Equivariant Sampling for Improving Diffusion Model-based Image Restoration

Supplementary Material

Our code is available in the supplementary ("EquS.zip" file). We organize our supplementary materials as follows:

A. Detailed explanations of the equations.

A.1. Section 4 Equation (7)

In the case of VP-SDE or DDPM sampling, $p_\theta(\mathbf{x}_0|\mathbf{x}_t)$ has the unique posterior mean

$$\mathbf{x}_{0|t} := \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)). \tag{1}$$

By plugging (72) into (1), one can retrieve the empirical Bayes optimal posterior mean [4, 17, 23] $\mathbf{x}_{0|t}$ from $p_\theta(\mathbf{x}_0|\mathbf{x}_t)$ using either $\mathbf{s}_\theta(\mathbf{x}_t, t)$ or $\epsilon_\theta(\mathbf{x}_t, t)$.

Explanation For the perturbation kernels $\{q(\mathbf{x}_i|\mathbf{x}_{i-1})\}_{i=1}^N$ used in the DDPM, the associated discrete Markov chain is defined as

$$\mathbf{x}_i = \sqrt{1-\beta_i}\mathbf{x}_{i-1} + \sqrt{\beta_i}\epsilon_{i-1}, \quad i = 1, \cdots, N, \tag{2}$$

where $\epsilon_{i-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. To obtain the limit of this Markov chain when $N \to \infty$, we could define an auxiliary set of noise scales $\{\boldsymbol{\beta}_i = N\beta_i\}_{i=1}^N$, and re-write (2) as below

$$\mathbf{x}_i = \sqrt{1-\frac{\boldsymbol{\beta}_i}{N}}\mathbf{x}_{i-1} + \sqrt{\frac{\boldsymbol{\beta}_i}{N}}\epsilon_{i-1}, \quad i = 1, \cdots, N. \tag{3}$$

In the limit of $N \to \infty$, $\{\boldsymbol{\beta}_i\}_{i=1}^N$ becomes a function $\beta(t)$ indexed by $t \in [0, 1]$. Let $\beta\left(\frac{i}{N}\right) = \boldsymbol{\beta}_i$, $\mathbf{x}(\frac{i}{N}) = \mathbf{x}_i$, $\epsilon(\frac{i}{N}) = \epsilon_i$. We can rewrite the Markov chain in (3) as the following with $\Delta t = \frac{1}{N}$ and $t \in \{0, 1, \cdots, \frac{N-1}{N}\}$, *i.e.*,

$$\begin{aligned}\mathbf{x}(t+\Delta t) &= \sqrt{1-\beta(t+\Delta t)\Delta t}\mathbf{x}(t) + \sqrt{\beta(t+\Delta t)\Delta t}\epsilon(t)\\ &\approx \mathbf{x}(t) - \frac{1}{2}\beta(t+\Delta t)\Delta t\mathbf{x}(t) + \sqrt{\beta(t+\Delta t)\Delta t}\epsilon(t)\\ &\approx \mathbf{x}(t) - \frac{1}{2}\beta(t)\Delta t\mathbf{x}(t) + \sqrt{\beta(t)\Delta t}\epsilon(t),\end{aligned} \tag{4}$$

where the approximate equality holds when $\Delta t \ll 1$. Therefore, in the limit of $\Delta t \to 0$, (4) converges to the following VP SDE, *i.e.*,

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}_t\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}. \tag{5}$$

This scenario precisely corresponds to the case in (70) when $\mathbf{f}(\mathbf{x}_t, t) = -\frac{1}{2}\beta(t)\mathbf{x}_t$ and $g(t) = \sqrt{\beta(t)}$.

In [23], they propose the following reverse-time SDE, *i.e.*,

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}_t, t) - g^2(t)\nabla_{\mathbf{x}}\log p_t(\mathbf{x}_t)]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}, \tag{6}$$

with a similar functional form, which gives the following iteration rule for $i \in \{0, 1, \cdots, N-1\}$:

$$\mathbf{x}_i = \mathbf{x}_{i+1} - \mathbf{f}(\mathbf{x}_{i+1}, i+1) + g^2(i+1)\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + g(i+1)\epsilon_{i+1}, \tag{7}$$

where the trained score-based model $\mathbf{s}_\theta(\mathbf{x}_i, i)$ is conditioned on iteration number i.

The ancestral sampling of DDPM [12] ((65)) matches its reverse diffusion counterpart when $\beta_i \to 0$ for all i (which happens when $\Delta t \to 0$ since $\beta_i = \boldsymbol{\beta}_i \Delta t$, matching the same case in (4)).

To achieve this, let us start from the sampling process in DDPM ((65)), *i.e.*,

$$\mathbf{x}_i = \frac{1}{\sqrt{\alpha_{i+1}}}\left(\mathbf{x}_{i+1} - \frac{\beta_{i+1}}{\sqrt{1-\bar{\alpha}_{i+1}}}\epsilon_\theta(\mathbf{x}_{i+1}, i+1)\right) + \sqrt{\beta_{i+1}}\epsilon_{i+1}. \tag{8}$$

Recall that $\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) \simeq -\dfrac{\epsilon_\theta(\mathbf{x}_{i+1}, i+1)}{\sqrt{1-\bar{\alpha}_{i+1}}}$, $\alpha_{i+1} = 1 - \beta_{i+1}$, we further have

$$\mathbf{x}_i = \frac{1}{\sqrt{1-\beta_{i+1}}}(\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1)) + \sqrt{\beta_{i+1}}\epsilon_{i+1} \tag{9}$$

$$= \left(1 + \frac{1}{2}\beta_{i+1} + o(\beta_{i+1})\right)(\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1)) + \sqrt{\beta_{i+1}}\epsilon_{i+1} \tag{10}$$

$$\approx \left(1 + \frac{1}{2}\beta_{i+1}\right)(\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1)) + \sqrt{\beta_{i+1}}\epsilon_{i+1} \tag{11}$$

$$= \left(1 + \frac{1}{2}\beta_{i+1}\right)\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + \frac{1}{2}\beta_{i+1}^2\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + \sqrt{\beta_{i+1}}\epsilon_{i+1} \tag{12}$$

$$\approx \left(1 + \frac{1}{2}\beta_{i+1}\right)\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + \sqrt{\beta_{i+1}}\epsilon_{i+1}. \tag{13}$$

This corresponds to the case where $\mathbf{f}(\mathbf{x}_i, i) = -\dfrac{1}{2}\beta_i\mathbf{x}_i$ and $g(i) = \sqrt{\beta_i}$ are substituted into the equation (7), *i.e.*,

$$\mathbf{x}_i = \mathbf{x}_{i+1} - \mathbf{f}(\mathbf{x}_{i+1}, i+1) + g^2(i+1)\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + g(i+1)\epsilon_{i+1} \tag{14}$$

$$= \mathbf{x}_{i+1} + \frac{1}{2}\beta_{i+1}\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + \sqrt{\beta_{i+1}}\epsilon_{i+1} \tag{15}$$

$$= (1 + \frac{1}{2}\beta_{i+1})\mathbf{x}_{i+1} + \beta_{i+1}\mathbf{s}_\theta(\mathbf{x}_{i+1}, i+1) + \sqrt{\beta_{i+1}}\epsilon_{i+1}. \tag{16}$$

Therefore, the original ancestral sampler of (65) is essentially a different discretization to the same reverse-time SDE. This unifies the sampling method in DDPM [12] as a numerical solver to the reverse-time VP-SDE [23]. We show that DDPM [11] and VP-SDE [23] could have the same sampling process ((13) and (16)), *i.e.*,

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) := \mathcal{N}\left(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2\mathbf{I}\right), \tag{17}$$

where $\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) := \dfrac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \dfrac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$ and $\sigma_t^2 := \dfrac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$. However, $\mathbf{x}_0$ is intractable but can be predicted by neural networks. By plugging (72) into (1), one can retrieve the empirical Bayes optimal posterior mean $\mathbf{x}_{0|t}$ from $p_\theta(\mathbf{x}_0|\mathbf{x}_t)$ using either $\mathbf{s}_\theta(\mathbf{x}_t, t)$ or $\epsilon_\theta(\mathbf{x}_t, t)$, *i.e.*,

$$\mathbf{x}_{0|t} := \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)) \tag{18}$$

$$\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, t))) \tag{19}$$

$$\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{(1-\bar{\alpha}_t)}\epsilon_\theta(\mathbf{x}_t, t)). \tag{20}$$

This process can also be interpreted as performing Gaussian denoising using Tweedie's formula [8], as suggested in [4, 17]. For further details, please refer to the corresponding papers. In generation tasks, one only need to train $\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)$ (*i.e.*, to train $\epsilon_\theta(\mathbf{x}_t, t)$) to execute the reverse process as described in (17).

A.2. Section 4 Equation (8)

In DMIR, one can obtain the posterior mean $\tilde{\mathbf{x}}_{0|t}$ by replacing $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ with $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y})$, i.e.,

$$\tilde{\mathbf{x}}_{0|t} := \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, y] = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})). \tag{21}$$

Explanation Following the previous procedure, the original ancestral sampler of (65) is essentially a different discretization to the same reverse-time SDE. This unifies the sampling method in DDPM [12] as a numerical solver to the reverse-time VP-SDE [23]. We show that DDPM [11] and VP-SDE [23] could have the same sampling process ((13) and (16)) as implied in (17).

In DMIR, one can obtain the posterior mean $\tilde{\mathbf{x}}_{0|t}$ from $p_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{y})$ by replacing $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ with $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y})$, *i.e.*,

$$\begin{aligned}
\tilde{\mathbf{x}}_{0|t} := \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, y] &= \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})) && (22)\\
&\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, \mathbf{y}, t))) && (23)\\
&\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{(1-\bar{\alpha}_t)}\epsilon_\theta(\mathbf{x}_t, \mathbf{y}, t)). && (24)
\end{aligned}$$

Using $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y}) = \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ also allows us to obtain $\tilde{\mathbf{x}}_{0|t}$ from $p_\theta(\mathbf{x}_0|\mathbf{x}_t, \mathbf{y})$

$$\begin{aligned}
\tilde{\mathbf{x}}_{0|t} &= \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})) && (25)\\
&= \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)(\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t))) && (26)\\
&\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)\mathbf{s}_\theta(\mathbf{x}_t, \mathbf{y}, t)) + \frac{1-\bar{\alpha}_t}{\sqrt{\bar{\alpha}_t}}\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) && (27)\\
&\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + \sqrt{(1-\bar{\alpha}_t)}\epsilon_\theta(\mathbf{x}_t, \mathbf{y}, t)) + \frac{1-\bar{\alpha}_t}{\sqrt{\bar{\alpha}_t}}\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) && (28)
\end{aligned}$$

Assuming the availability of problem-specific scores for all noise levels, *i.e.*, $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})$, diffusion models can address IR tasks via (17). For problem-specific DMIR, we can directly train $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})$ (*i.e.*, train $\epsilon_\theta(\mathbf{x}_t, \mathbf{y}, t)$). In problem-agnostic DMIR, we only have $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$, thus one need to approximate $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ in (28).

A.3. Section 4 Equation (9) and (10)

However, we only have $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ in problem-agnostic DMIR, thus to approximate $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$, various methods can be broadly classified into

$$\begin{aligned}
\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) &\simeq \mathbf{A}^\dagger(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}), && (29)\\
&\simeq \mathbf{A}^\top(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}). && (30)
\end{aligned}$$

where $\mathbf{A}^\dagger$ denotes the pseudo-inverse of $\mathbf{A}$, $\mathbf{A}^\top$ represents the transpose of $\mathbf{A}$.
Explanation In this part, we discuss various scenarios in separate sections and summarize these approaches in Table 1.
DDNM [26]. They first estimate a $\mathbf{x}_0$ from $\mathbf{x}_t$ and the predicted noise $\epsilon_t = \epsilon_\theta(\mathbf{x}_t, t)$. They denote the estimated $\mathbf{x}_0$ at time-step t as $\mathbf{x}_{0|t}$, which can be formulated as:

$$\mathbf{x}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{(1-\bar{\alpha}_t)}\epsilon_\theta(\mathbf{x}_t, t)). \tag{31}$$

where they further fix the range-space (*i.e.*, $\mathbf{A}^\dagger\mathbf{A}\mathbf{x}$) as $\mathbf{A}^\dagger\mathbf{y}$ and leave the null-space (*i.e.*, $(\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\mathbf{x}$) unchanged, yielding a rectified estimation $\tilde{\mathbf{x}}_{0|t}$ as:

$$\tilde{\mathbf{x}}_{0|t} = \mathbf{A}^\dagger\mathbf{y} + (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A})\mathbf{x}_{0|t}, \tag{32}$$

Guidance Expression	Training on $(\mathbf{x}_t, y)$	$\mathbf{x}_t \to y$ differentiable	No gradient	Related Works
$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}\|\mathbf{x}_t)$	✘ need training	✘ required	✘ need	—
$\nabla_{\mathbf{x}_t} \|\mathbf{y} - \mathbf{A}\mathbf{x}_{0\|t}\|_2^2$	✔ no training	✘ required	✘ need	DiffPIR [31]
$\sqrt{\bar{\alpha}_t} \frac{\partial \mathbf{x}_{0\|t}}{\partial \mathbf{x}_t} \mathbf{A}^\top (\mathbf{y} - \mathbf{A}\mathbf{x}_{0\|t})$	✔ no training	✔ not required	✘ need	DPS [4], ZAPS [1]
$\sqrt{\bar{\alpha}_t} \frac{\partial \mathbf{x}_{0\|t}}{\partial \mathbf{x}_t} \mathbf{A}^\dagger (\mathbf{y} - \mathbf{A}\mathbf{x}_{0\|t})$	✔ no training	✔ not required	✘ need	ΠGDM [21], CDDB (deep) [3]
$\mathbf{A}^\dagger (\mathbf{y} - \mathbf{A}\mathbf{x}_{0\|t})$	✔ no training	✔ not required	✔ no need	DDRM [16], DDNM [26], DPPG [9], DeqIR [3]
$\mathbf{A}^\top (\mathbf{y} - \mathbf{A}\mathbf{x}_{0\|t})$	✔ no training	✔ not required	✔ no need	CDDB [3], DPPG [9]

Table 1. Comparison of different guidance methods.

which could be rewritten as

$$\begin{aligned} \tilde{\mathbf{x}}_{0|t} &= \mathbf{A}^\dagger \mathbf{y} + (\mathbf{I} - \mathbf{A}^\dagger \mathbf{A})\mathbf{x}_{0|t} && (33) \\ &= \mathbf{x}_{0|t} + \mathbf{A}^\dagger (\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}) && (34) \end{aligned}$$

This gives the first case, *i.e.*,

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \simeq \mathbf{A}^\dagger (\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}). \tag{35}$$

DDRM [16]. In DDNM [26], they show that for the noise-free situation, the final result $\mathbf{x}_0$ of DDRM is essentially yielded through a special case of DDNM. We here give a brief deduction.

