LATINO-PRO: LAtent consisTency INverse sOlver with PRompt Optimization

Alessio Spagnoletti
Jean Prost, Andrés Almansa, Nicolas Papadakis, Marcelo Pereyra

Université Paris-Cité

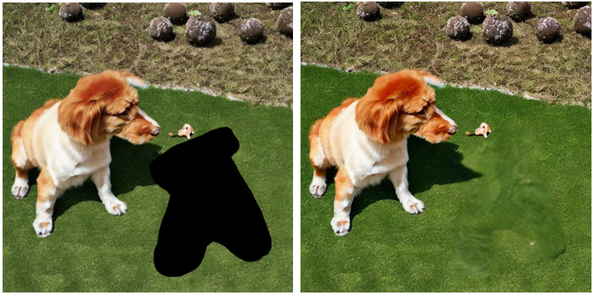
October 4, 2025

Inverse problems

Solving **inverse problems** is the task of **restoring data**, represented as vectors $\boldsymbol{x} \in \mathbb{R}^n$ given corrupted versions $\boldsymbol{y} \in \mathbb{R}^d$. Usually, this corruption process is expressed as

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

where $\mathcal{A}$ is the (possibly nonlinear) **forward measurement** operator, and $\mathbf{n}$ is some noise, e.g. Gaussian.



(a) Image with missing data (b) Inpainted image

Figure: **Inpainting** (Rout et al. [6])

Background: Latent Diffusion Models (LDMs)

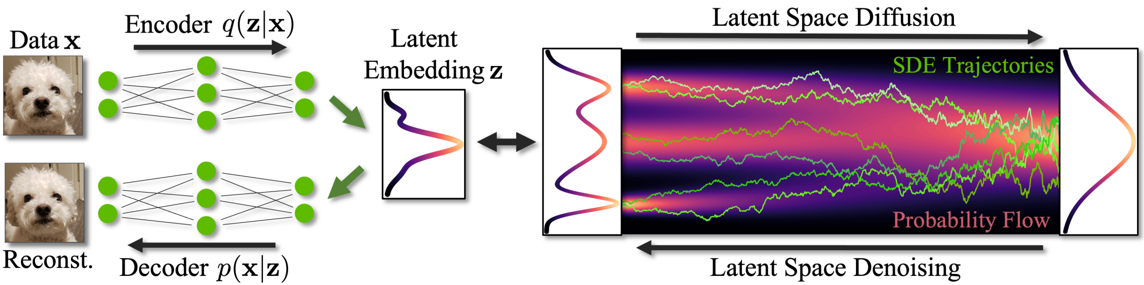


Figure: Latent Diffusion scheme (Source NeurIPS 2023 Tutorial)

Some Diffusion-based inverse solvers

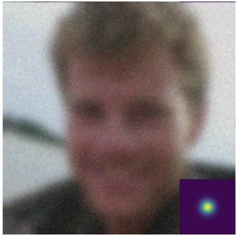


Figure: Blurry image

$\Rightarrow$

Figure: Denoised image

1. **Diffusion Models for Plug-and-Play Image Restoration (DiffPIR)** [8] adopts a plug-and-play method where the prior adopted is a DM.
2. **Diffusion Posterior Sampling (DPS)** [2] adopts a Bayesian approach to compute the posterior probability $p(x_t|y)$ at each step of the diffusion process.
3. **Posterior Sampling with Latent Diffusion (PSLD)** [6] implements some "tricks" to adapt DPS to a **LDM**
4. **P2L** [3] optimizes the prompt c while sampling
5. **TReg** [5] optimizes the null prompt $c_\emptyset$ while sampling, exploiting the Classifier Free Guidance (CFG) scheme

Drawbacks of LDM-based algorithms

Current State of the Art (SOTA) LDM-based methods present the following problems:

- **Elevated number of steps**, meaning that on average ~ 1000 Neural Function Evaluations (NFEs) are needed
- **High memory usage** since methods like DPS require to compute gradients, and in the latent space, this also involves the $\mathcal{D}$ and/or $\mathcal{E}$: $\nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{y}|\boldsymbol{z}_t) \propto \nabla_{\boldsymbol{z}_t} \|\mathcal{A}\mathcal{D}(\boldsymbol{z}_0^{(t)}(\boldsymbol{z}_t)) - \boldsymbol{y}\|_2^2$

These two reasons prevent scalability and force to "low" resolutions ($\leq 512^2$).

