Solving Inverse Problems with Diffusion Model-based Priors

EE367/CS448I: Computational Imaging

stanford.edu/class/ee367

Lecture 13

Gordon Wetzstein

Stanford University

Recap

- Model data distribution using score function (gradient of log prior)
- Score function can be learned robustly via denoising score matching
- Diffusion models can be represented as SDEs/ODEs → rich literature on solvers, sampling, etc.
- Generate new examples using sampling

Overview

- Diffusion Model Sampling (continued)

- Image Editing/Translation with SDEdit

- Solving Inverse Problems with Posterior Sampling

Diffusion Model Sampling

(continued)

(Continuous) Diffusion Models as SDEs

noise with variance $\mathrm{d}t$

- Forward SDE: $d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)d\mathbf{w}$

drift coefficient

diffusion coefficient

- Reverse SDE: $d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_x \log p_t(\mathbf{x})]\mathrm{d}t + g(t)d\widetilde{\mathbf{w}}$

standard Wiener process, i.e., noise with variance $\mathrm{d}t$

Song et al., ICLR 2021

B. Anderson, "Reverse-time diffusion equation models", Stochastic Processes and their Applications, 1982

(Continuous) Diffusion Models as SDEs

- Forward SDE: $d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)d\mathbf{w}$

- Discretize SDE using the Euler-Maruyama (or Euler) method:

$$x_t = x_{t-1} + \mathbf{f}(x_{t-1}, t)\Delta t + g(t)\sqrt{\Delta t}\epsilon$$

noise, $\epsilon \sim \mathcal{N}(0,1)$

- Same idea for reverse SDE

Denoising Diffusion Probabilistic Models (DDPM)

- DDPM is a specific form of the general, discretized SDE
- Variance-preserving forward diffusion: $x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1}$

noise schedule, i.e., how much noise to add at time step t

Ho et al., NeurIPS 2020

Denoising Diffusion Probabilistic Models (DDPM)

- DDPM is a specific form of the general, discretized SDE
- Variance-preserving forward diffusion: $x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1}$
- One-step forward diffusion:

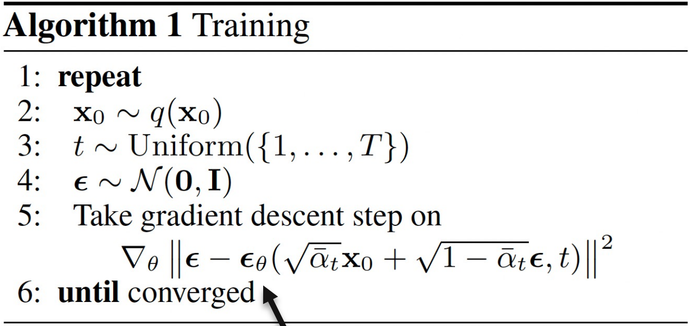
$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \sim \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t)I\right)$$

$$\alpha_t = 1-\beta_t, \qquad \bar{\alpha}_t = \Pi_{s=1}^{t}\alpha_s, \quad \epsilon \sim \mathcal{N}(0, I)$$

- Reverse diffusion: $x_{t-1} = \dfrac{1}{\sqrt{\alpha_t}}\left(x_t - \dfrac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\, x_t, t)\right) + \sqrt{\beta_t}\, \epsilon_t$

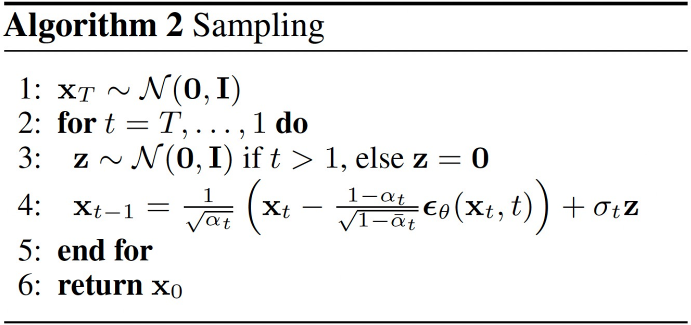
Ho et al., NeurIPS 2020

Denoising Diffusion Probabilistic Models (DDPM)



Algorithm 1 Training

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



Algorithm 2 Sampling

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

train a noise estimation network!

Ho et al., NeurIPS 2020

Denoising Diffusion Probabilistic Models (DDPM)

- DDPM is a specific form of the general SDE :

$$\mathbf{f}(\mathbf{x}, t) = -\frac{1}{2}\beta(t)\mathbf{x}, \quad g(t) = \sqrt{\beta(t)}$$

- Not straightforward to see relationship between SDE and update rules via Euler; see Song et al., Appendix B (Eqs. 22-25) for detailed derivation

- The DDPM sampling scheme requires many steps (e.g., T=1,000 or more)

Ho et al., NeurIPS 2020; Song et al., ICLR 2021

Other SDE Formulations / Sampling Schemes

- DDIM (Song et al.) significantly reduces the number of steps compared with DDPM using a non-Markovian derivation

- Karras et al. derived a unified SDE-based formulation that leads to a family of viable drift & diffusion coefficients, with both DDPM & DDIM being special cases