For noisy linear inverse problem $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$ where $\mathbf{n} \sim \mathcal{N}(0, \sigma_{\mathbf{y}}^2)$, DDRM first uses SVD to decompose $\mathbf{A}$ as $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, then use $\bar{\mathbf{y}} = \mathbf{\Sigma}^\dagger \mathbf{U}^\top \mathbf{y}$ and $\bar{\mathbf{x}}_{0|t} = \mathbf{V}^\top \mathbf{x}_{0|t}$ for derivation. Each element in $\bar{\mathbf{y}}$ and $\bar{\mathbf{x}}_{0|t}$ corresponds to a singular value in $\mathbf{\Sigma}$ (the nonexistent singular value is defined as 0), hence it is possible to modify $\mathbf{x}_{0|t}$ element-wisely according to each singular value. Then one can yield the final result $\mathbf{x}_0$ by $\mathbf{x}_0 = \mathbf{V}\bar{\mathbf{x}}_0$.

Specifically, when $t = 0$ and $\sigma_{\mathbf{y}} = 0$, we can rewrite the formula of the i-th element of $\bar{\mathbf{x}}_0$ as:

$$\bar{\mathbf{x}}_0^{(i)} = \begin{cases} \bar{\mathbf{x}}_{0|1}^{(i)}, & s_i = 0 \\ \bar{\mathbf{y}}^{(i)}, & s_i \neq 0 \end{cases} \tag{36}$$

To simplify the representation, we define a diagonal matrix $\mathbf{\Sigma}_1$:

$$\mathbf{\Sigma}_1^{(i)} = \begin{cases} 0, & s_i = 0 \\ 1, & s_i \neq 0 \end{cases} \tag{37}$$

Then we can rewrite $\bar{\mathbf{x}}_0$ as

$$\bar{\mathbf{x}}_0 = \mathbf{\Sigma}_1 \bar{\mathbf{y}} + (\mathbf{I} - \mathbf{\Sigma}_1)\bar{\mathbf{x}}_{0|1} \tag{38}$$

and yield the result $\mathbf{x}_0$ by left multiplying $\mathbf{V}$:

$$\mathbf{x}_0 = \mathbf{V}\bar{\mathbf{x}}_0 = \mathbf{V}\mathbf{\Sigma}_1 \bar{\mathbf{y}} + \mathbf{V}(\mathbf{I} - \mathbf{\Sigma}_1)\bar{\mathbf{x}}_{0|1} \tag{39}$$

This result is essentially a special range-null space decomposition:

$$\begin{aligned} \mathbf{x}_0 &= \mathbf{V}\mathbf{\Sigma}_1 \bar{\mathbf{y}} + \mathbf{V}(\mathbf{I} - \mathbf{\Sigma}_1)\bar{\mathbf{x}}_{0|1} \\ &= \mathbf{V}\mathbf{\Sigma}_1 \mathbf{\Sigma}^\dagger \mathbf{U}^\top \mathbf{y} + \mathbf{V}(\mathbf{I} - \mathbf{\Sigma}_1)\mathbf{V}^\top \mathbf{x}_{0|1} \\ &= \mathbf{V}\mathbf{\Sigma}^\dagger \mathbf{U}^\top \mathbf{y} + (\mathbf{I} - \mathbf{V}\mathbf{\Sigma}_1 \mathbf{V}^\top)\mathbf{x}_{0|1} \\ &= \mathbf{A}^\dagger \mathbf{y} + (\mathbf{I} - \mathbf{A}^\dagger \mathbf{A})\mathbf{x}_{0|1} \end{aligned} \tag{40}$$

Now we can see that $\mathbf{V}\mathbf{\Sigma}_1 \bar{\mathbf{y}} = \mathbf{A}^\dagger \mathbf{y}$ is the range-space (*i.e.*, $\mathbf{A}^\dagger \mathbf{A}\mathbf{x}$) part while $\mathbf{V}(\mathbf{I} - \mathbf{\Sigma}_1)\bar{\mathbf{x}}_{0|1} = (\mathbf{I} - \mathbf{A}^\dagger \mathbf{A})\mathbf{x}_{0|1}$ is the null-space (*i.e.*, $(\mathbf{I} - \mathbf{A}^\dagger \mathbf{A})\mathbf{x}$) part.

CDDB [5]. In CDDB[1], they use the following guidance to ensure data consistency [5]

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \simeq \mathbf{A}^\top(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}). \tag{41}$$

However, one may note a predefined term ρ before $\mathbf{A}^\top(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t})$. In their official implementation, this is considered as a constant (the step size, default is set to 1.0).
DPS [4] and ΠGDM [21]. DPS proposes the following Jensen approximation[2]

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) \simeq \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_{0|t}) = \frac{\partial \mathbf{x}_{0|t}}{\partial \mathbf{x}_t} \frac{\partial \|\mathbf{A}\mathbf{x}_{0|t} - \mathbf{y}\|_2^2}{\partial \mathbf{x}_{0|t}} = \underbrace{\frac{\partial \mathbf{x}_{0|t}}{\partial \mathbf{x}_t}}_{\text{Jacobian}} \underbrace{\mathbf{A}^\top(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t})}_{\text{vector}} \tag{42}$$

of which the chain rule is based on the denominator layout notation [28], $\nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) \simeq \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_{0|t})$ is based on the Theorem 1 in [4]. Here, we can see that the gradient term can be represented as the Jacobian (J) vector (V) product (JVP). In the original implementation of DPS, the two terms are not computed separately, but computed directly as $\nabla_{\mathbf{x}_t} \|\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}\|_2^2$, where the whole term can be handled with backpropagation.

On the other hand, ΠGDM proposes

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) \simeq \mathcal{N}(\mathbf{A}\mathbf{x}_{0|t}, \mathbf{A}\mathbf{A}^\top + \mathbf{I}) = \underbrace{\frac{\partial \mathbf{x}_{0|t}}{\partial \mathbf{x}_t}}_{\text{Jacobian}} \underbrace{\mathbf{A}^\dagger(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t})}_{\text{vector}}, \tag{43}$$

where $\mathbf{A}^\dagger := \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}$ is the Moore-Penrose pseudo-inverse. Using the JVP for implementation, it is no longer required that the whole term is differentiable.

We know that $\mathbf{x}_{0|t}$ comes from

$$\mathbf{x}_{0|t} = \frac{1}{\sqrt{\bar\alpha_t}}(\mathbf{x}_t - \epsilon_\theta(\mathbf{x}_t, t)). \tag{44}$$

Hence, we have

$$\frac{\partial \mathbf{x}_{0|t}}{\partial \mathbf{x}_t} = \frac{\partial(\frac{1}{\sqrt{\bar\alpha_t}}(\mathbf{x}_t - \epsilon_\theta(\mathbf{x}_t, t)))}{\partial \mathbf{x}_t} \tag{45}$$

$$= \frac{1}{\sqrt{\bar\alpha_t}} \frac{\partial((\mathbf{x}_t - \epsilon_\theta(\mathbf{x}_t, t)))}{\partial \mathbf{x}_t} \tag{46}$$

Thus, to balance the $\frac{1}{\sqrt{\bar\alpha_t}}$ term brought by Jacobian, we can multiply the $\sqrt{\bar\alpha_t}$ to the left side of (42) and (43), *i.e.*,

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \simeq \sqrt{\bar\alpha_t} \underbrace{\frac{\partial \mathbf{x}_{0|t}}{\partial \mathbf{x}_t}}_{\text{Jacobian}} \underbrace{\mathbf{A}^\top(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t})}_{\text{vector}}, \tag{47}$$

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \simeq \sqrt{\bar\alpha_t} \underbrace{\frac{\partial \mathbf{x}_{0|t}}{\partial \mathbf{x}_t}}_{\text{Jacobian}} \underbrace{\mathbf{A}^\dagger(\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t})}_{\text{vector}}. \tag{48}$$

In ΠGDM, they use (48).

However, it should be noted that DPS uses a predefined term instead of $\sqrt{\bar\alpha_t}$, the term is

$$-\zeta_i(\frac{1 - \bar\alpha_t}{\sqrt{\bar\alpha_{t-1}}\beta_t}). \tag{49}$$

[1]Please note that they use I^{2}SB [18] as pre-trained models.

[2]We ignore scaling constants that are related to the measurement noise for simplicity. For implementation, this can be absorbed into the choice of step sizes.

where ζ_i is the step size in [4], *i.e.*,

$$\zeta_i = 1/\|\mathbf{y} - \mathbf{A}(\mathbf{x}_{0|t})\|. \tag{50}$$

They use different step sizes for different tasks, please refer to DPS [4] for details.
DDPG [9] DDPG considers combination of two guidance methods, *i.e.*,

$$\begin{aligned}\tilde{\mathbf{x}}_{0|t} &= \mathbf{x}_{0|t} - \mu_t \left((1-\delta_t)\mathbf{g}_{BP}(\mathbf{x}_{0|t}) + \delta_t \mathbf{g}_{LS}(\mathbf{x}_{0|t})\right), \\ &=: \mathbf{x}_{0|t} - \mu_t\, \mathbf{g}_{\delta_t}(\mathbf{x}_{0|t})\end{aligned} \tag{51}$$

where

$$\mathbf{g}_{BP}(\mathbf{x}_{0|t}) = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \eta\mathbf{I})^{-1}(\mathbf{A}\mathbf{x}_{0|t} - \mathbf{y}), \tag{52}$$

$$\mathbf{g}_{LS}(\mathbf{x}_{0|t}) = c\mathbf{A}^T(\mathbf{A}\mathbf{x}_{0|t} - \mathbf{y}), \tag{53}$$

where c is a constant (DDPG sets $c = 1$), $\eta > 0$ is the regularization hyperparameter. They replace $\mathbf{A}^\dagger$ with a regularized version $\mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \eta\mathbf{I}_m)^{-1}$ [24, 25]. If we set $\eta = 0$, we get

$$\mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \eta\mathbf{I}_m)^{-1} \overset{\eta=0}{=} \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}, \tag{54}$$

which is the Moore-Penrose pseudo-inverse $\mathbf{A}^\dagger := \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}$ used in ΠGDM.
DiffPIR [30] They use a first-order proximal operator method as follows:

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \simeq \nabla_{\mathbf{x}_{0|t}} \|\mathbf{y} - \mathbf{A}\mathbf{x}_{0|t}\|^2. \tag{55}$$

We ignore scaling constants that are related to the measurement noise for simplicity. For implementation, this can be absorbed into the choice of step sizes. This approximation can be considered as one step of gradient descent, and we can calculate it numerically.

A.4. Section 5 Equation (14)

Using this property, we can get the equivariant inverse mapping, i.e.,

$$\mathbf{x}^*_{0|t} := T_f^{-1}\mathcal{D}_\theta(T_f\mathbf{x}_t), \tag{56}$$

here T_f^{-1} represents the inverse transformation of T_f. $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$, $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$, and $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})$ can also be computed.
Explanation $\mathbf{x}^*_{0|t}$ can be computed as

$$\begin{aligned}\mathbf{x}^*_{0|t} := T_f^{-1}\mathcal{D}_\theta(T_f\mathbf{x}_t) &= T_f^{-1}\left(\frac{1}{\sqrt{\bar{\alpha}_t}}(T_f\mathbf{x}_t + (1-\bar{\alpha}_t)\nabla_{T_f\mathbf{x}_t}\log p_t(T_f\mathbf{x}_t)\right) && (57)\\ &\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(T_f^{-1}T_f\mathbf{x}_t + (1-\bar{\alpha}_t)T_f^{-1}\mathbf{s}_\theta(T_f\mathbf{x}_t, t)) = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + (1-\bar{\alpha}_t)T_f^{-1}\mathbf{s}_\theta(T_f\mathbf{x}_t, t)) && (58)\\ &\simeq \frac{1}{\sqrt{\bar{\alpha}_t}}(T_f^{-1}T_f\mathbf{x}_t + \sqrt{(1-\bar{\alpha}_t)}T_f^{-1}\epsilon_\theta(T_f\mathbf{x}_t, t)) = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t + \sqrt{(1-\bar{\alpha}_t)}T_f^{-1}\epsilon_\theta(T_f\mathbf{x}_t, t)). && (59)\end{aligned}$$

where $T_f^{-1}T_f = \mathbf{I}$. $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ can be computed as

$$\begin{aligned}\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) &:= T_f^{-1}\nabla_{T_f\mathbf{x}_t}\log p_t(T_f\mathbf{x}_t) && (60)\\ &\simeq T_f^{-1}\mathbf{s}_\theta(T_f\mathbf{x}_t, t) && (61)\\ &\simeq -\frac{T_f^{-1}\epsilon_\theta(T_f\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}. && (62)\end{aligned}$$

$\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ is computed based on $\mathbf{x}^*_{0|t}$. $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})$ is the composition of $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ and $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$.