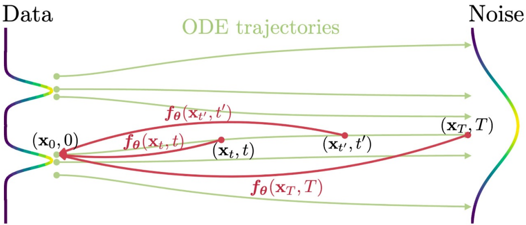
Latent Consistency Models (LCMs)

Consistency Models (CMs) [7] accelerate **sampling from diffusion models**. They satisfy:

Definition (Consistency function)

Given a small $\eta > 0$ and a trajectory $\{\mathbf{x}_t\}_{t \in [\eta, T]}$ of the PF-ODE, we define the *consistency function* as $G_\theta : (\mathbf{x}_t, t) \to \mathbf{x}_\eta$.

ensuring **self-consistency** across timesteps.



Probability flow ODE:

$$\mathrm{d}\mathbf{x} = \left[\boldsymbol{f}(\mathbf{x}, t) - \frac{1}{2} g^2(t) \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \right] \mathrm{d}t$$

$$\Big\| \quad \{p_t(\mathbf{x})\}_{t \in [0, T]}$$

$$\mathrm{d}\mathbf{x} = \left[\boldsymbol{f}(\mathbf{x}, t) - g^2(t) \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \right] \mathrm{d}t + g(t)\, \mathrm{d}\mathbf{w}$$

Latent Consistency Models (LCMs)

Consistency Models (CMs) [7] accelerate **sampling from diffusion models**. They satisfy:

Definition (Consistency function)

Given a small $\eta > 0$ and a trajectory $\{\mathbf{x}_t\}_{t \in [\eta, T]}$ of the PF-ODE, we define the *consistency function* as $G_\theta : (\mathbf{x}_t, t) \rightarrow \mathbf{x}_\eta$.

ensuring **self-consistency** across timesteps.

Latent Consistency Models (LCMs) extend this idea to the **latent space** of a pre-trained **LDM**:

- Learn **single-step mapping** from noisy latents $\mathbf{z}_t$ to clean latents $\mathbf{z}_0$.
- High **sample quality** with very few steps $(N = 1 - 4)$.
- Given timesteps $t_1 > t_2 > \cdots > t_{N-1} > \eta$, the multistep consistency sampling process is

$$\hat{\mathbf{z}}_T \sim \mathcal{N}(0, \mathrm{Id}), \quad \mathbf{z} = G_\theta(\hat{\mathbf{z}}_T, T)$$

$$\text{For } n = 1 \text{ to } N-1:$$

$$\hat{\mathbf{z}}_{t_n} = \mathbf{z} + \sqrt{(1-\alpha_{t_n}) - (1-\alpha_\eta)}\epsilon \quad \text{with} \quad \epsilon \sim \mathcal{N}(0, \mathrm{Id})$$

$$\mathbf{z} = G_\theta(\hat{\mathbf{z}}_{t_n}, t_n),$$

LATINO: Gradient-Free Posterior Sampling with LCMs

We introduce **LATINO**, a novel Plug-and-Play (PnP) method leveraging pre-trained text-to-image Latent Consistency Models (LCMs). LATINO samples from the posterior $p(\boldsymbol{x} \mid \boldsymbol{y}, c)$:

- **Gradient-free** sampling with very few function evaluations (only 8 NFEs).
- Efficient scaling to high-resolution images ($\geq 1024^2$) with low GPU memory usage.
- Naturally **prompt-conditioned**, enabling semantic control by users.
- Based on a novel Langevin-inspired PnP approach designed specifically for LCMs, employing **stochastic auto-encoders (SAE)**.

