - \boldsymbol{\epsilon} \right\|_p^p$
6: **until** converged

Algorithm 2 Inference in T iterative refinement steps

1: $\boldsymbol{y}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\boldsymbol{z} = \mathbf{0}$
4: $\boldsymbol{y}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\boldsymbol{y}_t - \frac{1-\alpha_t}{\sqrt{1-\gamma_t}} f_\theta(\boldsymbol{x}, \boldsymbol{y}_t, \gamma_t) \right) + \sqrt{1-\alpha_t}\boldsymbol{z}$
5: **end for**
6: **return** $\boldsymbol{y}_0$

Solving Inverse Problem with Diffusion Model

Unsupervised methods are more desirable

1. Unconditional sampling:

$$d\boldsymbol{x} = [-\frac{\beta(t)}{2}\boldsymbol{x} - \beta(t)\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t)]dt + \sqrt{\beta(t)}d\boldsymbol{w}$$

2. Conditional sampling given measurements $\boldsymbol{y}$:

$$d\boldsymbol{x} = [-\frac{\beta(t)}{2}\boldsymbol{x} - \beta(t)\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t|\boldsymbol{y})]dt + \sqrt{\beta(t)}d\boldsymbol{w} \quad (9)$$

3. Apply Bayes rule:

$$d\boldsymbol{x} = [-\frac{\beta(t)}{2}\boldsymbol{x} - \beta(t)(\nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{x}_t) + \nabla_{\boldsymbol{x}_t}\log p_t(\boldsymbol{y}|\boldsymbol{x}_t))]dt + \sqrt{\beta(t)}d\boldsymbol{w} \quad (10)$$

Solving Inverse Problem with Diffusion Model

Difficulty of computing $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{y}|\boldsymbol{x}_t)$
Notice that

$$p(\boldsymbol{y}|\boldsymbol{x}_t) = \int p(\boldsymbol{y}|\boldsymbol{x}_0)p(\boldsymbol{x}_0|\boldsymbol{x}_t)d\boldsymbol{x}_0 \tag{11}$$

1. Given $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}_0) + \boldsymbol{n}$, where $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$, we have

$$p(\boldsymbol{y}|\boldsymbol{x}_0) \sim \mathcal{N}(\boldsymbol{y}|\mathcal{A}(\boldsymbol{x}_0), \sigma^2 \boldsymbol{I}) \tag{12}$$

2. Unfortunately, $p(\boldsymbol{x}_0|\boldsymbol{x}_t)$ cannot be computed

$$p(\boldsymbol{x}_0|\boldsymbol{x}_t) = \frac{p(\boldsymbol{x}_t|\boldsymbol{x}_0)p(\boldsymbol{x}_0)}{p(\boldsymbol{x}_t)} \tag{13}$$

Solving Inverse Problem with Diffusion Model

Possible solution: Approximate $\nabla_{\boldsymbol{x}_t} log p(\boldsymbol{y}|\boldsymbol{x}_t)$ with something tractable

In the case of Gaussian Noise, we have:

$$p(\boldsymbol{y}|\boldsymbol{x}_0) = \frac{1}{\sqrt{(2\pi)^m \sigma^{2m}}} \exp\left[-\frac{||\boldsymbol{y} - \mathcal{A}(\boldsymbol{x}_0)||_2^2}{2\sigma^2}\right] \tag{14}$$

, therefore we may directly do the following

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y}|\boldsymbol{x}_t) \approx -\frac{1}{\sigma^2} \nabla_{\boldsymbol{x}_t} ||\boldsymbol{y} - \mathcal{A}(\boldsymbol{x}_t)||_2^2 \tag{15}$$

This naive approximation achieves good performance on MRI reconstruction and is published in a top conference. However, the approximation is fundamentally wrong.

Solving Inverse Problem with Diffusion Model

A better approximation: DPS

$$\nabla_{\boldsymbol{x}_t} logp(\boldsymbol{y}|\boldsymbol{x}_t) \approx \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}_0) \tag{16}$$

, where

$$\begin{aligned} \hat{\boldsymbol{x}}_0 = E[\boldsymbol{x}_0|\boldsymbol{x}_t] &= \frac{1}{\sqrt{\bar{\alpha}(t)}}(\boldsymbol{x}_t + (1-\bar{\alpha}(t))\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)) \\ &\approx \frac{1}{\sqrt{\bar{\alpha}(t)}}(\boldsymbol{x}_t + (1-\bar{\alpha}(t))\nabla_{\boldsymbol{x}_t} \boldsymbol{s}^*_\theta(\boldsymbol{x}_t, t)) \end{aligned} \tag{17}$$