B. Details about Diffusion Models

In Denoising Diffusion Probabilistic Models (DDPM) [11], the T-step forward process, denoted as $q(\mathbf{x}_t|\mathbf{x}_{t-1})$, is employed to perturb the data by incrementally adding Gaussian noise according to a predefined noise schedule $\beta_{1:T}$:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}, \quad \epsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{63}$$

where $\mathbf{x}_t$ represents the noised image at time-step t, and $\mathcal{N}$ indicates the Gaussian distribution. Utilizing the advantageous properties of the Gaussian distribution, we could perform parameter substitutions and employ the reparameterization trick to obtain $q(\mathbf{x}_t|\mathbf{x}_0)$:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \alpha_t = 1-\beta_t, \bar{\alpha}_t = \prod_{i=0}^{t}\alpha_i \tag{64}$$

The T-step reverse process aims to invert the forward process using the posterior distribution $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$. This posterior can be derived from Bayes' theorem utilizing (63) and (64):

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sqrt{\beta_t}\boldsymbol{\epsilon}_t, \tag{65}$$

where $\epsilon_\theta(\mathbf{x}, t)$ is a function approximator designed to predict the total noise between $\mathbf{x}_t$ and $\mathbf{x}_0$ as expressed in (64), $\boldsymbol{\epsilon}_t$ is drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

To train ϵ_θ, DDPM randomly selects a clean image $\mathbf{x}_0$ and samples noise $\boldsymbol{\epsilon}$. A random time-step t is then chosen, and the network parameters θ in ϵ_θ are updated using the following gradient descent step [11]:

$$\nabla_\theta \left\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \boldsymbol{\epsilon}\sqrt{1-\bar{\alpha}_t}, t\right)\right\|_2^2. \tag{66}$$

By iteratively calculating $\mathbf{x}_{t-1}$ using (65), DDPM can yield clean images $\mathbf{x}_0$ from random noises $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. DDIM [20] replace the step in (65) with:

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_{0|t} + \hat{\sigma}_t\epsilon_\theta(\mathbf{x}_t; t) + \tilde{\sigma}_t\epsilon_t, \tag{67}$$

where

$$\hat{\sigma}_t = \sqrt{1-\bar{\alpha}_{t-1}-\tilde{\sigma}_t^2}, \tag{68}$$

$$\mathbf{x}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}_t; t)\right). \tag{69}$$

The hyperparameter $\tilde{\sigma}_t$ regulates the stochasticity of the whole reverse process. $\mathbf{x}_{0|t}$ are obtained from Gaussian denoising as implied in (64).

The entire process in DDPM can also be described using stochastic differential equations (SDE) [23], with the forward process defined as:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}_t, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}, \tag{70}$$

where $\mathbf{w}$ is the standard Wiener process, $\mathbf{f}(\cdot, t)$ is a vector-valued function called the drift coefficient, and $g(\cdot)$ is a scalar function known as the diffusion coefficient.

This process can be reversed by solving the reverse-time SDE/ordinary differential equation (ODE) [2, 15, 23] governed by the score function:

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}_t, t) - g^2(t)\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}, \tag{71}$$

where $p_t(\mathbf{x}_t)$ is the marginal probability density at timestep t, and the only unknown part $\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)$ can be modelled as so-called score function $\mathbf{s}_\theta(\mathbf{x}_t, t)$ with score matching methods [13, 22].

When $\mathbf{f}(\mathbf{x}_t, t) = -0.5\beta(t)\mathbf{x}_t$ and $g(t) = \sqrt{\beta(t)}$ (VP-SDE in [23]), score-based diffusion models transforms into the continuous-time version of DDPM where:

$$\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) \simeq \mathbf{s}_\theta(\mathbf{x}_t, t) \simeq -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}. \tag{72}$$

In other words, they can also utilize DDPM models for the reverse process. Both $\mathbf{s}_\theta(\mathbf{x}_t, t)$ and $\epsilon_\theta(\mathbf{x}_t, t)$ can represent diffusion models.

Algorithm 1 Original Sampling

```
Require: $\mathbf{y}$, $\epsilon_\theta$, $T$, $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  for $t = T, ..., 1$ do
    //Calculate the predicted $\mathbf{x}_{0|t}$
    $\mathbf{x}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - (1 - \bar{\alpha}_t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t))$
    $\tilde{\mathbf{x}}_{0|t} = \mathbf{x}_{0|t} + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ //Update $\mathbf{x}_{0|t}$ based on the guidance
    $\mathbf{x}_{t-1} \sim p(\mathbf{x}_{t-1}|\mathbf{x}_t, \tilde{\mathbf{x}}_{0|t})$ //One step reverse process
  end for
  return $\mathbf{x}_0$
```

Algorithm 2 Equivariant Sampling (EquS)

```
Require: $\mathbf{y}$, $\epsilon_\theta$, $T$, $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\text{seq} = [0, 1] * (T/2)$
  for $t = T, ..., 1$ do
    if $\text{seq}(t) = 1$:
      Calculate $\tilde{\mathbf{x}}_{0|t}$ based on $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ and $\nabla^*_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$
    else:
      Calculate $\tilde{\mathbf{x}}_{0|t}$ based on $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ and $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$
    $\mathbf{x}_{t-1} \sim p(\mathbf{x}_{t-1}|\mathbf{x}_t, \tilde{\mathbf{x}}_{0|t})$
  end for
  return $\mathbf{x}_0$
```

C. Experiment Details

Degradation operators. Specifically, we consider block-based CS with a 0.25 compression ratio, text mask for inpainting, 4× bicubic downsampling for SR, Gaussian blur for deblurring, and use an average grayscale operator in colorization.

Comparison methods. For noise-free tasks, we compare against include DPS [4], DDRM [16], DDNM [26], ΠGDM [21], CDDB [3] and DeqIR [3]. EquS$^+$ is integrated into DDRM, CDDB, and DDNM (see Algorithm 1 and Algorithm 2 for details). For noisy tasks, comparison methods include DDNM$^+$ [26] and DDPG [9], where EquS is integrated into both DDNM$^+$ and DDPG. We use EquS for noisy IR tasks and colorization. ZAPS [1] was not included as a comparison method because its code was not available at the time of our submission.

Datasets and pre-trained models. For validation, we use ImageNet 1K [6] and CelebA 1K [14] datasets, with images resized to 256×256. To ensure fair comparisons, we apply the same pretrained diffusion models across all methods: [7] for ImageNet and [19] for CelebA-HQ.

Evaluation metrics. We employ PSNR, SSIM [27], and LPIPS [29] as the main evaluation metrics for most IR tasks. For colorization specifically, we utilize the Consistency metric (referred to as *Cons*) [26] along with FID [10], since PSNR and SSIM do not adequately capture performance in this context. Generally, improved performance is indicated by higher PSNR and SSIM values, together with lower values for LPIPS, FID, and *Cons*.

DDNM [26]. For DDNM and DDNM$^+$ we use their official repository. They used a DDIM [20] sampler with 100 NFEs. DDNM$^+$ did nott implement the time-travel trick in [26]. DDNM+EquS$^+$ and DDNM$^+$+EquS use the same sampler used in DDNM and DDNM$^+$.

CDDB [3]. For CDDB, we use the guidance method used in their official repository and use the DDIM sampler with 100 NFE. CDDB+EquS$^+$ uses the same sampler used in DDNM and DDNM$^+$.

DDRM [16]. For DDRM, we use their official repository and use DDIM sampler with 100 NFEs. DDRM + EquiS$^+$ uses the same sampler as used in DDRM.

ΠGDM [21]. For ΠGDM, we use the pseudocode in their paper[3] and use DDIM [20] sampler with 100 NFEs (detailed in their paper).

DeqIR [3]. We use the official repository for implementation. For super-resolution, deblurring, and colorization tasks, the number of iterations is set to 15, as reported in their paper [3]. For inpainting and compressed sensing tasks, the iteration count is set to 25 (25 is reported for inpainting, while the iteration for compressed sensing is not specified in their paper and

[3]We didn't find their official repository.

CelebA Inpainting	DDNM [26] PSNR↑/SSIM↑/LPIPS↓	DDNM-**EquS**$^+$ PSNR↑/SSIM↑/LPIPS↓
Square	35.29 / 0.977 / 0.018	**36.33 / 0.981 / 0.017**
Circle	33.61 / 0.971 / 0.022	**34.61 / 0.975 / 0.021**

Table 2. Quantitative results on ImageNet. **bold**: improved scores. Our method is robust in these situations.

supplementary).

DPS [4]. We use the guidance method used in DPS and use the DDIM sampler with 100 NFEs.

DDPG [9]. We use the default hyperparameter settings in their official repository, which also apply to DDPG+EquS.

Why EquS and EquS$^+$ do not introduce additional computational overhead? This is because TAS employs a quadratic sampling schedule without altering the length of the subsequence S, ensuring the same number of function evaluations (NFEs) as previous methods. Additionally, EquS and EquS$^+$ only incorporate simple transformations, which can be efficiently implemented using versatile libraries such as torchvision.

D. More Quantitative Results

Degradations operators. Specifically, we consider Walsh-Hadamard CS with a 0.25 compression ratio, a different text mask for inpainting, 4× Average Pooling for SR, anisotropic blur for deblurring.

Datasets and pre-trained models. For validation, we use the ImageNet 1K [6] and CelebA 1K [14] datasets, with images resized to 256×256. To ensure fair comparisons, we apply the same pre-trained diffusion models across all methods: [7] for ImageNet and [19] for CelebA-HQ.

Comparison methods. We compare against include DPS [4], DDRM [16], DDNM [26], ΠGDM [21], CDDB [3] and DeqIR [3]. EquS$^+$ is integrated into DDRM, CDDB, and DDNM (see Algorithm 1 and Algorithm 2 for details). ZAPS [1] was not included as a comparison method because its code was unavailable at the time of our submission.

Evaluation metrics. We employ PSNR, SSIM [27], and LPIPS [29] as the main evaluation metrics for most IR tasks. For colorization specifically, we utilize the Consistency metric (referred to as *Cons*) [26] along with FID [10], since PSNR and SSIM do not adequately capture performance in this context. Generally, improved performance is indicated by higher PSNR and SSIM values, alongside lower values for LPIPS, FID, and *Cons*.

Quantitative results. We show quantitative results in Table 3. **Walsh-Hadamard CS.** CDDB-EquS$^+$ outperforms most other methods on benchmarks. Specifically, compared to the competitive IR method ΠGDM, CDDB-EquS$^+$ achieves an LPIPS improvement of up to 0.080 and a PSNR gain of up to 1.64 dB on ImageNet. **Anisotropic Deblurring.** Notably, DDNM-EquS$^+$ outperforms most other methods on benchmarks. **Inpainting.** CDDB-EquS$^+$ outperforms most other methods on benchmarks. For instance, compared to the competitive IR method CDDB, CDDB-EquS$^+$ achieves a PSNR gain of up to 0.96 dB on CelebA. **Average Pooling SR.** DDNM-EquS$^+$ outperforms most other methods on CelebA. Although ΠGDM achieves superior performance on the ImageNet, its reliance on gradient computation introduces additional computational overhead.

E. More Ablation Studies

We show quantitative results of ablation studies in Table 4. Note that FID requires a sufficiently large dataset to accurately estimate the statistics of the data distribution (typically 50k images), whereas the benchmark (for consistency with prior works) uses only 1k samples, leading to a high variance of FID.