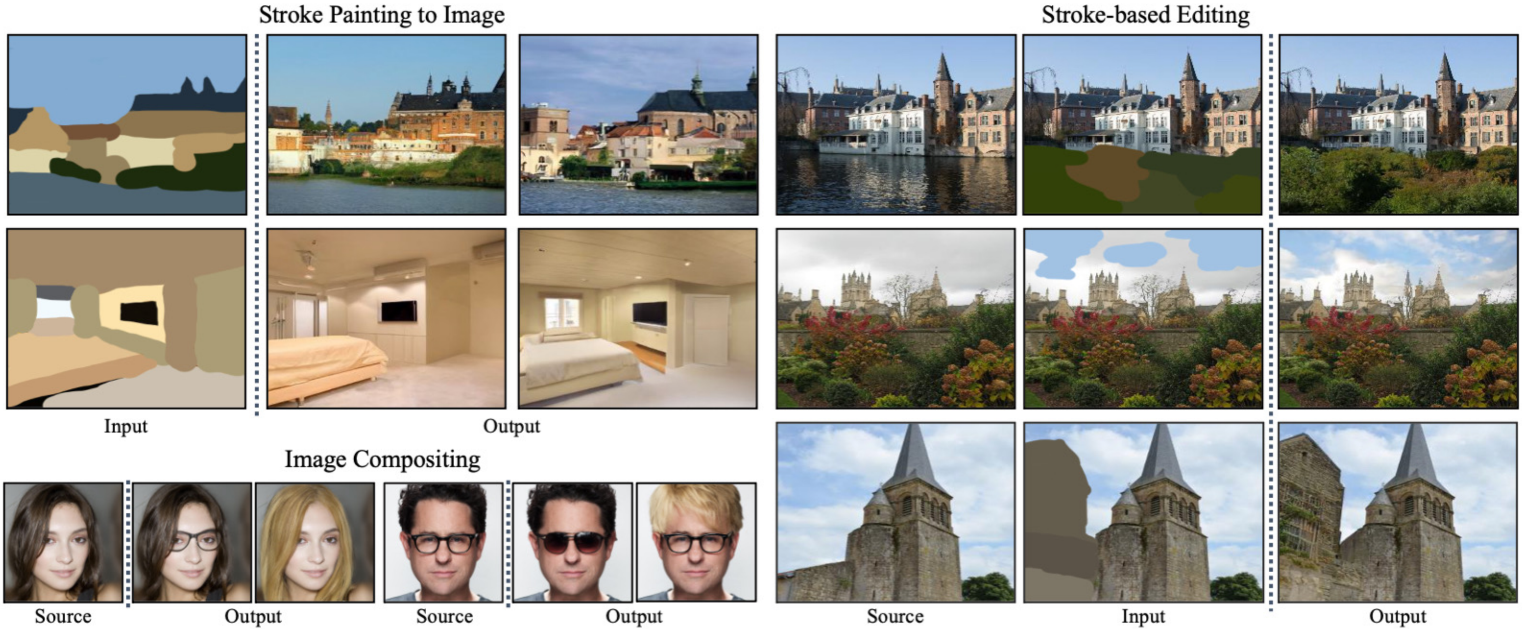
- Rich literature on sampling schemes; e.g., StableDiffusion (<SD3) used DPM-Solver++ (Lu et al.)

- Newer versions of diffusion models, including FLUX and SD3, use a slightly different mechanism - flow matching (Lipman et al.; Hawley)

Song et al., ICLR 2021; Karras et al., NeurIPS 2022; Lu et al., NeurIPS 2022; Lipman et al., ICLR 2023; **Hawley, blog post 2024**