We further propose **LATINO-PRO**, which integrates automatic prompt optimization via stochastic proximal gradient methods:

- Automatically finds optimal prompts: $\hat{c}(\boldsymbol{y}) = \arg\max_{c \in \mathbb{R}^k} p(\boldsymbol{y} \mid c)$.
- Corrects incomplete or misleading prompts efficiently.
- Still requires minimal computational effort (only 68 NFEs).

LATINO: Langevin Diffusion Formulation

Consider sampling from the posterior distribution via an **overdamped Langevin diffusion**:

$$\mathrm{d}\boldsymbol{x}_s = \nabla \log p(\boldsymbol{y}|\boldsymbol{x}_s)\mathrm{d}s + \nabla \log p(\boldsymbol{x}_s|c)\mathrm{d}s + \sqrt{2}\mathrm{d}\boldsymbol{w}_s, \tag{1}$$

where $\boldsymbol{w}_s$ is an n-dimensional Brownian motion. Under mild assumptions, the process converges exponentially fast to $p(\boldsymbol{x}|\boldsymbol{y}, c)$ as $s \to \infty$. Exact solutions are generally intractable; hence approximations are required (e.g., Euler-Maruyama leading to ULA).

Limitations of ULA:

- Explicit Euler step integration.
- Stability constraints: small step-size δ required.
- Potentially large discretization bias.

LATINO: Split Integration Approach

LATINO employs a **split integration approximation**:

$$\boldsymbol{u} = \tilde{\mathbf{x}}_0 + \int_0^\delta \nabla \log p(\tilde{\mathbf{x}}_s|c)\mathrm{d}s + \sqrt{2}\mathrm{d}\mathbf{w}_s, \quad \tilde{\mathbf{x}}_0 = \mathbf{x}_k, \tag{2}$$

$$\mathbf{x}_{k+1} = \boldsymbol{u} + \delta\nabla \log p(\boldsymbol{y}|\mathbf{x}_{k+1}). \tag{3}$$

Advantages of this splitting:

- **Accuracy**: No discretization bias in the prior step.
- **Stability**: Implicit Euler integration ensures numerical stability for all $\delta > 0$.
- **Efficiency**: The implicit Euler step translates into a tractable proximal step:

  $$\mathbf{x}_{k+1} = \mathrm{prox}_{\delta g_y}(\tilde{\mathbf{x}}_{k+1}), \quad g_y(\boldsymbol{x}) = -\log p(\boldsymbol{y}|\boldsymbol{x})$$

  efficiently solvable in common inverse problems (e.g., deblurring, super-resolution).

LATINO approximates the prior step by a SAE derived from a CM, maintaining computational feasibility.

LATINO pipeline

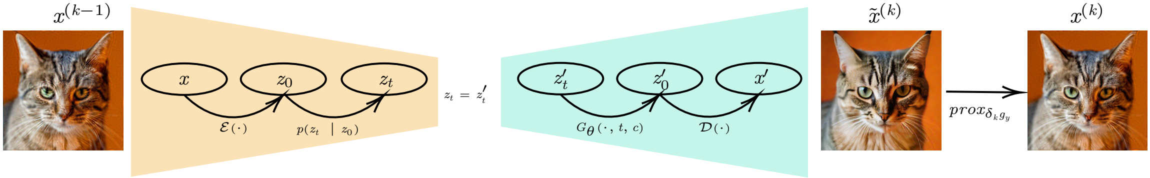


Figure: One step of the LATINO solver, a discretization of the Langevin SDE which targets the posterior $p(\mathbf{x}|\mathbf{y}, c)$. The current iterate $\mathbf{x}_k$ is encoded by the VAE encoder and propagated forward via a noising diffusion kernel $p(\mathbf{z}_t|\mathbf{z}_0)$. This process is then reversed via the latent consistency model and the VAE decoder, followed by the proximal operator to involve the likelihood $p(\mathbf{y}|\mathbf{x})$.

Auto-Encoding Stable Diffusion

Goal: Construct a stochastic auto-encoder $(\mathfrak{E}_t, \mathfrak{D}_{t,c})$ contracting random variables towards $p(\mathbf{x}|c)$ with $p(\mathbf{x}|c)$ as a fixed point.