ImageNet	Anisotropic Deblurring	Walsh-Hadamard CS 25%	Inpainting	$4\times$ Average Pooling SR
Method	PSNR↑/SSIM↑/LPIPS↓	PSNR↑/SSIM↑/LPIPS↓	PSNR↑/SSIM↑/LPIPS↓	PSNR↑/SSIM↑/LPIPS↓
$\mathbf{A}^\dagger\mathbf{y}$	21.67 / 0.560 / 0.428	15.66 / 0.360 / 0.585	14.53 / 0.778 / 0.250	24.21 / 0.711 / 0.401
DPS [4]	25.48 / 0.709 / 0.351	22.54 / 0.713 / 0.263	32.53 / 0.944 / 0.068	22.38 / 0.580 / 0.470
ΠGDM [21]	40.03 / 0.973 / 0.043	22.55 / 0.712 / 0.263	32.53 / 0.944 / 0.067	27.48 / 0.808 / 0.223
DeqIR [3]	39.66 / 0.973 / 0.030	14.17 / 0.341 / 0.554	26.42 / 0.905 / 0.080	26.18 / 0.772 / 0.242
CDDB [5]	25.57 / 0.717 / 0.331	21.71 / 0.628 / 0.335	32.07 / 0.948 / 0.039	23.42 / 0.634 / 0.410
CDDB-**EquS**$^+$	26.50 / 0.759 / 0.301	24.18 / 0.780 / 0.183	33.23 / 0.958 / 0.045	25.49 / 0.722 / 0.339
DDRM [16]	40.01 / 0.976 / 0.025	19.88 / 0.573 / 0.391	31.68 / 0.944 / 0.047	27.08 / 0.794 / 0.231
DDRM-**EquS**$^+$	41.91 / 0.983 / 0.016	23.50 / 0.771 / 0.189	32.89 / 0.954 / 0.052	27.30 / 0.803 / 0.246
DDNM [26]	40.81 / 0.978 / 0.022	21.70 / 0.629 / 0.335	32.09 / 0.948 / 0.231	27.06 / 0.792 / 0.220
DDNM-**EquS**$^+$	42.98 / 0.985 / 0.013	23.72 / 0.768 / 0.195	33.14 / 0.958 / 0.045	27.38 / 0.805 / 0.238
CelebA	Anisotropic Deblurring	Walsh-Hadamard CS 25%	Inpainting	$4\times$ Average Pooling SR
Method	PSNR↑/SSIM↑/LPIPS↓	PSNR↑/SSIM↑/LPIPS↓	PSNR↑/SSIM↑/LPIPS↓	PSNR↑/SSIM↑/LPIPS↓
$\mathbf{A}^\dagger\mathbf{y}$	23.13 / 0.708 / 0.341	15.10 / 0.360 / 0.610	15.58 / 0.783 / 0.271	27.23 / 0.809 / 0.399
DPS [4]	29.46 / 0.826 / 0.180	29.38 / 0.870 / 0.147	35.44 / 0.953 / 0.070	26.74 / 0.779 / 0.230
ΠGDM [21]	39.47 / 0.972 / 0.055	29.39 / 0.871 / 0.147	35.44 / 0.953 / 0.070	31.65 / 0.890 / 0.127
DeqIR [3]	43.32 / 0.985 / 0.024	16.12 / 0.520 / 0.395	31.36 / 0.950 / 0.042	31.44 / 0.889 / 0.125
CDDB [5]	29.52 / 0.829 / 0.171	27.61 / 0.832 / 0.180	35.67 / 0.965 / 0.029	27.10 / 0.790 / 0.218
CDDB-**EquS**$^+$	31.06 / 0.865 / 0.180	29.28 / 0.875 / 0.143	36.63 / 0.971 / 0.027	30.13 / 0.859 / 0.203
DDRM [16]	41.30 / 0.979 / 0.031	25.02 / 0.785 / 0.212	34.85 / 0.956 / 0.047	31.38 / 0.883 / 0.128
DDRM-**EquS**$^+$	41.29 / 0.979 / 0.030	28.23 / 0.867 / 0.144	35.83 / 0.962 / 0.045	32.16 / 0.900 / 0.137
DDNM [26]	43.19 / 0.985 / 0.024	27.54 / 0.832 / 0.180	35.63 / 0.965 / 0.029	31.39 / 0.884 / 0.119
DDNM-**EquS**$^+$	43.33 / 0.985 / 0.021	28.99 / 0.871 / 0.146	36.59 / 0.970 / 0.027	32.23 / 0.900 / 0.127

Table 3. Quantitative results of zero-shot DMIR methods on **ImageNet**(*top*) and **CelebA**(*bottom*), including four typical IR tasks.

CelebA-HQ	Deblur (Gaussian)				Deblur (Anisotropic)				Inpainting (Mask 1)				Inpainting (Mask 2)			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDNM	46.73	0.992	0.011	1.414	43.19	**0.985**	0.024	2.796	39.37	0.982	0.017	2.811	35.63	0.965	0.029	4.489
DDNM-**EquS**	46.85	0.992	0.011	1.391	**43.39**	**0.985**	0.023	2.700	**39.90**	**0.984**	**0.016**	**2.548**	36.37	0.969	0.028	4.505
DDNM-**EquS**$^+$	**46.88**	**0.993**	**0.010**	**1.238**	43.33	**0.985**	**0.021**	**2.482**	39.85	0.983	**0.016**	2.658	**36.59**	**0.970**	**0.027**	**4.334**
ImageNet	Deblur (Gaussian)				Deblur (Anisotropic)				Inpainting (Mask 1)				Inpainting (Mask 2)			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDNM	44.94	0.990	0.011	1.164	40.81	0.978	0.022	2.178	34.93	0.968	0.025	3.238	32.09	0.948	**0.039**	**3.845**
DDNM-**EquS**	45.37	0.991	0.009	0.954	41.52	0.982	0.018	1.710	35.65	0.973	0.023	2.969	32.83	0.955	0.045	4.624
DDNM-**EquS**$^+$	**46.99**	**0.993**	**0.006**	**0.547**	**42.74**	**0.985**	**0.013**	**1.073**	**36.13**	**0.976**	**0.021**	**2.595**	**33.14**	**0.958**	0.045	4.667
CelebA-HQ	Block-based CS 50%				Block-based CS 25%				Walshhadamard CS 50%				Walshhadamard CS 25%			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDNM	37.58	0.955	0.057	9.594	33.51	0.907	0.111	22.62	33.49	0.922	0.109	15.83	27.54	0.832	0.180	28.73
DDNM-**EquS**	37.87	0.958	0.056	**9.366**	34.01	0.916	0.109	**21.69**	34.38	0.931	0.102	13.95	28.72	0.857	0.165	25.02
DDNM-**EquS**$^+$	**38.09**	**0.960**	**0.051**	9.867	**34.76**	**0.926**	**0.091**	23.05	**34.99**	**0.939**	**0.089**	**11.76**	**28.99**	**0.871**	**0.146**	**22.78**
ImageNet	Block-based CS 50%				Block-based CS 25%				Walshhadamard CS 50%				Walshhadamard CS 25%			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDNM	33.86	0.918	0.088	10.61	28.59	0.795	0.225	30.12	29.27	0.858	0.150	18.44	21.70	0.629	0.335	64.47
DDNM-**EquS**	35.10	0.937	0.065	7.854	30.02	0.845	0.173	24.17	30.96	0.895	0.113	12.83	23.48	0.722	0.258	39.18
DDNM-**EquS**$^+$	**36.98**	**0.955**	**0.038**	**3.839**	**31.87**	**0.887**	**0.103**	**12.80**	**32.12**	**0.923**	**0.070**	**7.593**	**23.72**	**0.768**	**0.195**	**26.79**

Table 4. Quantitative results of ablation studies.

F. More Qualitative Results

F.1. More Results on Compressed Sensing

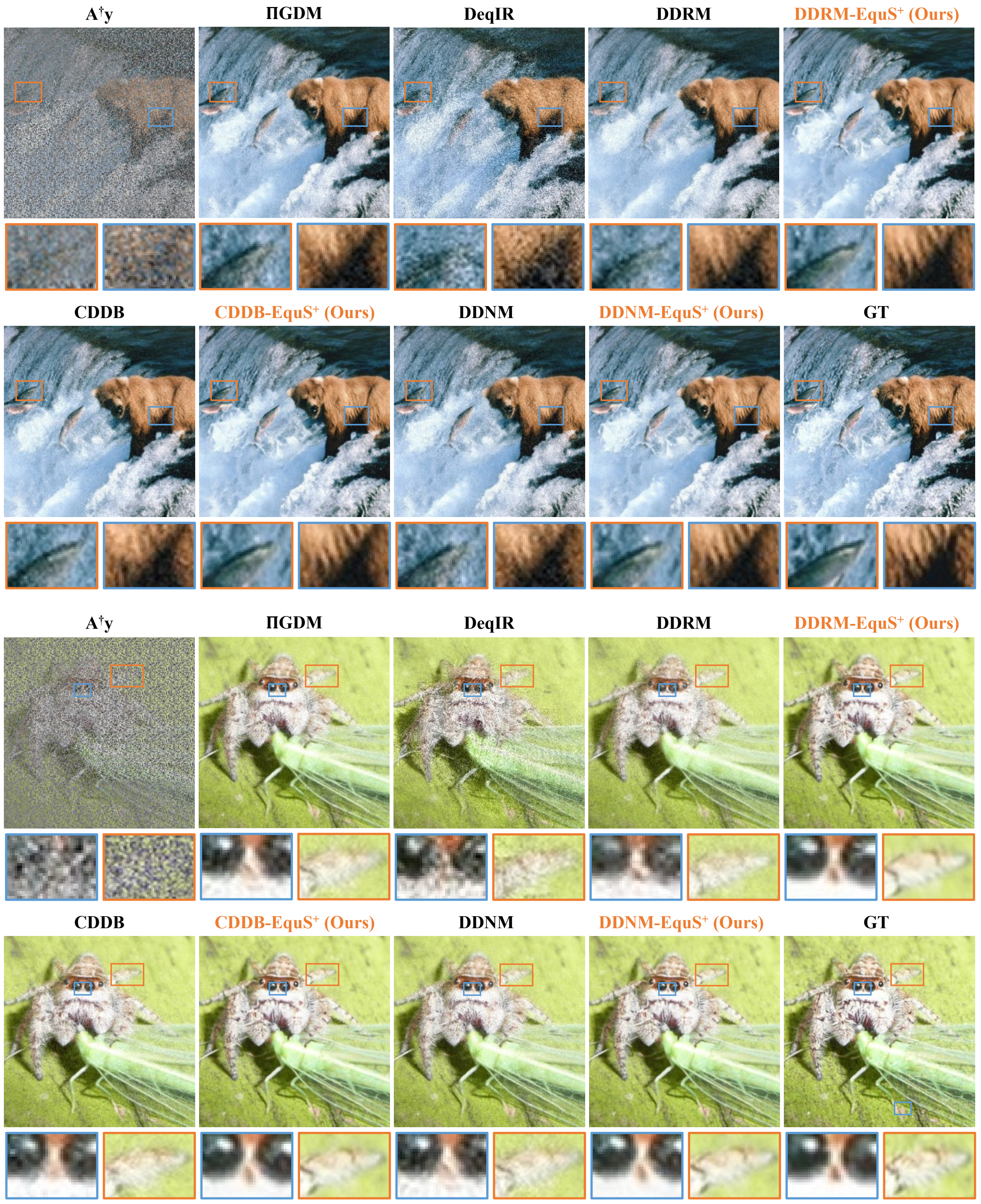

Figure 1. Qualitative results of image block-based compressed sensing (25%) on ImageNet.

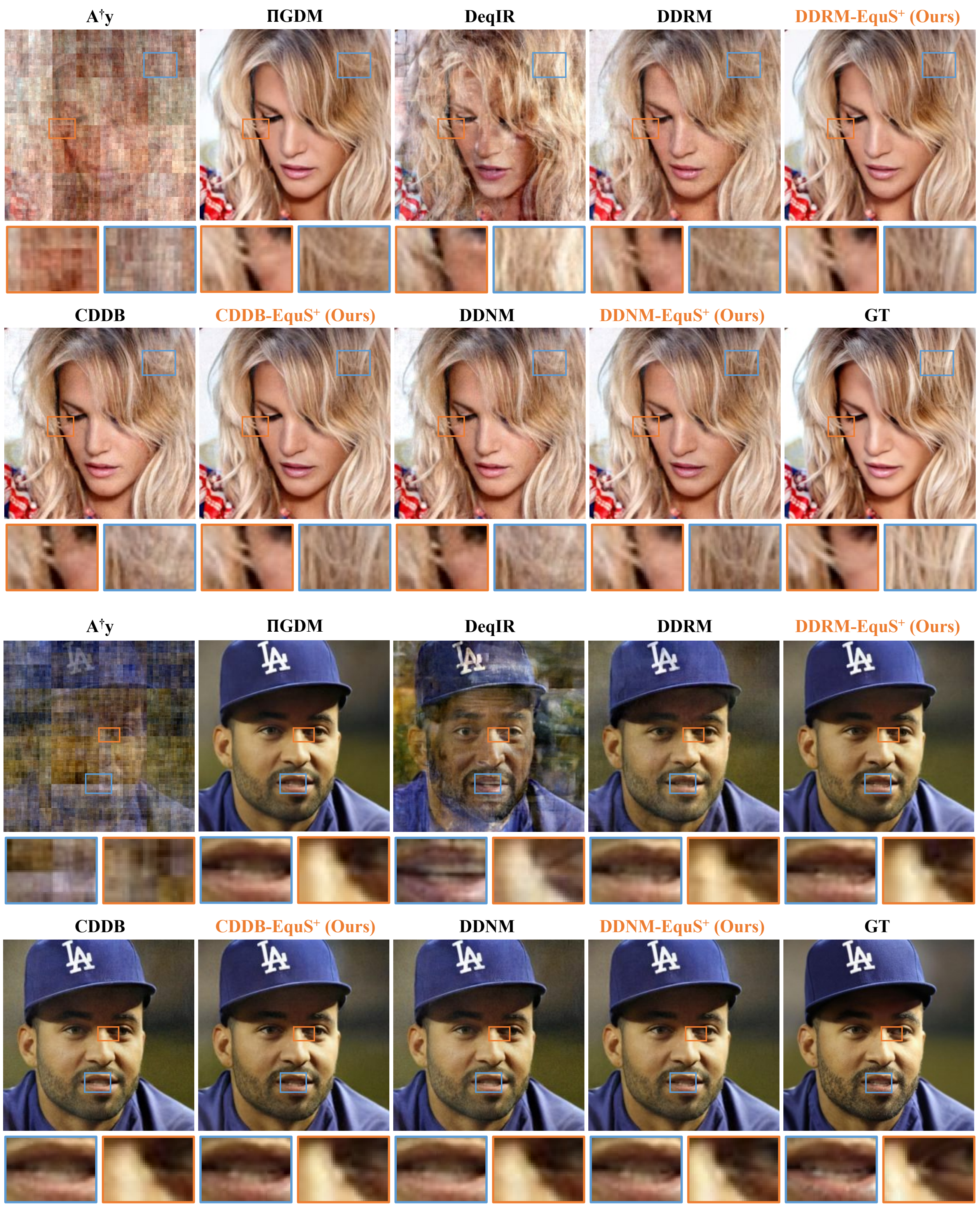

Figure 2. Qualitative results of image walshhadamard compressed sensing (25%) on CelebA.