Image Editing/Translation via SDEdit

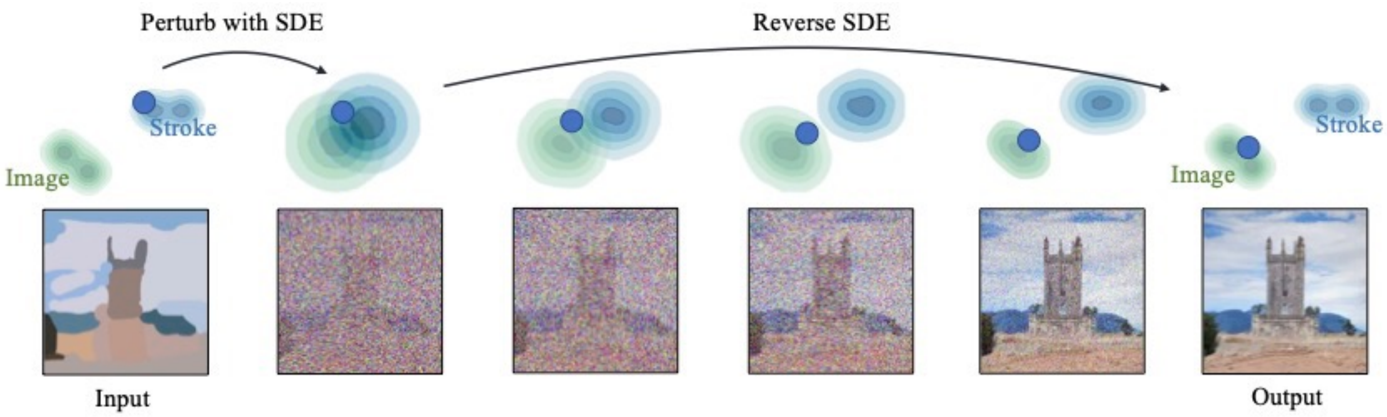
SDEdit



Meng et al., ICLR 2022

SDEdit

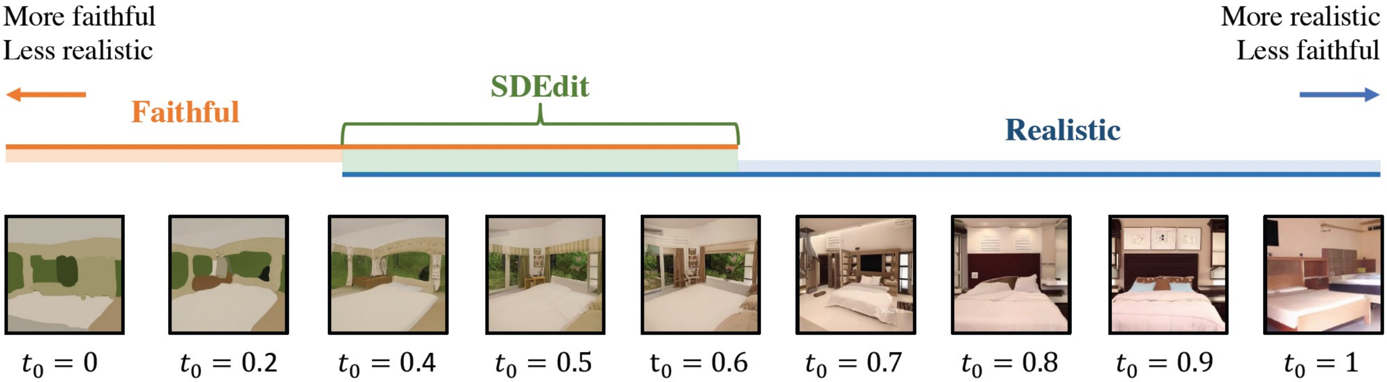
1. Add noise to input image via forward diffusion process
2. Apply reverse diffusion process



Meng et al., ICLR 2022

SDEdit

Question: how much noise to add?



Input/guide image

Meng et al., ICLR 2022

SDEdit

- SDEdit doesn't solve inverse problems, but it's a simple-enough image editing, or image-to-image translation, approach that it might be useful
- To solve an inverse problem, we need to take the image formation model into account!

A Bayesian Perspective of Inverse Problems

A Bayesian Perspective of Inverse Problems

…from lecture 10…

- Image formation model: $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\eta}, \quad \boldsymbol{b} \in \mathbb{R}^{M}, \boldsymbol{x} \in \mathbb{R}^{N}, \boldsymbol{A} \in \mathbb{R}^{M\times N}$

- Interpret as random variables: $$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{x}_i, 0),\ \boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2)$$ $$\boldsymbol{b}_i \sim \mathcal{N}\left((\boldsymbol{A}\boldsymbol{x})_i, \sigma^2\right)$$

- Probability of observation i: $$p(\boldsymbol{b}_i|\boldsymbol{x}_i, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\boldsymbol{b}_i - (\boldsymbol{A}\boldsymbol{x})_i)^2}{2\sigma^2}}$$

- Joint probability of all observations: $$p(\boldsymbol{b}|\boldsymbol{x}, \sigma) = \prod_{i=1}^{M} p(\boldsymbol{b}_i|\boldsymbol{x}_i, \sigma) \propto e^{-\frac{\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2^2}{2\sigma^2}}$$

A Bayesian Perspective of Inverse Problems

…from lecture 10…

- Bayes' rule:

$$\underset{\substack{\uparrow \\ \text{posterior}}}{p(\boldsymbol{x}|\boldsymbol{b},\sigma)} = \frac{p(\boldsymbol{b}|\boldsymbol{x},\sigma)\boldsymbol{p}(\boldsymbol{x})}{p(\boldsymbol{b})} \propto \underset{\substack{\uparrow \\ \text{image formation model} \\ \text{or, better, likelihood}}}{p(\boldsymbol{b}|\boldsymbol{x},\sigma)}\underset{\substack{\uparrow \\ \text{prior}}}{p(\boldsymbol{x})}$$

Maximum Likelihood

…from lecture 10…

- Bayes' rule: $p(\boldsymbol{x}|\boldsymbol{b},\sigma) = \frac{p(\boldsymbol{b}|\boldsymbol{x},\sigma)\boldsymbol{p}(\boldsymbol{x})}{p(\boldsymbol{b})} \propto p(\boldsymbol{b}|\boldsymbol{x},\sigma)p(\boldsymbol{x})$

posterior

image formation model
or, better, likelihood

prior

- Maximum likelihood (ML) solution:

$$\boldsymbol{x}_{ML} = \arg\min_x \; -\log\big(p(\boldsymbol{x}|\boldsymbol{b},\sigma)\big)$$

$$= \arg\min_x \; -\log\big(p(\boldsymbol{b}|\boldsymbol{x},\sigma)\big) - \log\big(p(\boldsymbol{x})\big)$$

$$\approx \arg\min_x \; \frac{1}{2\sigma^2}\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2^2$$

$\log(p(\boldsymbol{x})=1)=0$

Maximum-a-posteriori

…from lecture 10…

- Bayes’ rule:

$$\underset{\substack{\uparrow\\ \text{posterior}}}{p(\boldsymbol{x}|\boldsymbol{b},\sigma)} = \frac{p(\boldsymbol{b}|\boldsymbol{x},\sigma)\boldsymbol{p}(\boldsymbol{x})}{p(\boldsymbol{b})} \propto \underset{\substack{\uparrow\\ \text{image formation model}\\ \text{or, better, likelihood}}}{p(\boldsymbol{b}|\boldsymbol{x},\sigma)}\underset{\substack{\uparrow\\ \text{prior}}}{p(\boldsymbol{x})}$$

- Maximum-a-posterior (MAP) solution:

$$\begin{aligned}\boldsymbol{x}_{MAP} &= \arg\min_x \, -\log\big(p(\boldsymbol{x}|\boldsymbol{b},\sigma)\big)\\ &= \arg\min_x \, -\log\big(p(\boldsymbol{b}|\boldsymbol{x},\sigma)\big) - \log\big(p(\boldsymbol{x})\big)\\ &= \arg\min_x \, \frac{1}{2\sigma^2}\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2^2 + \Psi(\boldsymbol{x})\end{aligned}$$