Solving Inverse Problem with Diffusion Model

Solution:

$$\nabla_{\boldsymbol{x}_t} log p(\boldsymbol{y}|\boldsymbol{x}_t) \approx \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}_0) \tag{18}$$

In the case of Gaussian Noise, we have:

$$p(\boldsymbol{y}|\boldsymbol{x}_0) = \frac{1}{\sqrt{(2\pi)^m \sigma^{2m}}} \exp\left[-\frac{||\boldsymbol{y} - \mathcal{A}(\boldsymbol{x}_0)||_2^2}{2\sigma^2}\right] \tag{19}$$

, therefore

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y}|\boldsymbol{x}_t) \approx -\frac{1}{\sigma^2} \nabla_{\boldsymbol{x}_t} ||\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))||_2^2 \tag{20}$$

The Reverse process becomes:

$$d\boldsymbol{x} = [-\frac{\beta(t)}{2}\boldsymbol{x} - \beta(t)(s_{\theta^*}(\boldsymbol{x}_t, t) - \rho \nabla_{\boldsymbol{x}_t} ||\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))||_2^2)]dt + \sqrt{\beta(t)} d\boldsymbol{w} \tag{21}$$

Diffusion Posterior Sampling (DPS)

The resulting algorithm in discrete setting:

Algorithm 1 DPS - Gaussian

Require: $N, \boldsymbol{y}, \{\zeta_i\}_{i=1}^N, \{\tilde{\sigma}_i\}_{i=1}^N$

$\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

for $i = N-1$ **to** 0 **do**

  $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{s}_\theta(\boldsymbol{x}_i, i)$

  $\hat{\boldsymbol{x}}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}(\boldsymbol{x}_i + (1-\bar{\alpha}_i)\hat{\boldsymbol{s}})$

  $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

  $\boldsymbol{x}'_{i-1} \leftarrow \frac{\sqrt{\alpha_i}(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i}\boldsymbol{x}_i + \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_i}{1-\bar{\alpha}_i}\hat{\boldsymbol{x}}_0 + \tilde{\sigma}_i \boldsymbol{z}$

  $\boldsymbol{x}_{i-1} \leftarrow \boldsymbol{x}'_{i-1} - \zeta_i \nabla_{\boldsymbol{x}_i} \|\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0)\|_2^2$

end for

return $\hat{\mathbf{x}}_0$

Algorithm 2 DPS - Poisson

Require: $N, \boldsymbol{y}, \{\zeta_i\}_{i=1}^N, \{\tilde{\sigma}_i\}_{i=1}^N$

$\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

for $i = N-1$ **to** 0 **do**

  $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{s}_\theta(\boldsymbol{x}_i, i)$

  $\hat{\boldsymbol{x}}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}(\boldsymbol{x}_i + (1-\bar{\alpha}_i)\hat{\boldsymbol{s}})$

  $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

  $\boldsymbol{x}'_{i-1} \leftarrow \frac{\sqrt{\alpha_i}(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i}\boldsymbol{x}_i + \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_i}{1-\bar{\alpha}_i}\hat{\boldsymbol{x}}_0 + \tilde{\sigma}_i \boldsymbol{z}$

  $\boldsymbol{x}_{i-1} \leftarrow \boldsymbol{x}'_{i-1} - \zeta_i \nabla_{\boldsymbol{x}_i} \|\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0)\|_{\boldsymbol{\Lambda}}^2$

end for

return $\hat{\mathbf{x}}_0$

Figure: DPS algorithms

Diffusion Posterior Sampling (DPS)

The intuition is simple, we approximate $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y}|\boldsymbol{x}_t)$ with $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t))$, that is

1. Predict $\boldsymbol{x}_0$ from $\boldsymbol{x}_t$ with the posterior mean $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$
2. Take a gradient descent step on $\boldsymbol{x}_t$ such that the predicted $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$ agrees with the measurements $\boldsymbol{y}$

Diffusion Posterior Sampling (DPS)

Theoretical perspective:

$$p(\boldsymbol{y}|\boldsymbol{x}_t) = \int p(\boldsymbol{y}|\boldsymbol{x}_0)p(\boldsymbol{x}_0|\boldsymbol{x}_t)d\boldsymbol{x}_0$$