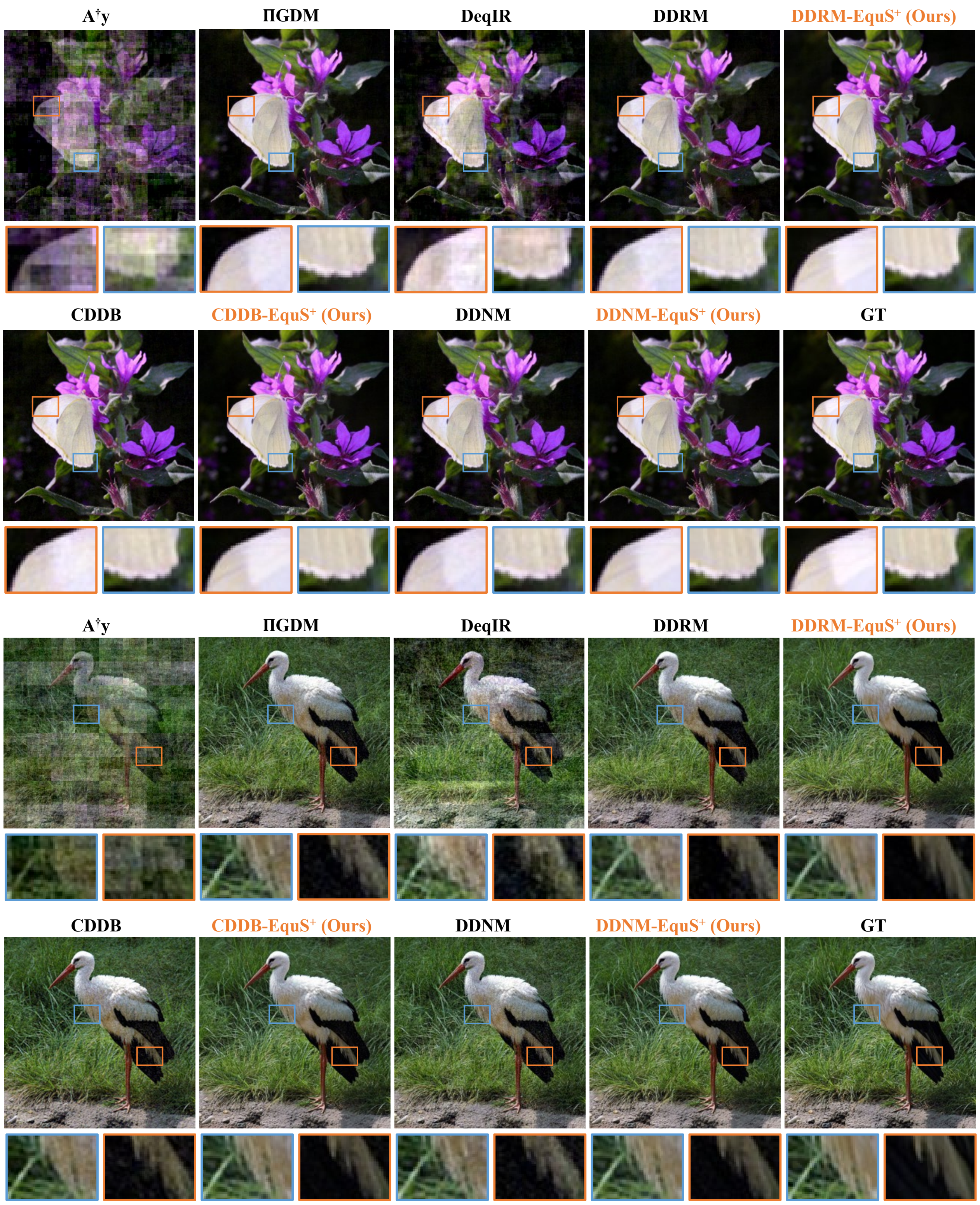

Figure 3. Qualitative results of image walshhadamard compressed sensing (50%) on ImageNet.

F.2. More Results on Super-Resolution

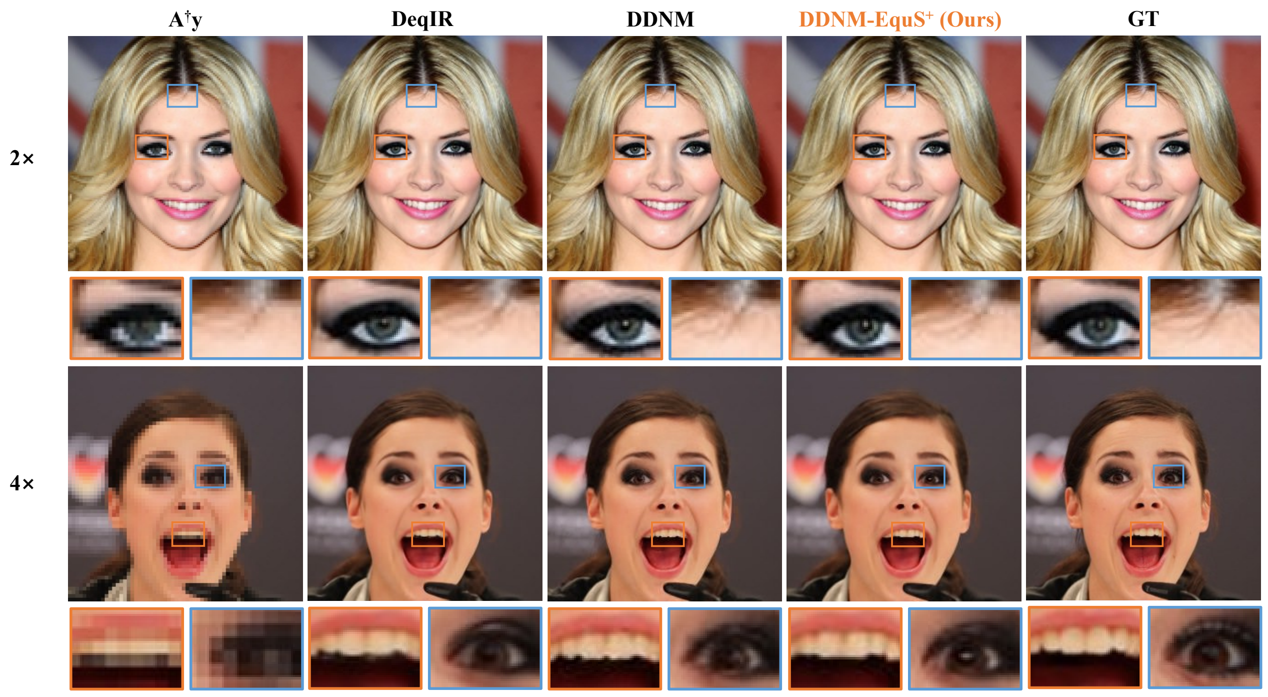

Figure 4. Qualitative results of image average pooling super-resolution on CelebA.

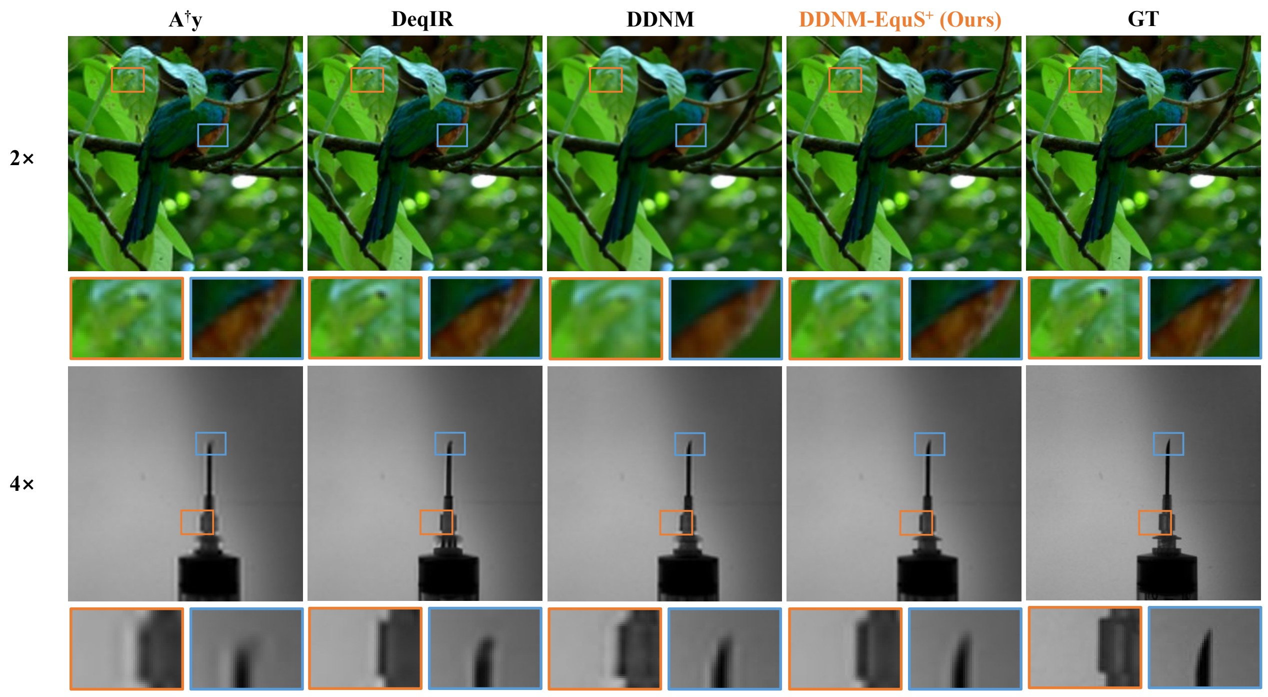

Figure 5. Qualitative results of image bicubic super-resolution on ImageNet.

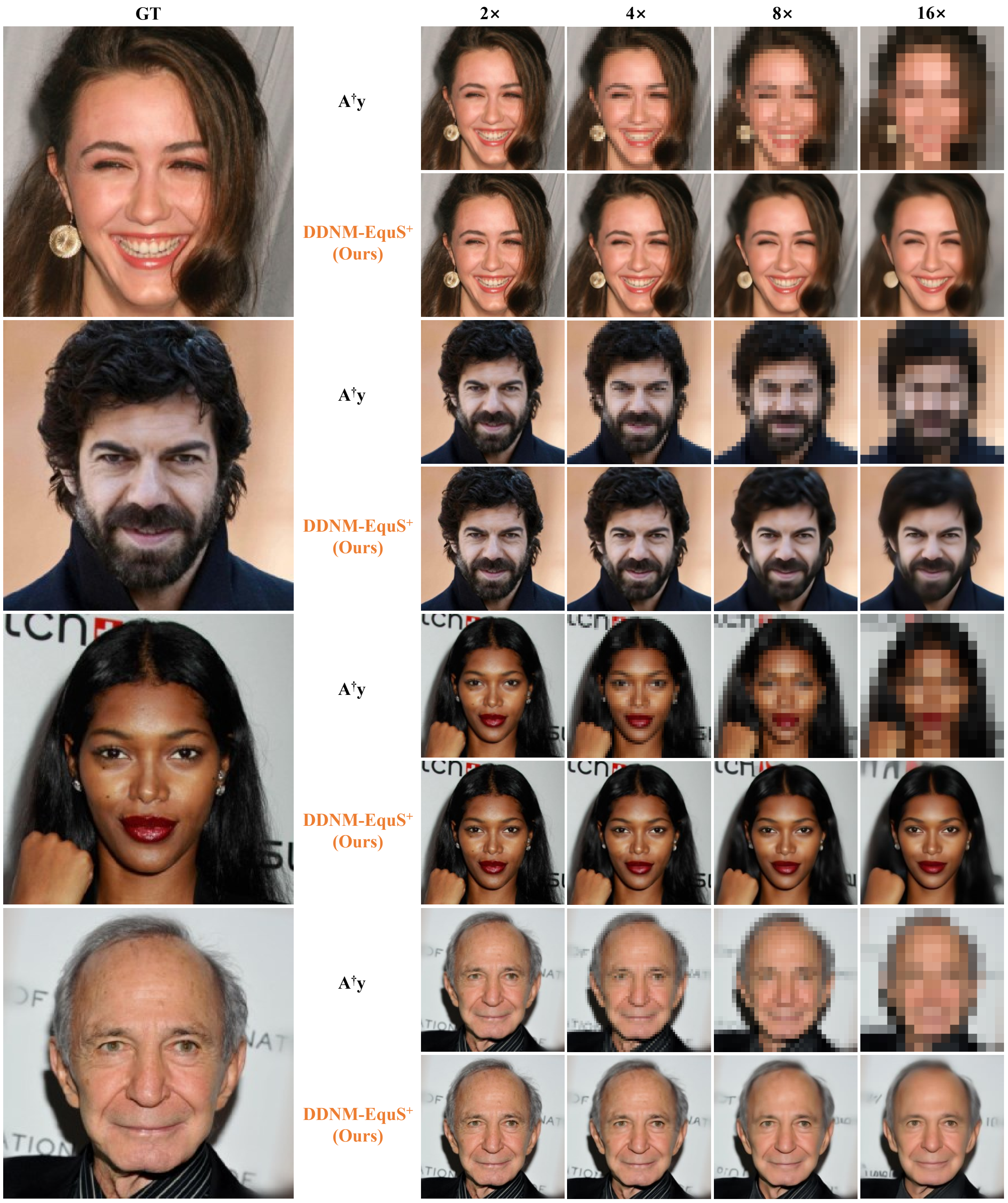

Figure 6. Qualitative results of image bicubic super-resolution for different scales on CelebA.