Posterior Sampling

- Bayes' rule: $$\underset{\underset{\text{posterior}}{\uparrow}}{p(\boldsymbol{x}|\boldsymbol{b},\sigma)} = \frac{p(\boldsymbol{b}|\boldsymbol{x},\sigma)\boldsymbol{p}(\boldsymbol{x})}{p(\boldsymbol{b})} \propto \underset{\underset{\text{image formation model or, better, likelihood}}{\uparrow}}{p(\boldsymbol{b}|\boldsymbol{x},\sigma)}\underset{\underset{\text{prior}}{\uparrow}}{p(\boldsymbol{x})}$$

- Posterior Sampling: sample feasible solutions of the inverse problem from the from the conditional distribution $p(\boldsymbol{x}|\boldsymbol{b},\sigma)$

ML

- simple to compute, e.g., using closed-form solution or (stochastic) gradient descent
- considers physical image formation model
- doesn’t consider priors
- Only suitable for well-posed, but not ill-posed inverse problems

MAP

- Solve using plug and play, e.g., using ADMM (see lectures 10, 11)
- considers physical image formation model
- considers priors
- “point estimate”, i.e., “gives a single solution”
- best for “slightly to moderately” ill-posed inverse problems

Posterior Sampling

- Solve e.g. using guided diffusion model sampling (this lecture)
- considers physical image formation model
- considers priors
- gives a different feasible solution each time
- best for “moderately to highly” ill-posed inverse problems

Posterior Sampling with Diffusion Models

Naive Approach

- Train a new diffusion model on $p_{data}(\mathbf{x}|\mathbf{b})$ instead of $p_{data}(\mathbf{x})$?
- This is usually not feasible, because $p_{data}(\mathbf{x}|\mathbf{b})$ depends on measurement operator $\boldsymbol{A}$ and the diffusion model would have to be trained for each operator $\boldsymbol{A}$!
- Instead: work with diffusion model pretrained on $p_{data}(\mathbf{x})$

Song et al., ICLR 2021

Sampling with Guidance

- Reverse SDE: $d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_x \log p_t(\mathbf{x})]\mathrm{d}t + g(t)d\widetilde{\mathbf{w}}$

- Replace unconditional distribution with conditional using Bayes' rule: $p_t(\mathbf{x}|\mathbf{b}) \propto p_t(\mathbf{b}|\mathbf{x})p_t(\mathbf{x})$

- Intuition: "guide" the diffusion model sampling by the measurements $\mathbf{b}$ to sample from $p_{data}(\mathbf{x}|\mathbf{b})$ instead of $p_{data}(\mathbf{x})$

Song et al., ICLR 2021

Sampling with Guidance

- Reverse SDE: $d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_x \log p_t(\mathbf{x})]\mathrm{d}t + g(t)d\widetilde{\mathbf{w}}$

- Replace unconditional distribution with conditional using Bayes' rule: $p_t(\mathbf{x}|\mathbf{b}) \propto p_t(\mathbf{b}|\mathbf{x})p_t(\mathbf{x})$

- Conditional reverse SDE:

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)(\nabla_x \log p_t(\mathbf{x}) + \nabla_x \log p_t(\mathbf{b}|\mathbf{x}))]\mathrm{d}t + g(t)d\widetilde{\mathbf{w}}$$

Song et al., ICLR 2021

Sampling with Guidance

- Likelihood of measurements: $p(\boldsymbol{b}|\boldsymbol{x},\sigma) \propto e^{-\frac{\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2^2}{2\sigma^2}}$

- Gradient of log-likelihood:

distribution of “clean” images

$$\boxed{\nabla_x \log p(\mathbf{b}|\mathbf{x})} = -\nabla_x \frac{\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_2^2}{2\sigma^2} = -\frac{1}{\sigma^2}\left(\boldsymbol{A}^T(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x})\right)$$

- Problem: $\boxed{\nabla_x \log p(\mathbf{b}|\mathbf{x_0}) \neq \nabla_x \log p_t(\mathbf{b}|\mathbf{x}_t)}$

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x},t) - g^2(t)(\nabla_x \log p_t(\mathbf{x}) + \boxed{\nabla_x \log p_t(\mathbf{b}|\mathbf{x})})]\mathrm{d}t + g(t)d\tilde{\mathbf{w}}$$

distribution of “noisy” images

Song et al., ICLR 2021

Sampling with Guidance

- Problem: $\nabla_x \log p(\mathbf{b}|\mathbf{x_0}) \neq \nabla_x \log p_t(\mathbf{b}|\mathbf{x}_{\boldsymbol{t}})$

- Problem: likelihood of the measurements is given by intractable integral: $p_t(\mathbf{b}|\mathbf{x}_{\boldsymbol{t}}) = \int p(\mathbf{b}|\mathbf{x_0})p(\mathbf{x_0}|\mathbf{x}_{\boldsymbol{t}})\boldsymbol{d}\mathbf{x_0}$

- Solution: need to approximate $\nabla_x \log p_t(\mathbf{b}|\mathbf{x}_{\boldsymbol{t}})$ somehow!