Stochastic Encoder ($\mathfrak{E}_t$):

$$\mathbf{z}_t|\mathbf{x} \sim \mathcal{N}(\sqrt{\alpha_t}\mathcal{E}(\mathbf{x}), (1-\alpha_t)\mathsf{Id}_k),$$

obtained by applying deterministic encoder $\mathcal{E}$ followed by the forward SDE $d\mathbf{x}_t = -\frac{\beta_t}{2}\mathbf{x}_t dt + \sqrt{\beta_t}d\mathbf{w}$.

Decoder ($\mathfrak{D}_{t,c}$): Maps latent $\mathbf{z}_t'$ to the ambient space:

$$\mathbf{x}' = \mathcal{D}(G_\theta(\mathbf{z}_t', t, c)).$$

Contraction and Fixed Point:

- If $\mathbf{x} \sim p(\mathbf{x}|c)$ exactly, encoding via $\mathfrak{E}_t$ and subsequent decoding through $\mathfrak{D}_{t,c}$ yields $\mathbf{x}'$ distributed as $p(\mathbf{x}|c)$ (fixed point property).
- For distributions different from $p(\mathbf{x}|c)$, $(\mathfrak{E}_t, \mathfrak{D}_{t,c})$ progressively contracts samples toward $p(\mathbf{x}|c)$.

Contraction Dynamics

Role of parameter t:

- Large t: Strong contraction towards $p(\mathbf{x}|c)$; behaves as a standard generative model.
- Small t: Approximate identity map $(\mathcal{E}, \mathcal{D})$, limited contraction.
- Intermediate t: Balances identity preservation and contraction toward target distribution.

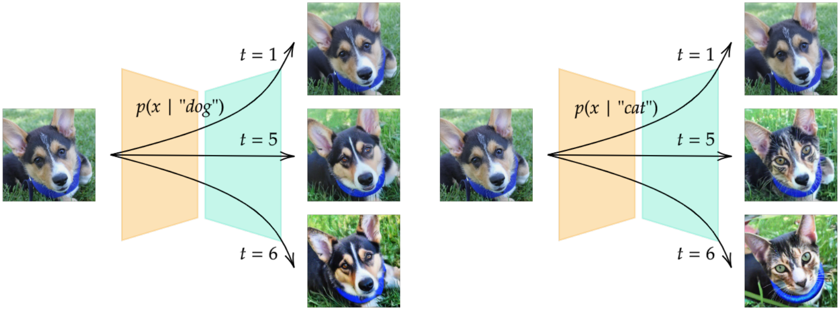


Figure: SAE applied to images in and out of distribution for different values of t, illustrating contraction towards $p(\mathbf{x}|c)$.

Prompt optimization via Maximum Marginal Likelihood Estimation (MMLE):
LATINO-PRO addresses the challenge of selecting optimal text prompts c by maximizing the marginal likelihood:

$$\hat{c}(\boldsymbol{y}) = \arg\max_{c \in \mathbb{R}^k} p(\boldsymbol{y} \mid c), \quad p(\boldsymbol{y} \mid c) = \mathbb{E}_{\boldsymbol{z} \mid c}[p(\boldsymbol{y}|\boldsymbol{z})]$$

Motivation:

- In ill-posed inverse problems, the likelihood $p(\boldsymbol{y}|\boldsymbol{z})$ is often weakily informative, thus the prior $p(\boldsymbol{z}|c)$ (encoded by the generative model) becomes critical.
- Directly solving MMLE is computationally intractable; hence, stochastic optimization methods are required.