F.3. More Results on Deblurring

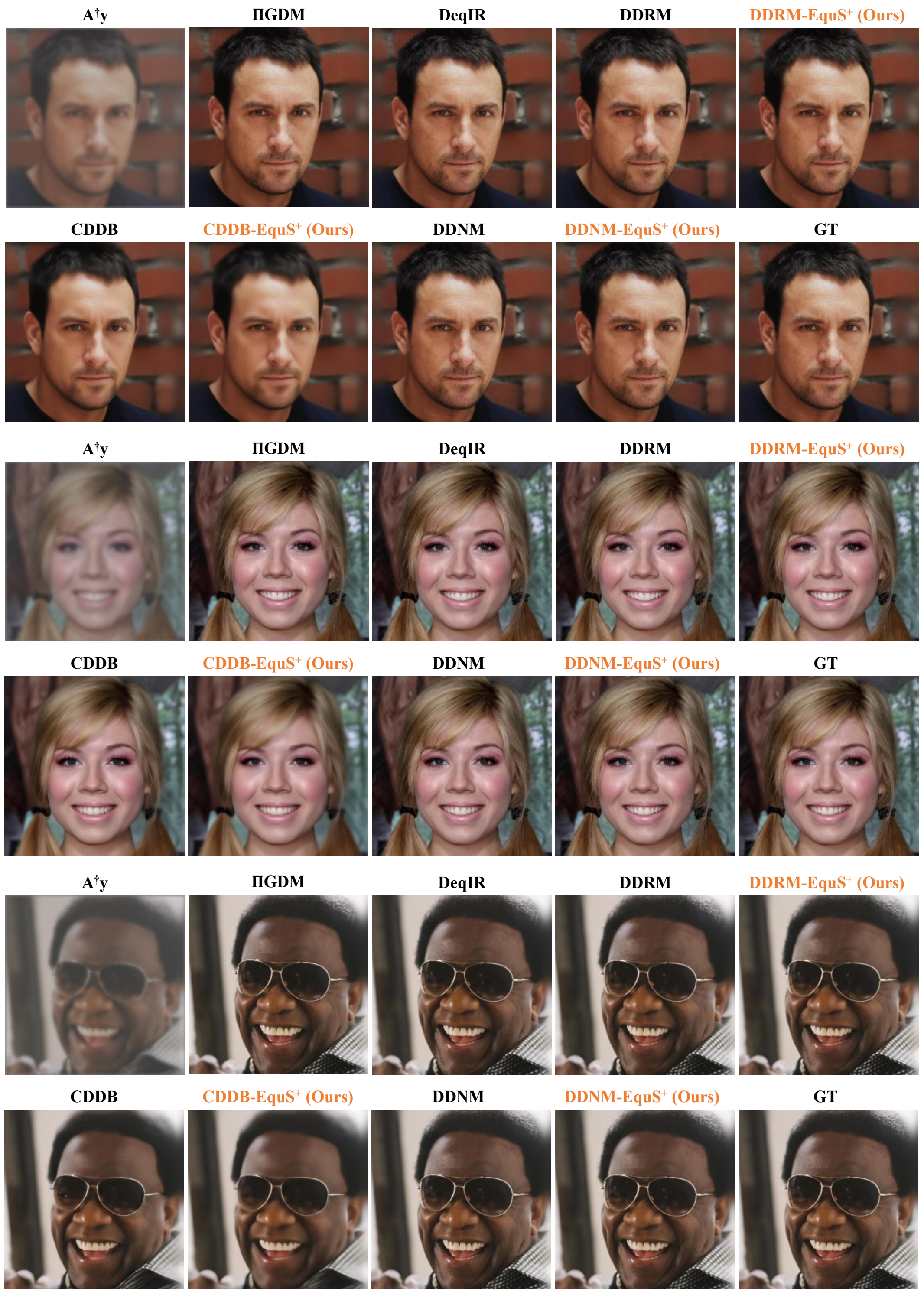

Figure 7. Qualitative results of image gaussian deblurring on CelebA.

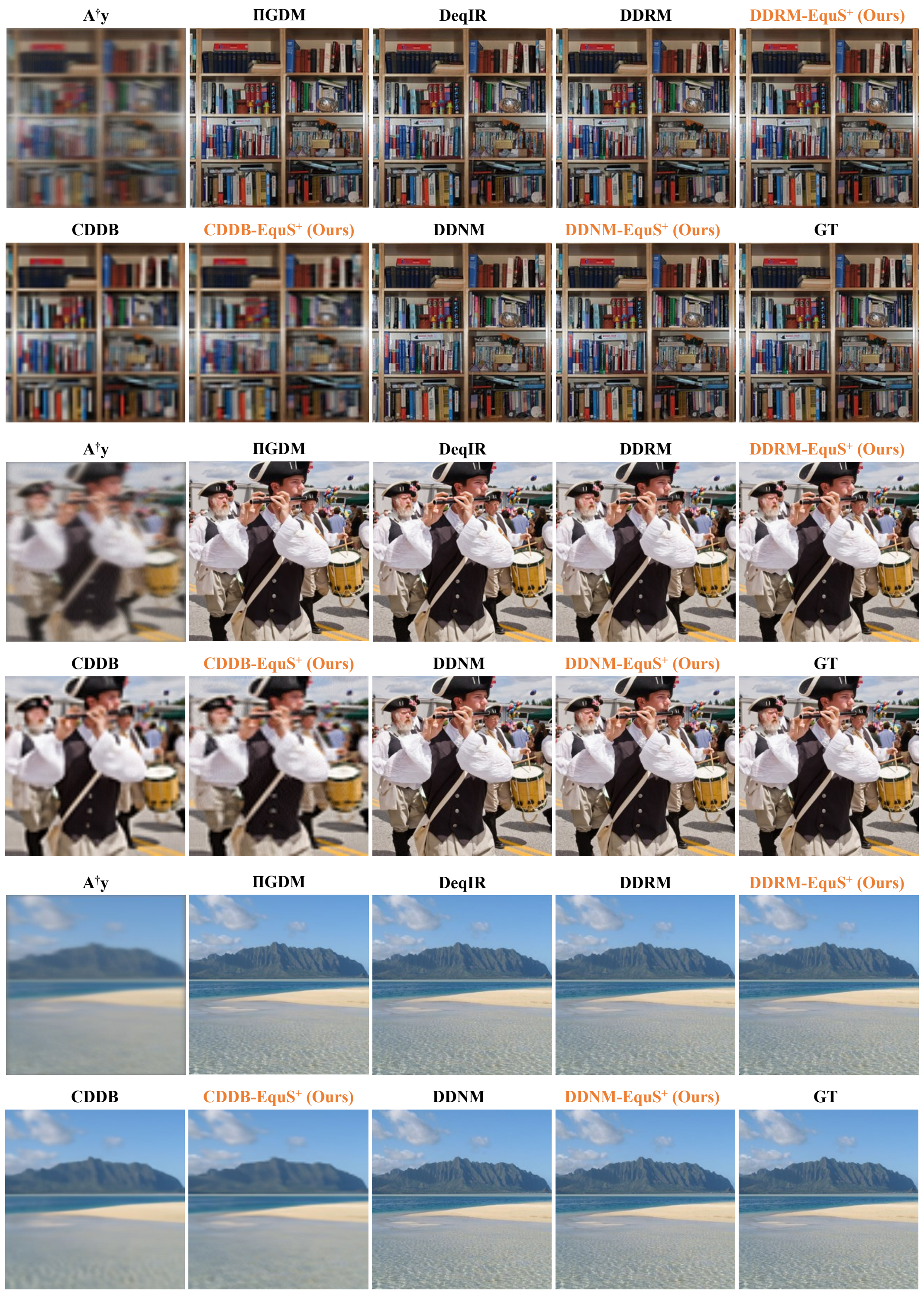

Figure 8. Qualitative results of image anisotropic deblurring on ImageNet.

F.4. More Results on Inpainting

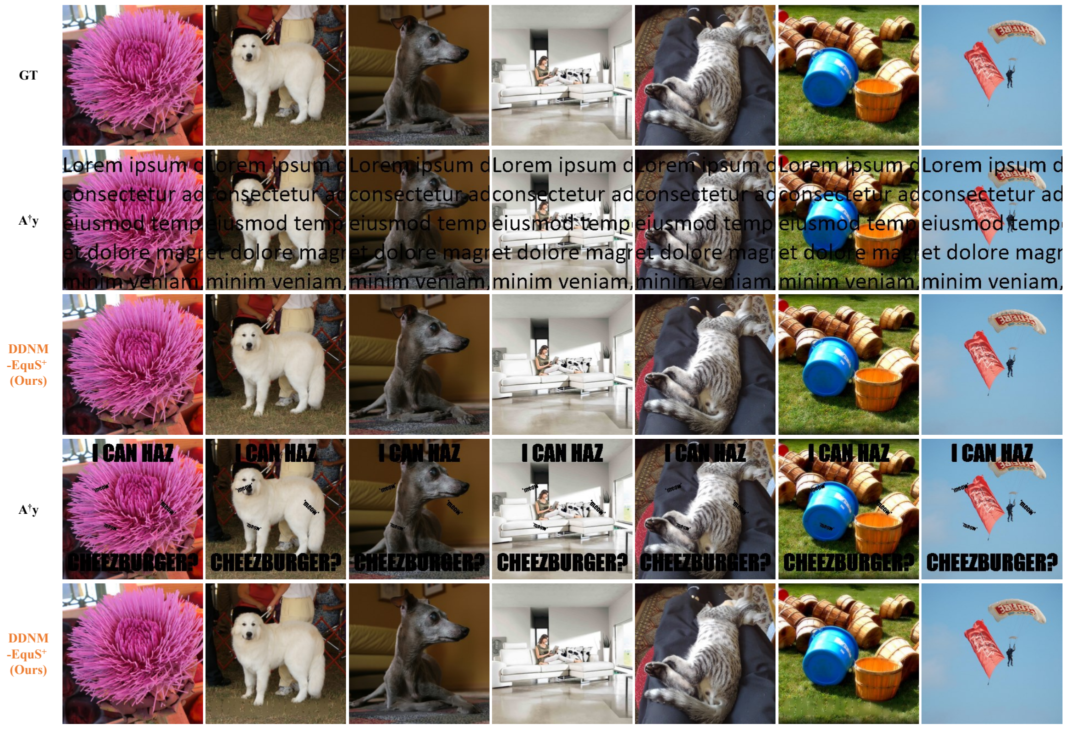

Figure 9. Qualitative results of image inpainting on ImageNet.

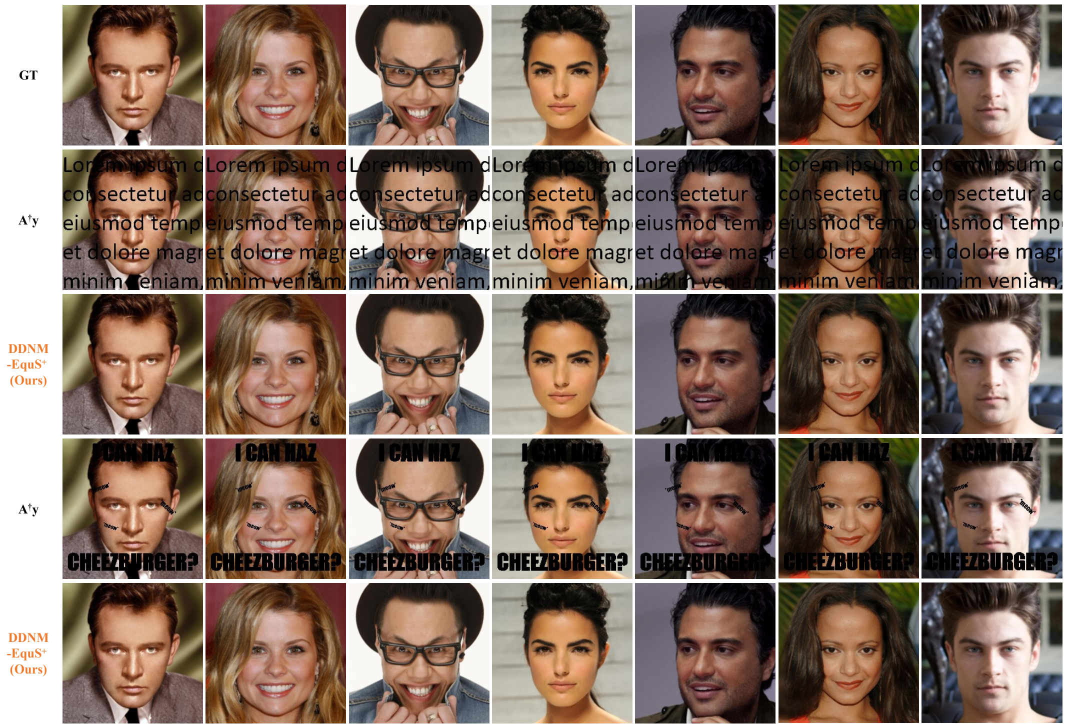

Figure 10. Qualitative results of image inpainting on CelebA.