Let's define

$$f(\boldsymbol{x}) = p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^m \sigma^{2m}}} \exp\left[-\frac{||\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})||_2^2}{2\sigma^2}\right] \tag{22}$$

, then

1. $p(\boldsymbol{y}|\boldsymbol{x}_t) = E_{\boldsymbol{x}\sim p(\boldsymbol{x}_0|\boldsymbol{x}_t)}[f(\boldsymbol{x})]$
2. $p(\boldsymbol{y}|\hat{\boldsymbol{x}}_0) = f(E_{\boldsymbol{x}\sim p(\boldsymbol{x}_0|\boldsymbol{x}_t)}(\boldsymbol{x}))$

Diffusion Posterior Sampling (DPS)

DPS approximates $E[f(\boldsymbol{x})]$ with $f(E[\boldsymbol{x}])$

- Definition: Let $\boldsymbol{x}$ be a random variable with distribution $p(\boldsymbol{x})$. For some function f, the Jenson gap is define as

$$\mathcal{J}(f, \boldsymbol{x} \sim p(\boldsymbol{x}) = E[f(\boldsymbol{x})] - f(E[\boldsymbol{x}]) \tag{23}$$

- Theorem: For the given measurement model with measurement noise $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$, the Jenson gap is upper bounded by

$$\mathcal{J} \leq \frac{d}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}} ||\nabla_{\boldsymbol{x}} \mathcal{A}(\boldsymbol{x})|| m_1 \tag{24}$$

, where $||\nabla_{\boldsymbol{x}} \mathcal{A}(\boldsymbol{x})|| := \max_{\boldsymbol{x}} ||\nabla_{\boldsymbol{x}} \mathcal{A}(\boldsymbol{x})||$ and $m_1 := \int ||\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0|| p(\boldsymbol{x}_0|\boldsymbol{x}_t) d\boldsymbol{x}_0$