LATINO-PRO: Stochastic Prompt Optimization

LATINO-PRO uses a **Stochastic Approximation Proximal Gradient (SAPG)** scheme:

$$c_{m+1} = \Pi_C \left[c_m + \gamma_m \nabla_c \log p(\boldsymbol{y}|c_m) \right],$$

where γ_m is a sequence of decreasing positive step-sizes and $C \subset \mathbb{R}^k$ is a convex set of admissible values for c. From **Fisher's identity** we get

$$\begin{aligned} \nabla_c \log p(\boldsymbol{y} \mid c_m) &= \mathbb{E}_{\boldsymbol{z}|\boldsymbol{y},c_m} [\nabla_c \log p(\boldsymbol{y}, \boldsymbol{z} \mid c_m)], \\ &= \mathbb{E}_{\boldsymbol{z}|\boldsymbol{y},c_m} [\nabla_c \log p(\boldsymbol{z} \mid c_m)], \end{aligned}$$

which motivates the approximation using samples from LATINO:

$$\nabla_c \log p(\boldsymbol{y}|c_m) \approx \nabla_c \log p(\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(N)}|c_m).$$

Key Practical Considerations:

- Automatic differentiation (AD) in latent space makes gradient computation tractable.
- Starting from a descriptive prompt (e.g., "a sharp photo of a dog") accelerates convergence and improves sample quality.
- Early stopping of prompt optimization provides regularization and improves results.

Quantitative results: AFHQ

		Deblur (Gaussian)		SR$\times 16$	
Method	**NFE↓**	**FID↓**	**PSNR↑**	**FID↓**	**PSNR↑**
LATINO-PRO	<u>68</u>	**18.37**	**26.82**	**30.40**	**21.52**
LATINO	**8**	<u>20.03</u>	<u>26.25</u>	42.14	<u>20.05</u>
P2L [3]	2000	85.80	20.96	121.7	19.99
TReg [5]	200	35.47	21.13	<u>37.13</u>	19.60
LDPS	1000	64.88	22.60	101.13	17.34
PSLD [6]	1000	125.5	20.52	113.4	16.48

Table: Results for Gaussian Deblurring with $\sigma = 5.0$, and $\times 16$ super-resolution, both with noise $\sigma_y = 0.01$ on the AFHQ-512 val dataset. Our LATINO and LATINO-PRO models are compared to recent state-of-the-art methods. Prompts: `a sharp photo of a dog` (resp. `a cat`) **Bold**: best, <u>underline</u>: second best.

Qualitative results: AFHQ

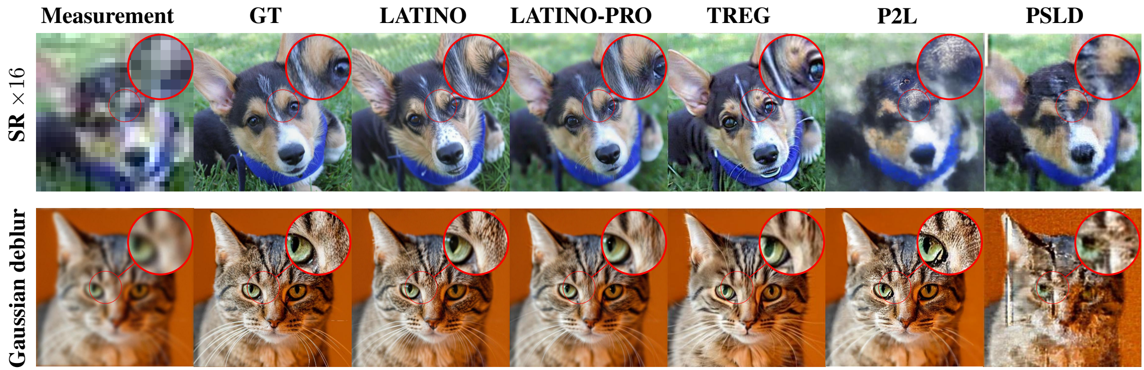


Figure: Results for Gaussian Deblurring with $\sigma = 5.0$, and $\times 16$ super-resolution, both with noise $\sigma_y = 0.01$ on the AFHQ-512 val dataset. Our LATINO and LATINO-PRO models are compared to recent state-of-the-art methods. Prompts: `a sharp photo of a dog` (resp. `a cat`).