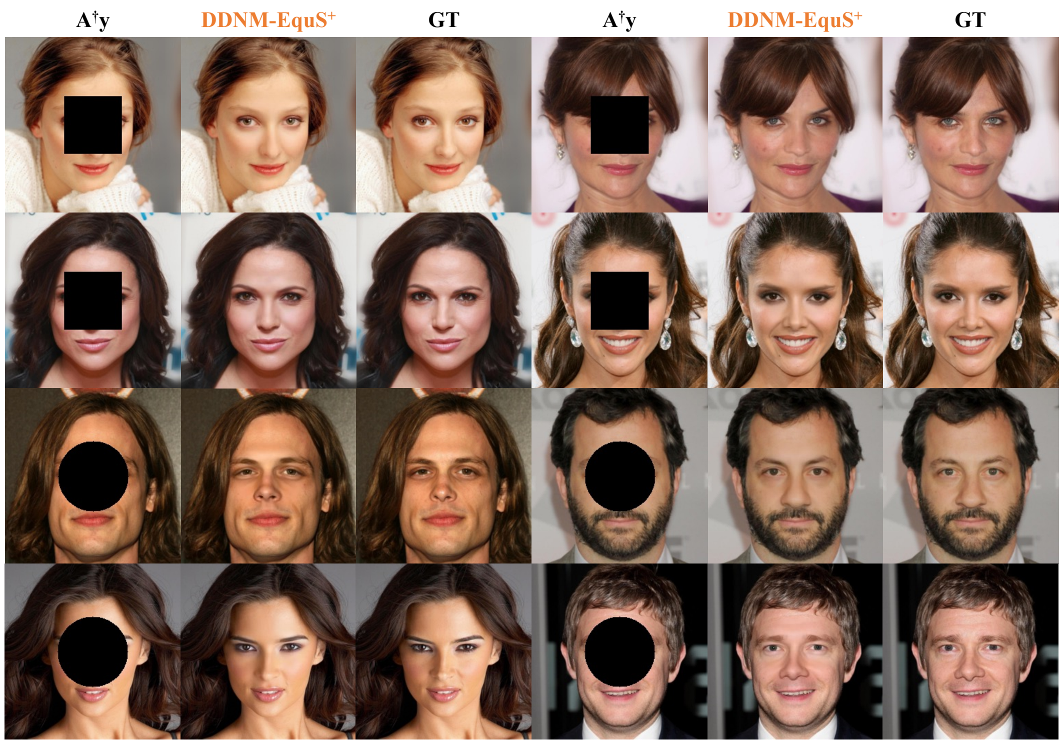

Figure 11. Qualitative results of image inpainting (black hole mask) on CelebA.

F.5. More Results on Colorization

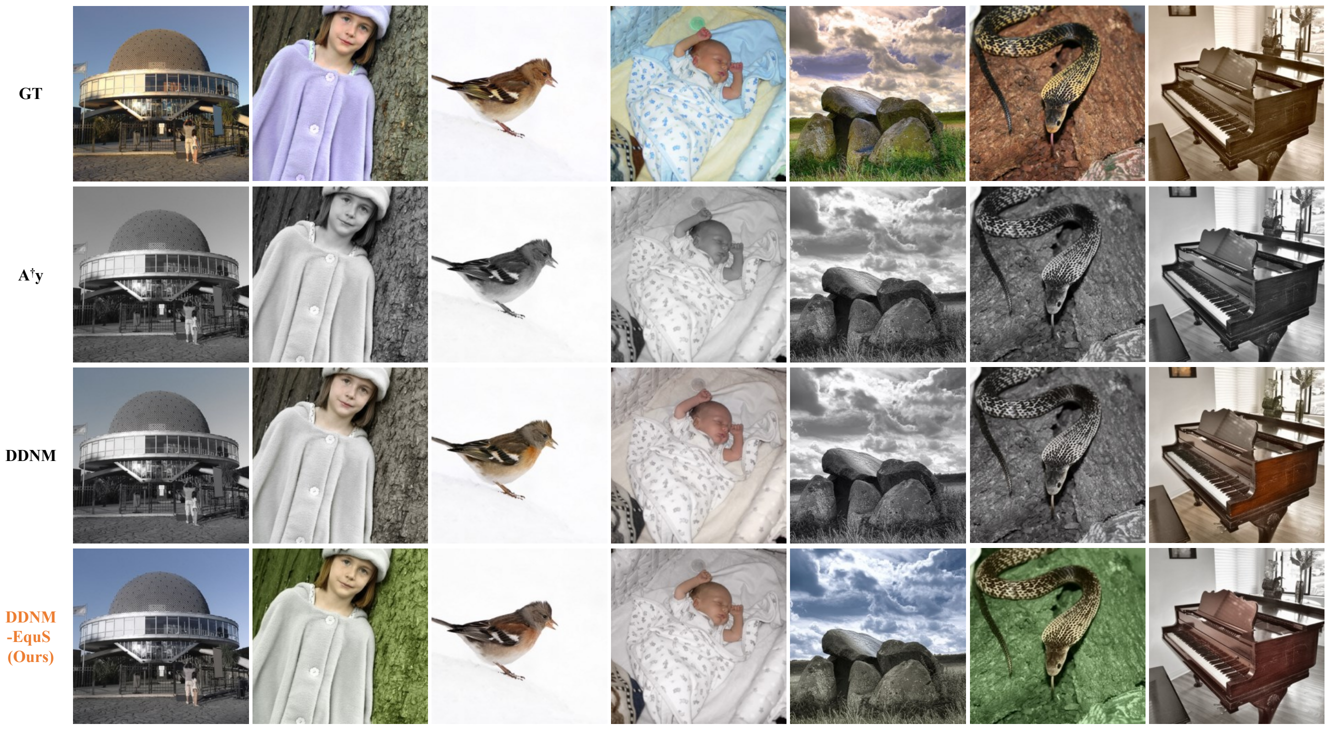

Figure 12. Qualitative results of image colorization on ImageNet.

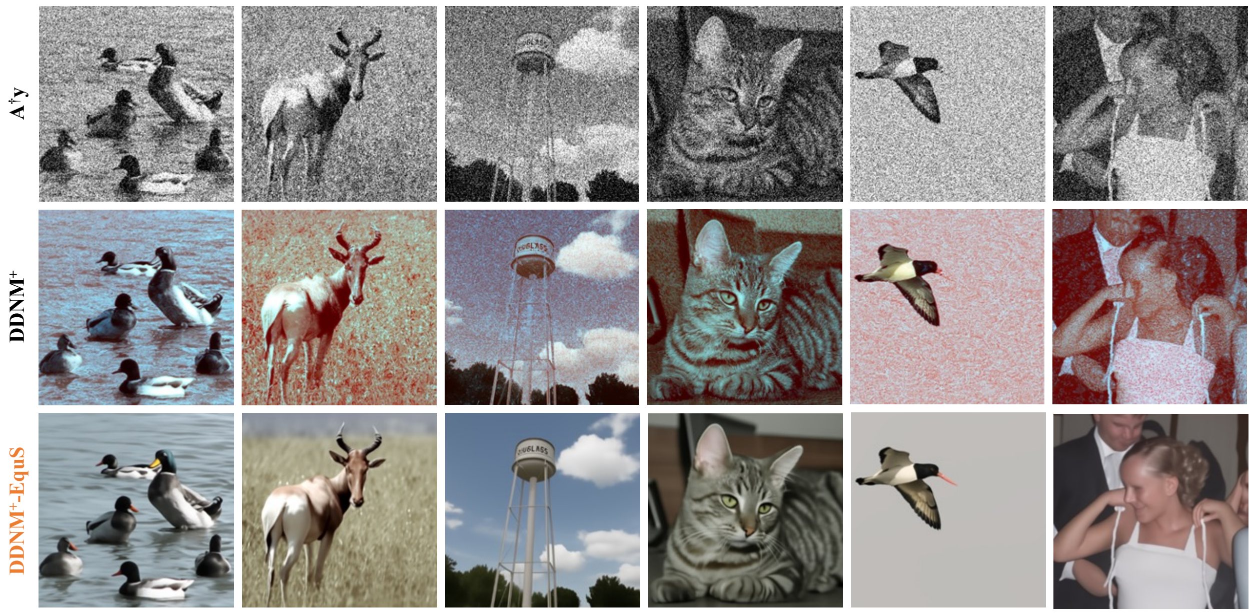

Figure 14. Qualitative results of colorization on Imagenet. Our method enables better visual quality.

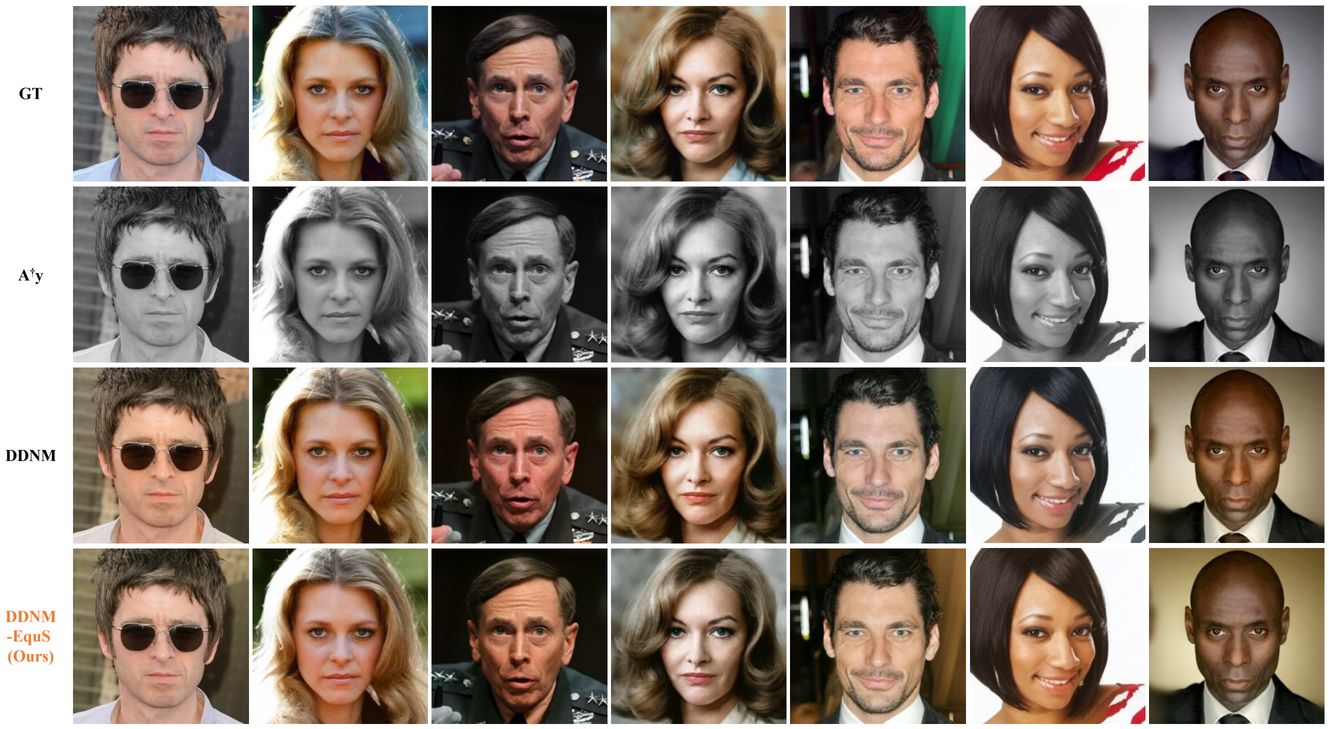

Figure 13. Qualitative results of image colorization on CelebA.

F.6. More Results on Noisy IR Tasks

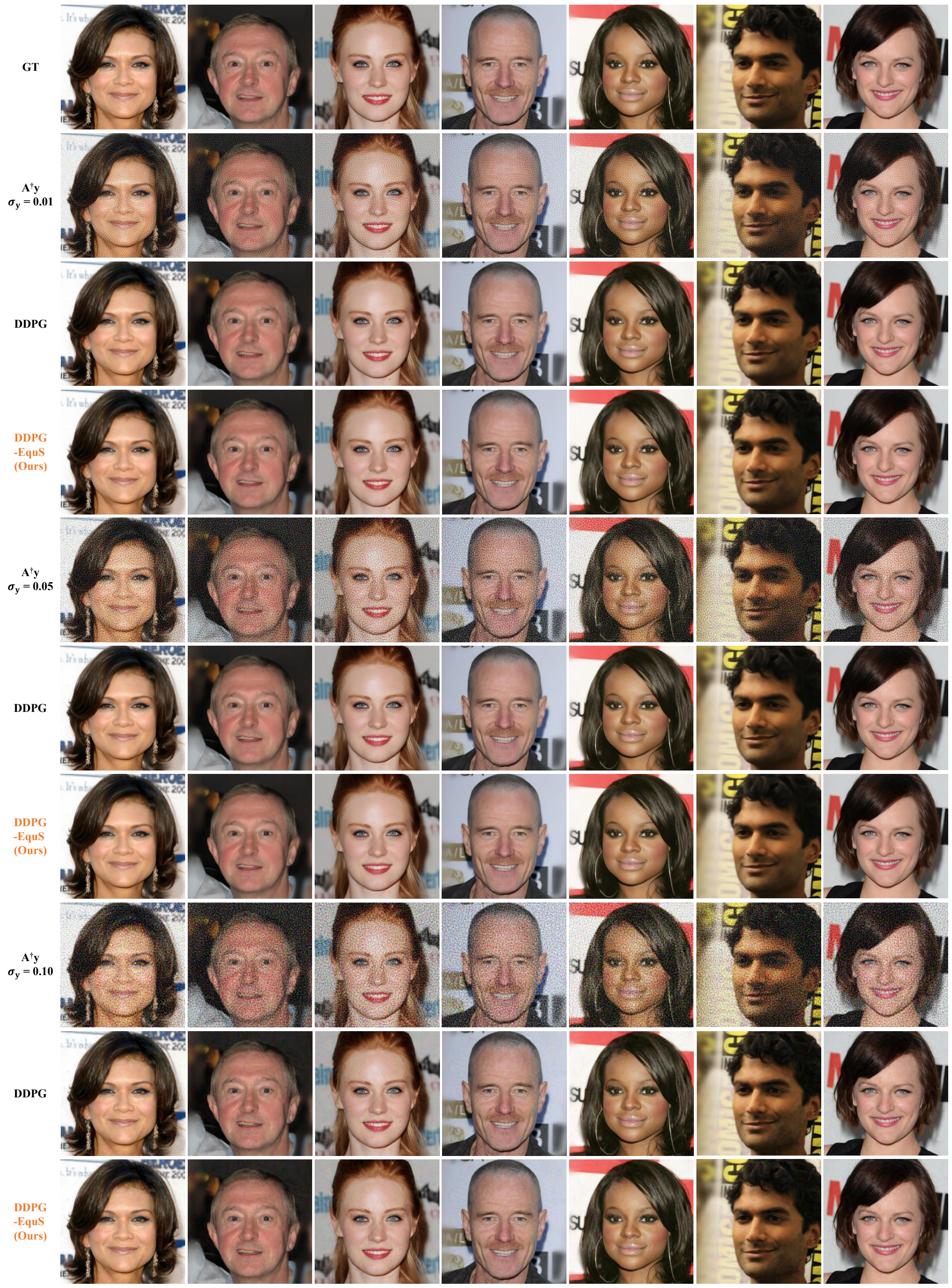

Figure 15. Qualitative results of noisy image gaussian deblurring on CelebA.