Score ALD

- Problem: $\nabla_x \log p(\mathbf{b}|\mathbf{x_0}) \neq \nabla_x \log p_t(\mathbf{b}|\boldsymbol{x_t})$

- Approach: $\nabla_x \log p(\mathbf{b}|\mathbf{x_0}) \approx \nabla_x \log p_t(\mathbf{b}|\boldsymbol{x_t})$

$$\approx -\frac{1}{\sigma^2 + \gamma_t^2}\left(\boldsymbol{A}^T(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})\right)$$

$\uparrow$

guidance strength - hyperparameter

Jalal et al., NeurIPS 2021

Diffusion Posterior Sampling (DPS)

- Problem: $\nabla_x \log p(\mathbf{b}|\mathbf{x_0}) \neq \nabla_x \log p_t(\mathbf{b}|\mathbf{x_{\mathit{t}}})$

- Approach: $\nabla_x \log p(\mathbf{b}|\mathbf{x_0}) \approx \nabla_x \log p_t(\mathbf{b}|\mathbf{x_0} = \mathbb{E}[\mathbf{x_0}|\mathbf{x_{\mathit{t}}}])$

- Tweedie's formula for variance preserving (i.e., DDPM) discrete forward diffusion (see Eq. 10 in Chung et al.):

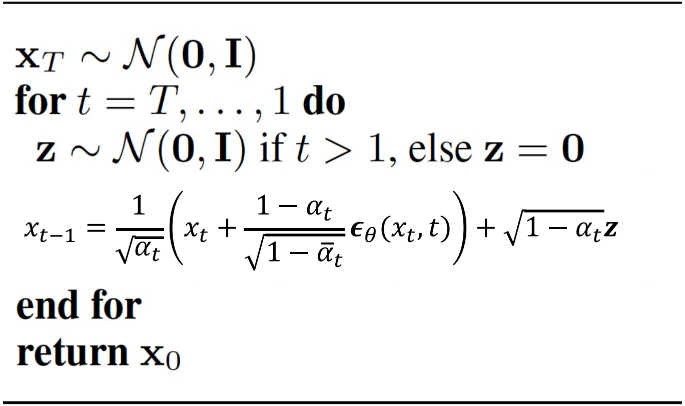
$$\hat{x}_0 = \mathbb{E}[\mathrm{x}_0|\mathrm{x}_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t + (1-\bar{\alpha}_t)\nabla_{x_t}\log p_t(x_t)\right)$$

Lu et al., NeurIPS 2022; Chung et al., ICLR 2023

DDPM vs. DPS

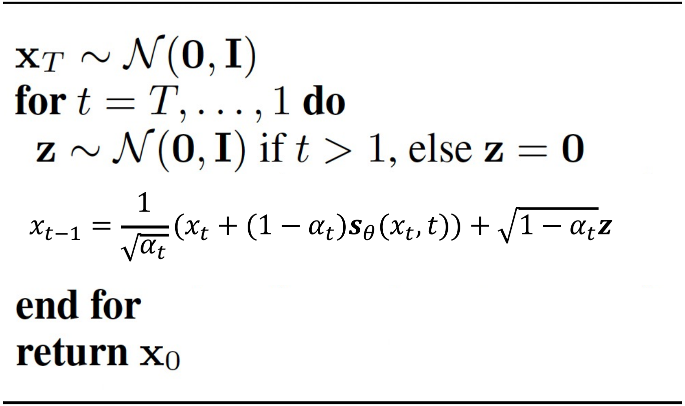
DDPM: noise prediction vs. score prediction

DDPM reverse diffusion w/ noise predictor



```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t + \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(x_t, t)\right) + \sqrt{1-\alpha_t}\boldsymbol{z}$
end for
return $\mathbf{x}_0$
```

DDPM reverse diffusion w/ score predictor



```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + (1-\alpha_t)\boldsymbol{s}_\theta(x_t, t)) + \sqrt{1-\alpha_t}\boldsymbol{z}$
end for
return $\mathbf{x}_0$
```

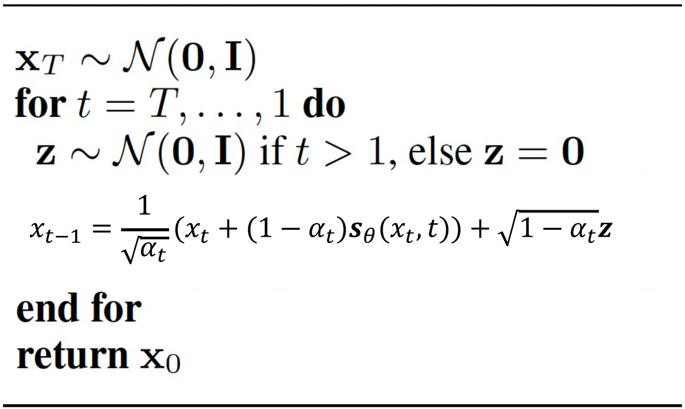
where $\bar{\alpha}_t \triangleq \prod_{i=1}^{t} \alpha_i$ and $\alpha_t \triangleq 1 - \beta_t$

- Noise prediction formulation is used in original paper; score prediction formulation in project
- Both formulations are equivalent