Quantitative results: FFHQ

		Deblur (Gaussian)			Deblur (Motion)			SR$\times 8$		
Method	**NFE↓**	**FID↓**	**PSNR↑**	**LPIPS↓**	**FID↓**	**PSNR↑**	**LPIPS↓**	**FID↓**	**PSNR↑**	**LPIPS↓**
LATINO-PRO	<u>68</u>	<u>31.98</u>	**29.11**	**0.292**	<u>27.80</u>	<u>27.14</u>	**0.301**	40.95	26.58	0.355
LATINO	**8**	33.94	<u>28.95</u>	<u>0.296</u>	29.17	26.88	0.318	37.13	26.22	0.356
P2L [3]	2000	**30.62**	26.97	0.299	28.34	**27.23**	<u>0.302</u>	**31.23**	<u>28.55</u>	**0.290**
LDPS	1000	45.89	27.82	0.334	58.66	26.19	0.382	36.81	**28.78**	<u>0.292</u>
PSLD [6]	1000	41.04	28.47	0.320	47.71	27.05	0.348	36.93	26.62	0.335
LDIR [4]	1000	35.61	25.75	0.341	**24.40**	24.40	0.376	<u>36.04</u>	25.79	0.345

Table: Results for Gaussian deblurring with $\sigma = 3.0$, motion deblurring, and $\times 8$ super-resolution, all with noise $\sigma_y = 0.01$ on the FFHQ-512 val dataset. Our LATINO and LATINO-PRO models are compared to recent state-of-the-art methods. Prompt: `a sharp photo of a face`. **Bold**: best, <u>underline</u>: second best.

Qualitative results: Food101 dataset

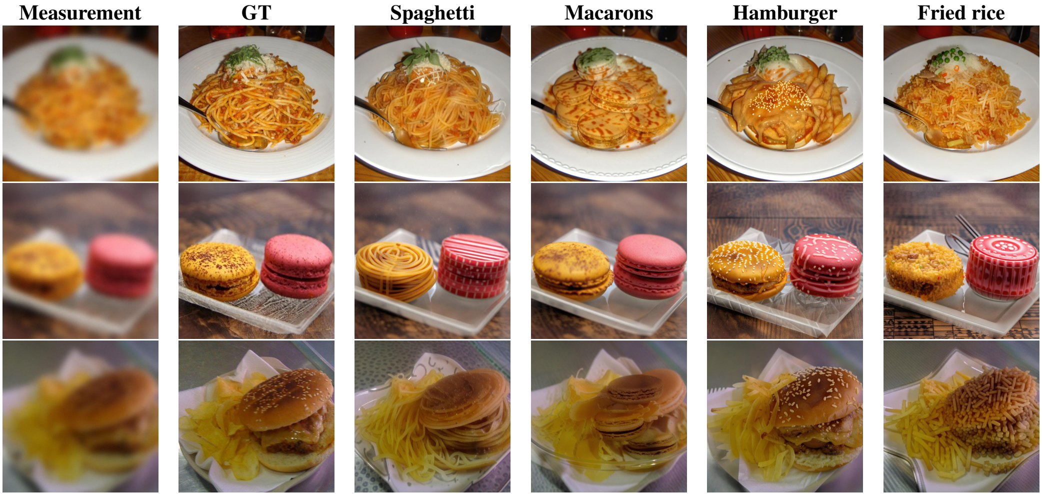
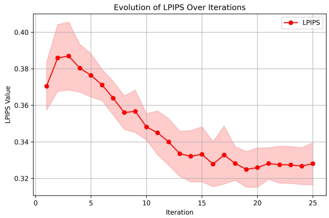


Figure: Qualitative results of the 8-steps LATINO on Food101 dataset [1] for semantic shift task

Prompt tuning: experimental results

Figure: Effect of prompt optimization on the AFHQ-dogs val dataset. Initial prompt: `a sharp photo of a cat.`



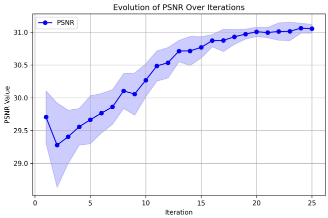


Figure: Metrics evolution during LATINO-PRO iterations. Initial prompt: `a sharp photo of a cat.`