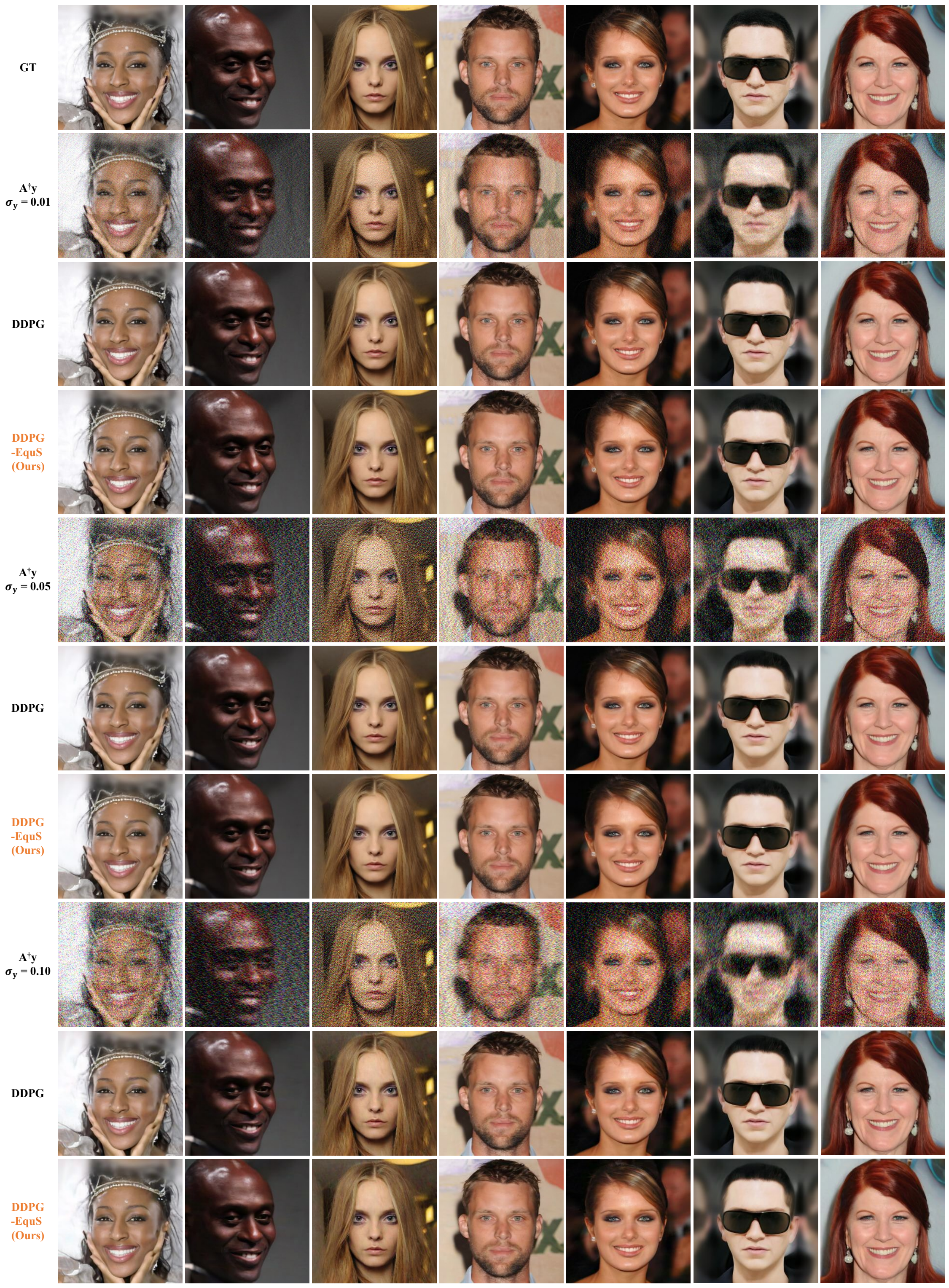
GT
$A^\dagger y$ $\sigma_y = 0.01$
DDPG
DDPG -EquS (Ours)
$A^\dagger y$ $\sigma_y = 0.05$
DDPG
DDPG -EquS (Ours)
$A^\dagger y$ $\sigma_y = 0.10$
DDPG
DDPG -EquS (Ours)

Figure 16. Qualitative results of noisy image motion deblurring on CelebA.

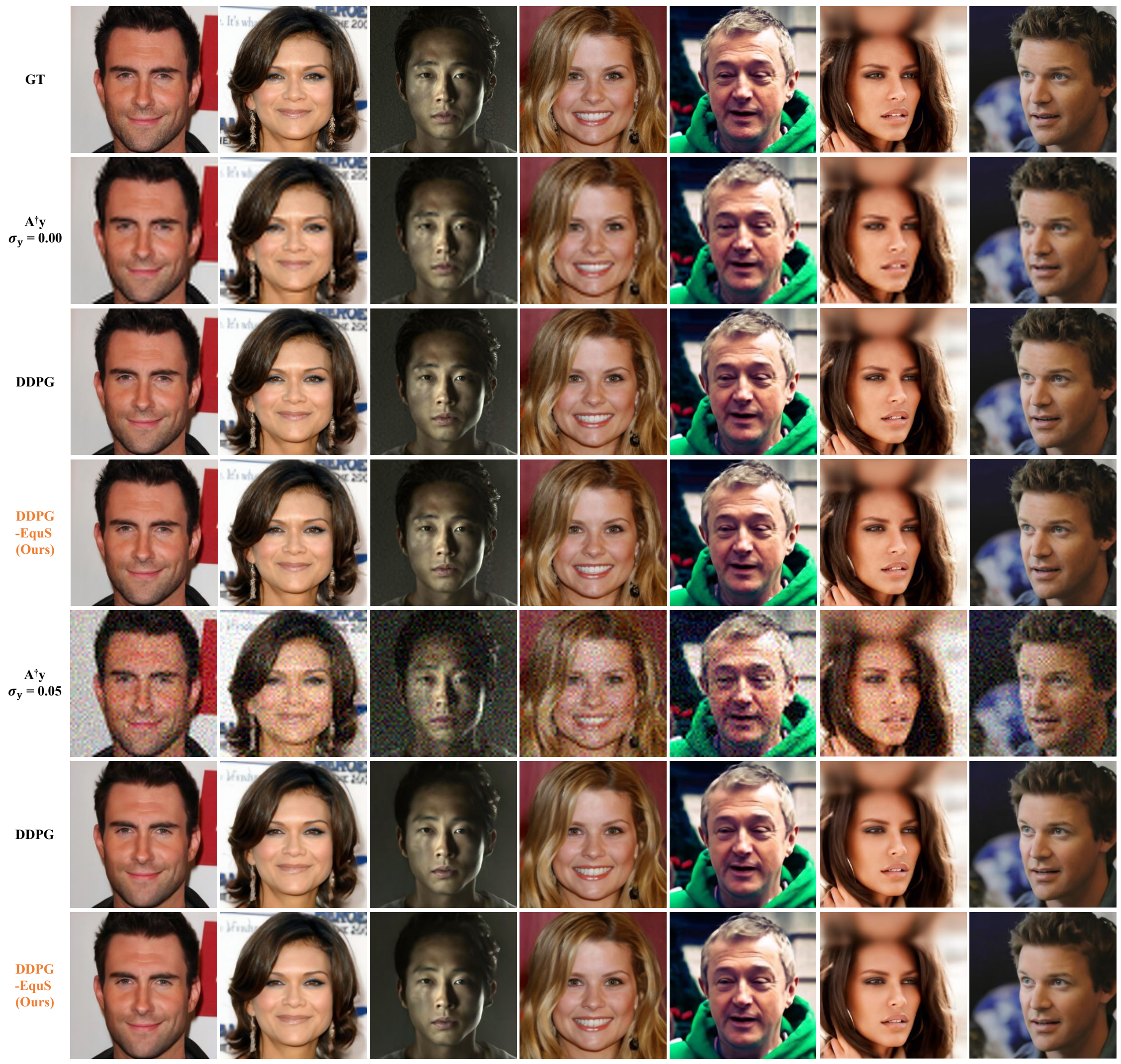

Figure 17. Qualitative results of noisy image super-resolution on CelebA.

G. Limitations and Future Work

There are limitations that can be studied in future work.

- Our method employs a straightforward approach to estimate the averaged equivariant inverse mapping. There exists a potential for improvement through the adoption of more sophisticated estimation techniques.
- Due to the impact of Jacobian vector products on guidance, the existing framework renders EquS unsuitable for DPS and ΠGDM. Future exploration of the generalizability of our approach remains a promising avenue.
- The pseudo-inverse of matrix $\mathbf{A}$ can be computed using Singular Value Decomposition (SVD) or the Moore-Penrose pseudo-inverse. Alternatively, it may be beneficial to leverage a model to learn the pseudo-inverse of $\mathbf{A}$.

References

[1] Yaşar Utku Alçalar and Mehmet Akçakaya. Zero-shot adaptation for approximate posterior sampling of diffusion models in inverse problems. *arXiv preprint arXiv:2407.11288*, 2024. 4, 8, 9

[2] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. 7

[3] Jiezhang Cao, Yue Shi, Kai Zhang, Yulun Zhang, Radu Timofte, and Luc Van Gool. Deep equilibrium diffusion restoration with parallel sampling. In *CVPR*, pages 2824–2834, 2024. 4, 8, 9, 10

[4] Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *ICLR*, 2023. 1, 2, 4, 5, 6, 8, 9, 10

[5] Hyungjin Chung, Jeongsol Kim, and Jong Chul Ye. Direct diffusion bridge using data consistency for inverse problems. *Advances in Neural Information Processing Systems*, 36, 2024. 5, 10

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2009. 8, 9

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 8, 9

[8] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. 2

[9] Tomer Garber and Tom Tirer. Image restoration by denoising diffusion models with iteratively preconditioned guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25245–25254, 2024. 4, 6, 8, 9

[10] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017. 8, 9

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. 2, 3, 7

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3

[13] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005. 7

[14] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International conference on machine learning*, 2018. 8, 9

[15] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 7

[16] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Advances in Neural Information Processing Systems*, 2022. 4, 8, 9, 10

[17] Kwanyoung Kim and Jong Chul Ye. Noise2score: tweedie's approach to self-supervised image denoising without clean images. *Advances in Neural Information Processing Systems*, 34:864–874, 2021. 1, 2

[18] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I2sb: image-to-image schrödinger bridge. In *Proceedings of the 40th International Conference on Machine Learning*, pages 22042–22062, 2023. 5

[19] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 8, 9

[20] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International conference on machine learning*, 2021. 7, 8

[21] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *ICLR*, 2023. 4, 5, 8, 9, 10

[22] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 7

[23] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International conference on machine learning*, 2021. 1, 2, 3, 7

[24] Tom Tirer and Raja Giryes. Image restoration by iterative denoising and backward projections. *IEEE Transactions on Image Processing*, 28(3):1220–1234, 2018. 6

[25] Tom Tirer and Raja Giryes. Super-resolution via image-adapted denoising cnns: Incorporating external and internal learning. *IEEE Signal Processing Letters*, 26(7):1080–1084, 2019. 6

[26] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *International conference on machine learning*, 2023. 3, 4, 8, 9, 10

[27] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 8, 9

[28] Jong Chul Ye. *Geometry of Deep Learning*. Springer, 2022. 5

[29] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 8, 9

[30] Yuanzhi Zhu, Kai Zhang, Jingyun Liang, Jiezhang Cao, Bihan Wen, Radu Timofte, and Luc Van Gool. Denoising diffusion models for plug-and-play image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1219–1229, 2023. 6

[31] Yuanzhi Zhu, Kai Zhang, Jingyun Liang, Jiezhang Cao, Bihan Wen, Radu Timofte, and Luc Van Gool. Denoising diffusion models for plug-and-play image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern RecognitionW*, 2023. 4