Ho et al., NeurIPS 2020

DDPM: score prediction variants

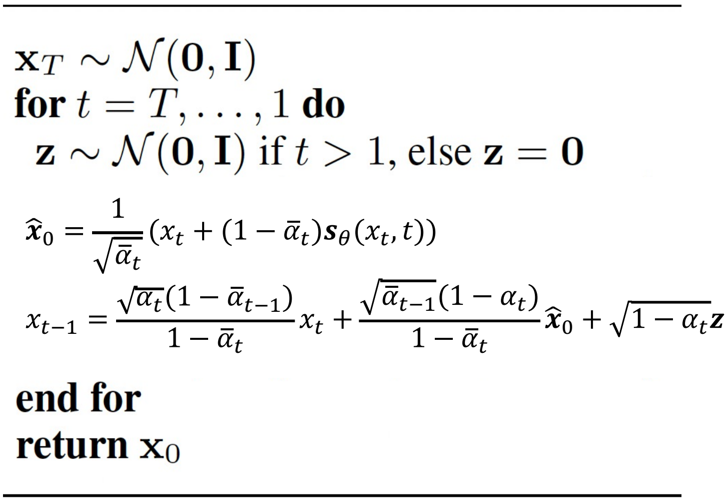
DDPM reverse diffusion w/ score predictor



```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t + (1-\alpha_t)\boldsymbol{s}_\theta(x_t, t)) + \sqrt{1-\alpha_t}\boldsymbol{z}$
end for
return $\mathbf{x}_0$
```

where $\bar{\alpha}_t \triangleq \prod_{i=1}^{t} \alpha_i$ and $\alpha_t \triangleq 1 - \beta_t$

DDPM reverse diffusion w/ score predictor



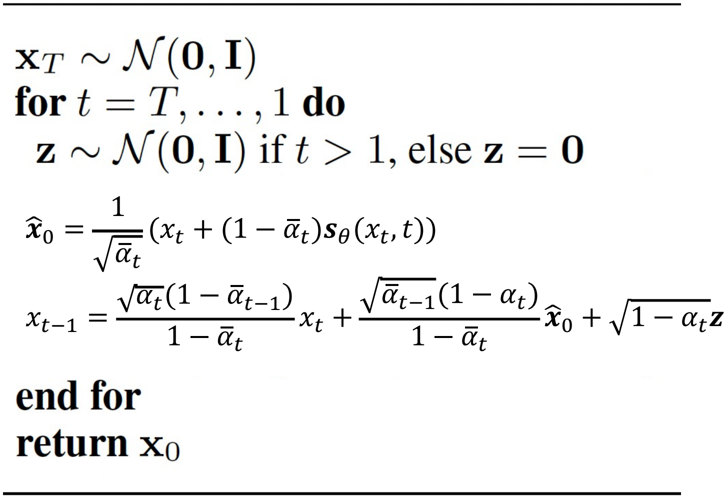
```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\widehat{\boldsymbol{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1-\bar{\alpha}_t)\boldsymbol{s}_\theta(x_t, t))$
  $x_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\widehat{\boldsymbol{x}}_0 + \sqrt{1-\alpha_t}\boldsymbol{z}$
end for
return $\mathbf{x}_0$
```

- Variant on right is slightly more complicated, but directly shows how the predicted clean image $\widehat{\boldsymbol{x}}_0$ is part of this

Ho et al., NeurIPS 2020

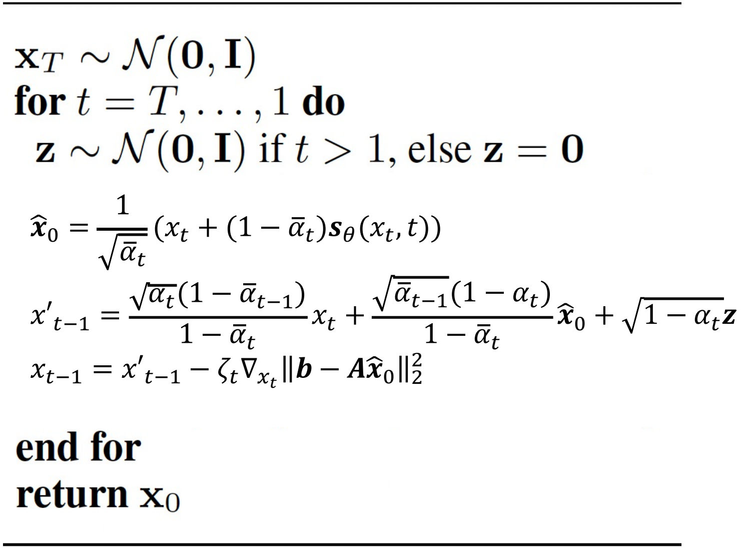
DDPM vs. DPS

DDPM reverse diffusion w/ score predictor



```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\hat{\boldsymbol{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1 - \bar{\alpha}_t)\boldsymbol{s}_\theta(x_t, t))$
  $x_{t-1} = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \hat{\boldsymbol{x}}_0 + \sqrt{1 - \alpha_t \boldsymbol{z}}$
end for
return $\mathbf{x}_0$
```

DPS reverse diffusion



```
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ do
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\hat{\boldsymbol{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t + (1 - \bar{\alpha}_t)\boldsymbol{s}_\theta(x_t, t))$
  $x'_{t-1} = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t} \hat{\boldsymbol{x}}_0 + \sqrt{1 - \alpha_t \boldsymbol{z}}$
  $x_{t-1} = x'_{t-1} - \zeta_t \nabla_{x_t} \|\boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{x}}_0\|_2^2$
end for
return $\mathbf{x}_0$
```

- DPS is equivalent to DDPM with one additional step!