Experiments

Figure: Results on solving linear inverse problems with Gaussian noise

Experiments

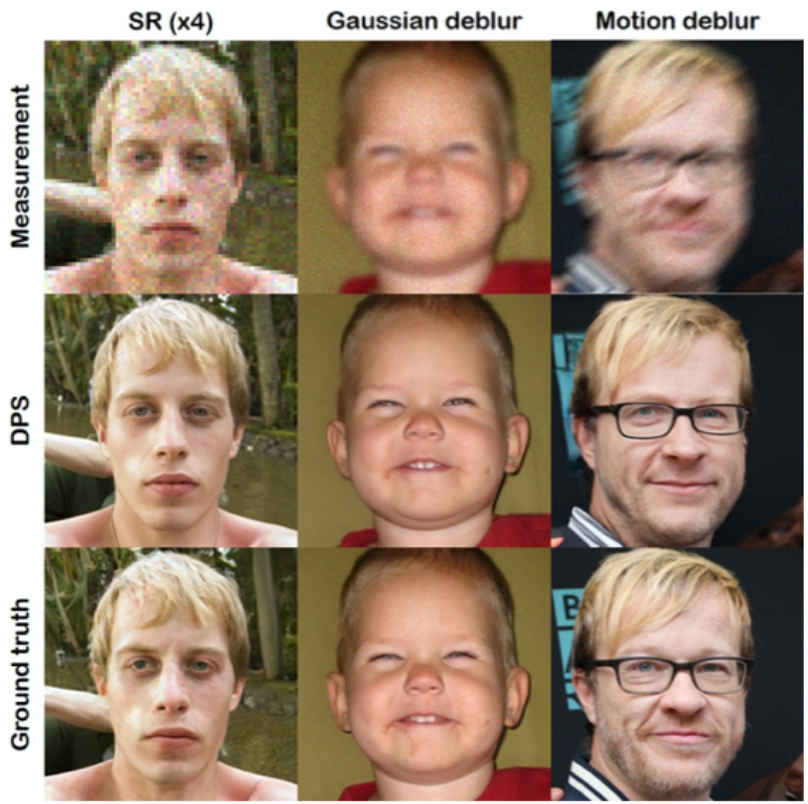

Figure: Results on solving linear inverse problems with Poisson noise

Experiments

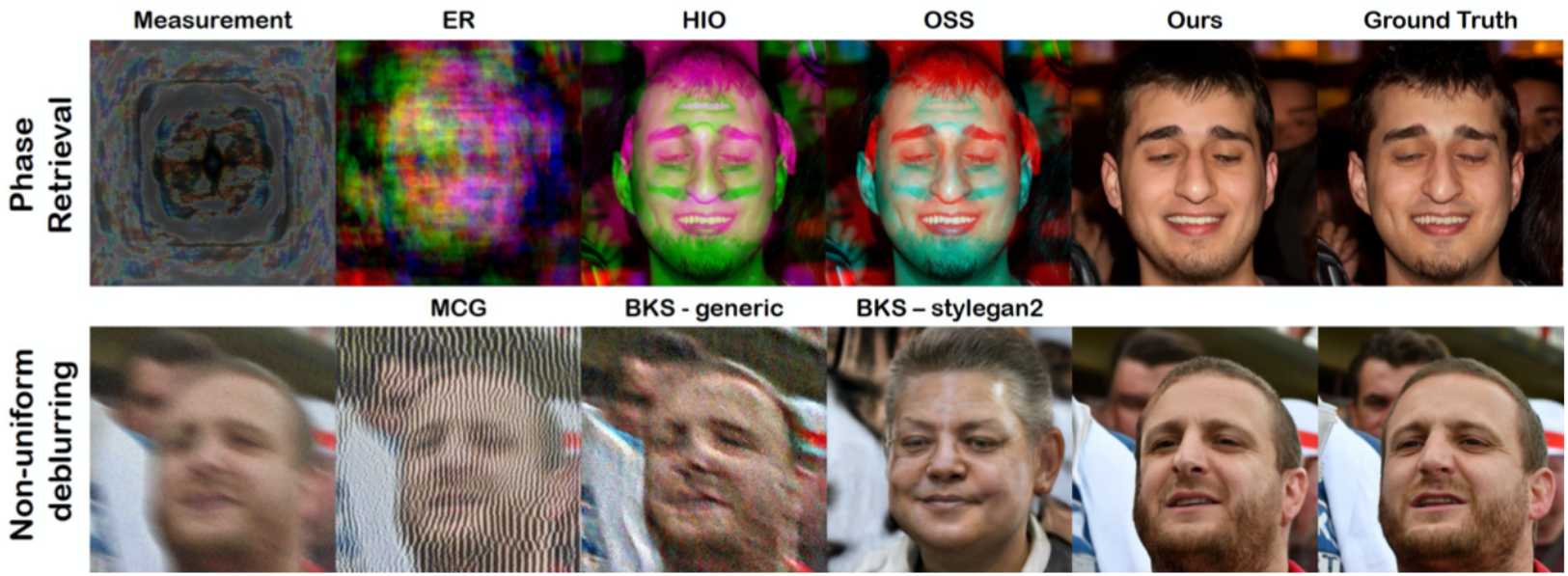

Figure: Results on solving nonlinear inverse problems with Gaussian noise

Theory

DPS lacks theoretical justification
Jenson Gap is not informative:

$$\mathcal{J} \leq \frac{d}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}} ||\nabla_{\boldsymbol{x}} \mathcal{A}(\boldsymbol{x})|| m_1$$

, where

$$||\nabla_{\boldsymbol{x}} \mathcal{A}(\boldsymbol{x})|| := \max_{\boldsymbol{x}} ||\nabla_{\boldsymbol{x}} \mathcal{A}(\boldsymbol{x})||$$

and

$$m_1 := \int ||\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0|| p(\boldsymbol{x}_0|\boldsymbol{x}_t) d\boldsymbol{x}_0$$

Theory

Question: Can DPS solve inverse problems in a perfectly recoverable setting ?

- Assume the images $\boldsymbol{x}_0 \in R^d$ reside in a linear subspace $\boldsymbol{x}_0 = \mathcal{S}\boldsymbol{w}_0, \mathcal{S} \in R^{dxl}, \boldsymbol{w}_0 \in \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_l)$
- Consider linear inverse problem only: $\boldsymbol{y} = \mathcal{A}\boldsymbol{x}_0 + \boldsymbol{n}$
- Assume $\mathcal{S}$ has orthonormal columns, i.e., $\mathcal{S}^T\mathcal{S} = \boldsymbol{I}_d$
- Assume the measurement operator satisfies $(\mathcal{AS})^T\mathcal{AS} \succ 0$

Theory

In the Discrete Setting:

- Forward Process:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I})$$

- Reverse process:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) := \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \Sigma_\theta(\boldsymbol{x}_t, t))$$

- Approximate $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ with $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}; \tilde{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I}) \tag{25}$$