Prompt tuning: experimental results

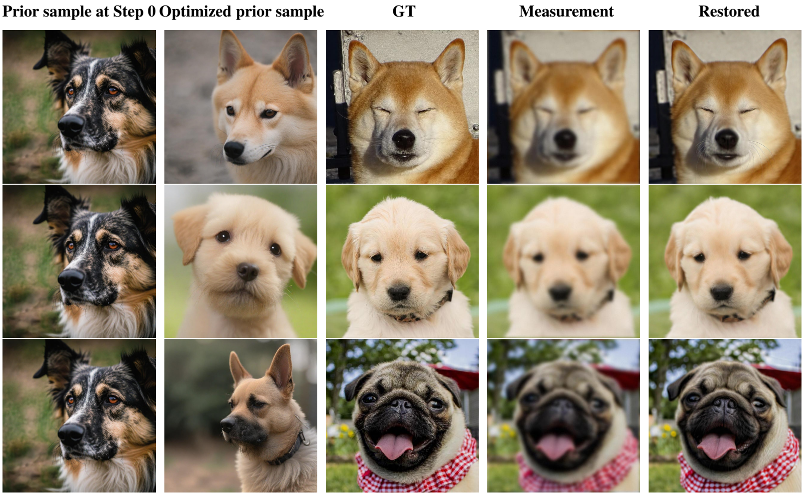


Figure: Effect of prompt optimization on the AFHQ-dogs val dataset. Initial prompt: `a sharp photo of a dog`.

Memory and time consumption

We provide an exhaustive comparison of our models with respect to current SOTA in terms of **memory consumption** and **time** needed. For algorithms TReg and P2L for which the official code release is not available, we implemented versions of the algorithms starting from the pseudocodes as described in [3, 5].

Method	GPU (Gb)	Time (s)	Resolution
LATINO	13.6	5.53	1024^2
LATINO-PRO	23.4	48.8	1024^2
TReg	~ 6.40	40.5	512^2
P2L	~ 10.6	600	512^2
LDPS	9.51	176	512^2
PSLD	10.3	185	512^2
LDPS-XL	42.5	694	1024^2
PSLD-XL	46.7	1044	1024^2
TReg-XL	~ 37.02	~ 240	1024^2
P2L-XL	~ 43.3	~ 3122	1024^2

Table: GPU Memory and Time consumption comparison

Future perspectives

- Analyze the **theoretical properties** of LATINO and LATINO-PRO, with special attention to **non-asymptotic** convergence results.
- Development of strategies to **automatically adjust** the parameters of LATINO and LATINO-PRO.
- Explore strategies for **decoding the prompt** embedding to reveal the optimized text prompt.
- Application to **blind** inverse problems.

Thank you for the attention

Thank you for the attention !

References I

L. Bossard, M. Guillaumin, and L. V. Gool.
Food-101 - mining discriminative components with random forests.
In *European Conference on Computer Vision*, 2014.

H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye.
Diffusion posterior sampling for general noisy inverse problems.
In *The Eleventh International Conference on Learning Representations*, 2022.

H. Chung, J. C. Ye, P. Milanfar, and M. Delbracio.
Prompt-tuning latent diffusion models for inverse problems.
In *Forty-first International Conference on Machine Learning*, 2024.

L. He, H. Yan, M. Luo, H. Wu, K. Luo, W. Wang, W. Du, H. Chen, H. Yang, Y. Zhang, and J. Lv.
Fast and stable diffusion inverse solver with history gradient update, 2024.

References II

J. Kim, G. Y. Park, H. Chung, and J. C. Ye.
Regularization by texts for latent diffusion inverse solvers.
In *The Thirteenth International Conference on Learning Representations*, 2025.

L. Rout, N. Raoof, G. Daras, C. Caramanis, A. Dimakis, and S. Shakkottai.
Solving linear inverse problems provably via posterior sampling with latent diffusion models.
Advances in Neural Information Processing Systems, 36:49960–49990, 2023.

Y. Song, P. Dhariwal, M. Chen, and I. Sutskever.
Consistency models.
In *International Conference on Machine Learning*, 2023.

Y. Zhu, K. Zhang, J. Liang, J. Cao, B. Wen, R. Timofte, and L. V. Gool.
Denoising diffusion models for plug-and-play image restoration.
2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1219–1229, 2023.