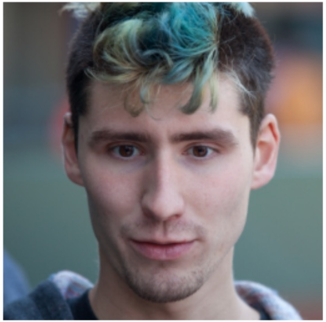
Chung et al., ICLR 2023

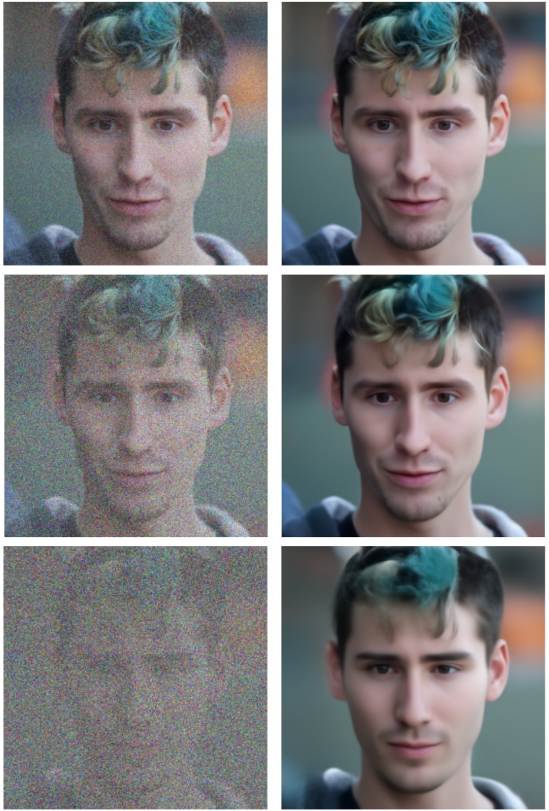
Results from Diffusion Project

Diffusion Project

Task 2: Single-step Denoising

x_0





	x_t	Denoised	PSNR/LPIPS
t=30			37.4 0.0378
t=100			32.7 0.0941
t=300			27.3 0.203