- Training objective:

$$\min_\theta E_{t,\boldsymbol{x}_0} ||\tilde{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)||_2 \tag{26}$$

Theory

Theorem 1: Unconditional Sampling

- Consider a one step diffusion model

$$\boldsymbol{x}_1(\boldsymbol{x}_0, \epsilon) = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \beta_t \boldsymbol{\epsilon} \tag{27}$$

, then the training objective becomes

$$\theta^* = \text{argmin}_\theta E_{\boldsymbol{x}_0, \boldsymbol{\epsilon}} ||\tilde{u}_1(\boldsymbol{x}_1(\boldsymbol{x}_0, \boldsymbol{\epsilon}), \boldsymbol{x}_0) - \mu_\theta(\boldsymbol{x}_1(\boldsymbol{x}_0, \boldsymbol{\epsilon}))||_2^2 \tag{28}$$

- We generate a new image by first randomly sampling from a Gaussian $\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, then sample from $p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)$, i,e, $\boldsymbol{x}_0 = \mu_\theta(\boldsymbol{x}_1)$
- Assume $\mu_\theta(\cdot)$ is a linear network, i.e., $\mu_\theta(\boldsymbol{x}_1(\boldsymbol{x}_0, \boldsymbol{\epsilon})) = \theta \boldsymbol{x}_1(\boldsymbol{x}_0, \boldsymbol{\epsilon})$, then $\theta^* = c\mathcal{S}\mathcal{S}^T$

Conditional Sampling with DPS

Theorem 2:
Under the simplified setting, with a properly chosen step size, we can perfectly recover $\boldsymbol{x}_0$ from $\boldsymbol{y}_0$

Algorithm 1 DPS - Gaussian

Require: $N, \boldsymbol{y}, \{\zeta_i\}_{i=1}^N, \{\tilde{\sigma}_i\}_{i=1}^N$

$\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
for $i = N-1$ **to** 0 **do**
  $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{s}_\theta(\boldsymbol{x}_i, i)$
  $\hat{\boldsymbol{x}}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}(\boldsymbol{x}_i + (1-\bar{\alpha}_i)\hat{\boldsymbol{s}})$
  $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\boldsymbol{x}'_{i-1} \leftarrow \frac{\sqrt{\alpha_i}(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i}\boldsymbol{x}_i + \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_i}{1-\bar{\alpha}_i}\hat{\boldsymbol{x}}_0 + \tilde{\sigma}_i \boldsymbol{z}$
  $\boldsymbol{x}_{i-1} \leftarrow \boldsymbol{x}'_{i-1} - \zeta_i \nabla_{\boldsymbol{x}_i} \|\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0)\|_2^2$
end for
return $\hat{\mathbf{x}}_0$

Algorithm 2 DPS - Poisson

Require: $N, \boldsymbol{y}, \{\zeta_i\}_{i=1}^N, \{\tilde{\sigma}_i\}_{i=1}^N$

$\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
for $i = N-1$ **to** 0 **do**
  $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{s}_\theta(\boldsymbol{x}_i, i)$
  $\hat{\boldsymbol{x}}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}(\boldsymbol{x}_i + (1-\bar{\alpha}_i)\hat{\boldsymbol{s}})$
  $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\boldsymbol{x}'_{i-1} \leftarrow \frac{\sqrt{\alpha_i}(1-\bar{\alpha}_{i-1})}{1-\bar{\alpha}_i}\boldsymbol{x}_i + \frac{\sqrt{\bar{\alpha}_{i-1}}\beta_i}{1-\bar{\alpha}_i}\hat{\boldsymbol{x}}_0 + \tilde{\sigma}_i \boldsymbol{z}$
  $\boldsymbol{x}_{i-1} \leftarrow \boldsymbol{x}'_{i-1} - \zeta_i \nabla_{\boldsymbol{x}_i} \|\boldsymbol{y} - \mathcal{A}(\hat{\boldsymbol{x}}_0)\|_{\boldsymbol{\Lambda}}^2$
end for
return $\hat{\mathbf{x}}_0$

Figure: DPS algorithms

Conclusion

1. DPS works well on various of inverse problems
2. DPS utilized pretrained diffusion model in an unsupervised manner
3. Under extremely simplified settings, DPS is able to solve inverse problems perfectly.

Quizes

1. What structure will a matrix have if it is the linear operator for inpainting?
2. Why is the likelihood $p(\boldsymbol{y}|\boldsymbol{x}_t)$ intractable?