Diffusion Project

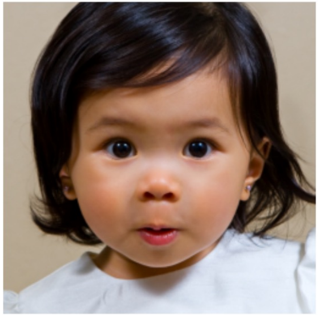
Task 3: Unconditional Image Generation

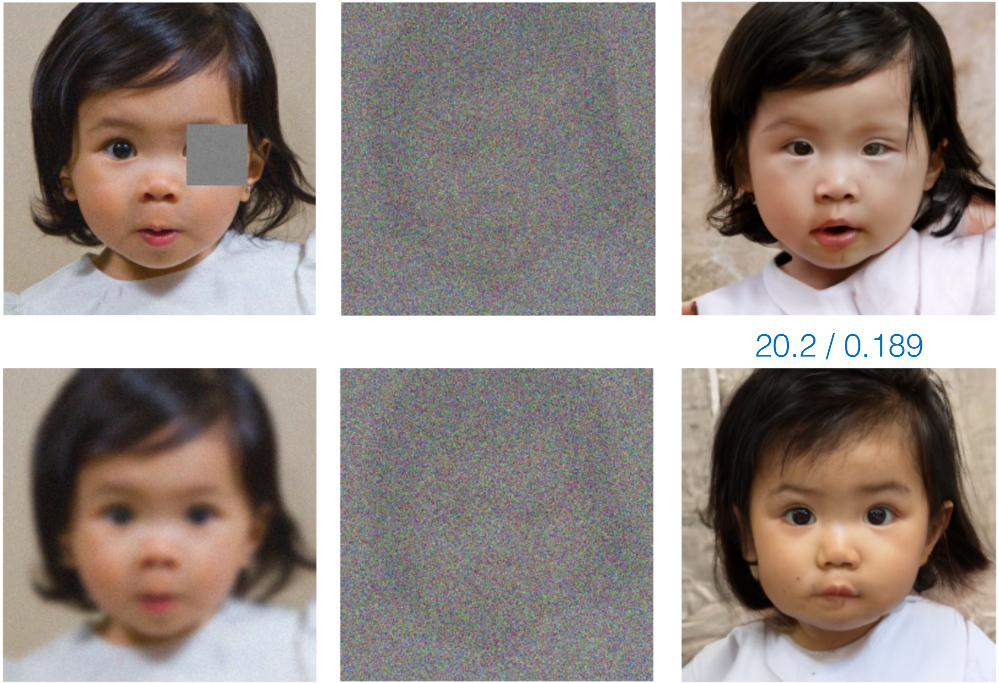
Example results

Diffusion Project

Task 4: SDEdit for Inverse Problems

Ground Truth





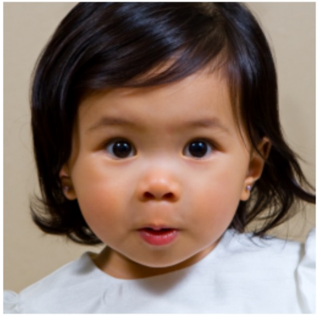
20.1 / 0.232

Diffusion Project

Task 5: ScoreALD for Inverse Problems

PSNR/LPIPS

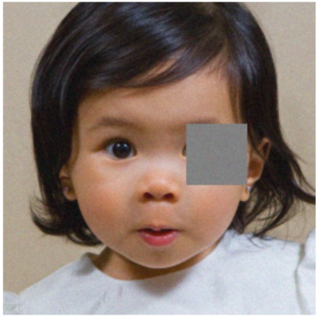
Ground Truth

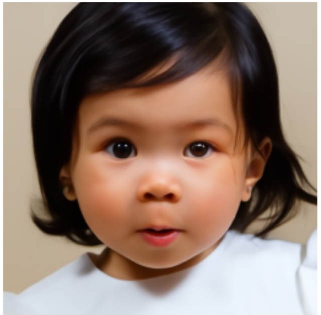


Input

Reconstruction

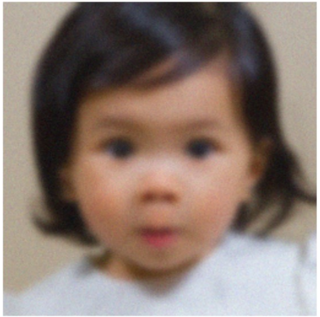
Inpainting

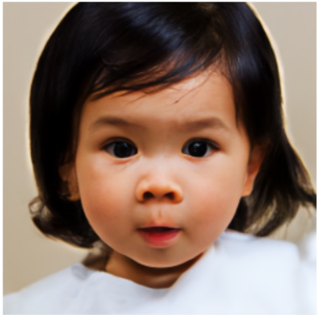




28.5 / 0.0605

Deconvolution





22.2 / 0.144

Diffusion Project

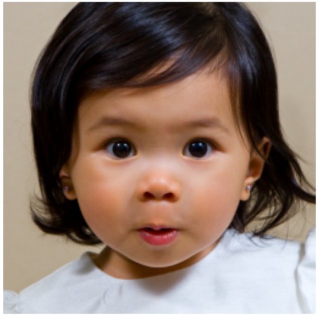
Task 6: DPS for Inverse Problems

PSNR/LPIPS

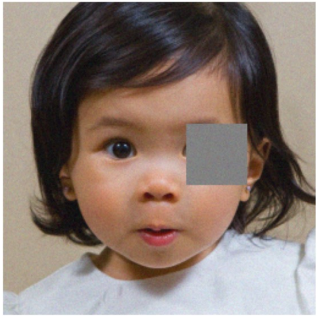
Input

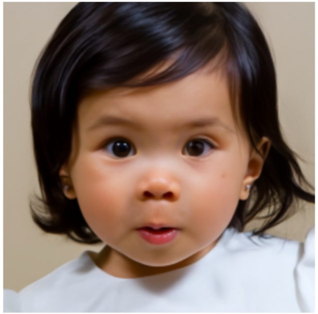
Reconstruction

Ground Truth



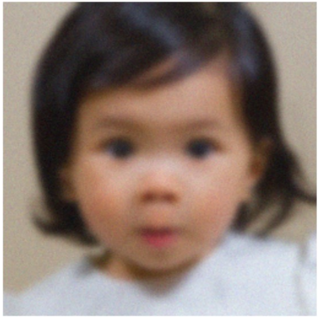
Inpainting

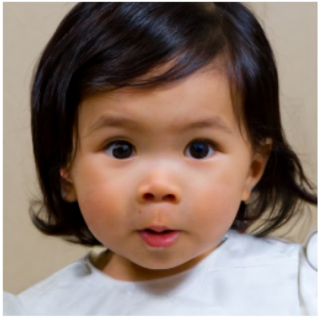




34.0 / 0.0239

Deconvolution





28.4 / 0.0518

Summary

- DPS works for linear and nonlinear inverse problems with Gaussian or Poisson noise! Also easy to understand and implement. That's why we discussed it

- However, Score-ALD and DPS are two of *many* approaches to approximate $\nabla_x \log p_t(\mathbf{b}|\mathbf{x}_t)$

- See Daras et al. "A Survey on Diffusion Models for Inverse Problems", arxiv 2024, for a great and comprehensive survey

References and Further Reading

Diffusion Models:

- Y. Song, S. Ermon, "Generative modeling by estimating gradients of the data distribution", NeurIPS 2019
- **J. Ho, A. Jain, P. Abbeel, "Denoising Diffusion Probabilistic Models", NeurIPS 2020 (DDPM)**
- J. Song, C. Meng, S. Ermon, "Denoising diffusion implicit models", ICLR, 2021 (DDIM)
- Y. Song, J. Sohl-Dickstein, D. Kingma, A. Kumar, S. Ermon, B. Poole, "Score-based generative modeling through stochastic differential equations", ICLR 2021
- T. Karras, M. Aittala, T. Aila, S. Laine, "Elucidating the Design Space of Diffusion-Based Generative Models", NeurIPS 2022
- **C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.Y. Zhu, S. Ermon, "SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations", ICLR 2022**
- C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, J. Zhu, "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps", NeurIPS 2022
- C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, J. Zhu, "DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models", arxiv 2022
- Y. Lipman, R. Chen, H. Ben-Hamu, M. Nickel, M. Le, "Flow Matching for Generative Modeling", ICLR 2023
- **G. Daras, H. Chung, C. Lai, Y. Mitsufuji, J. Ye, P. Milanfar, A. Dimakis, M. Delbracio, "A Survey on Diffusion Models for Inverse Problems", arxiv 2024**
- **A. Jalal, M. Arvinte, G. Daras, E. Price, A. Dimakis, J. Tamir, "Robust Compressed Sensing MRI with Deep Generative Priors", NeurIPS 2021**
- H. Chung, B. Sim, D. Ryu, J. Ye, Improving Diffusion Models for Inverse Problems using Manifold Constraints, NeurIPS 2022
- **H. Chung, J. Kim, M. Mccann, M. Klasky, J. Ye, "Diffusion Posterior Sampling for General Noisy Inverse Problems", ICLR 2023**
- **S. Hawley, “Flow with what you know”, https://drscotthawley.github.io/blog/posts/FlowModels.html, blog post 2024**

Plug & play image restoration – original paper and extension for diffusion models

- S. Venkatakrishnan, C. Bouman, B. Wohlberg. Plug-and-play priors for model based reconstruction. In 2013 IEEE Global Conference on Signal and Information Processing, 2013
- Y. Zhu, K. Zhang, J. Liang, J. Cao, B. Wen, R. Timofte, L. Gool. Denoising diffusion models for plug-and-play image restoration, CVPR 2023